Formal Methods in Software Engineering

Satisfiability Modulo Theories — Spring 2025

Konstantin Chukharev

§1 First-Order Theories

Motivation

Consider the signature $\Sigma = \langle \Sigma^S, \Sigma^F \rangle$ for a fragment of number theory:

- $\Sigma^S = \{\texttt{Nat}\}$, $\Sigma^F = \{0, 1, +, <\}$
- $\text{rank}(0) = \text{rank}(1) = \langle \texttt{Nat} \rangle$
- $\text{rank}(+) = \langle \texttt{Nat}, \texttt{Nat}, \texttt{Nat} \rangle$
- $\text{rank}(<) = \langle \texttt{Nat}, \texttt{Nat}, \texttt{Bool} \rangle$

1. Consider the Σ-sentence: $\forall x : \texttt{Nat}. \neg(x < x)$
   - Is it *valid*, that is, true under *all* interpretations?
   - No, e.g., if we interpret $<$ as *equals* or *divides*.

2. Consider the Σ-sentence: $\neg \exists x : \texttt{Nat}. (x < 0)$
   - Is it *valid*?
   - No, e.g., if we interpret Nat as the set of *all* integers.

3. Consider the Σ-sentence: $\forall x : \texttt{Nat}. \forall y : \texttt{Nat}. \forall z : \texttt{Nat}. (x < y) \land (y < z) \to (x < z)$
   - Is it *valid*?
   - No, e.g., if we interpret $<$ as the *successor* relation.

Motivation [2]

In practice, we often *do not care* about satisfiability or validity in *general*, but rather with respect to a *limited class* of interpretations.

A practical reason:

- When reasoning in a particular application domain, we typically have *specific* data types/structures in mind (e.g., integers, strings, lists, arrays, finite sets, …).
- More generally, we are typically *not* interested in *arbitrary* interpretations, but rather in *specific* ones.

Theories formalize this domain-specific reasoning: we talk about satisfiability and validity *with respect to a theory* or "*modulo a theory*".

A computational reason:

- The validity problem for FOL is *undecidable* in general.
- However, the validity problem for many *restricted* theories, is *decidable*.

First-Order Theories

Hereinafter, we assume that we have an infinite set of variables X.

Definition 1 (Theory): A first-order *theory* $\mathcal{T}$ is a pair[1] $\langle \Sigma, \boldsymbol{M} \rangle$, where

- $\Sigma = \langle \Sigma^S, \Sigma^F \rangle$ is a first-order signature,
- $\boldsymbol{M}$ is a class[2] of Σ-interpretations over X that is *closed under variable re-assignment*.

Definition 2: $\boldsymbol{M}$ is *closed under variable re-assignment* if every Σ-interpretation that differs from one in $\boldsymbol{M}$ in the way it interprets the variables in X is also in $\boldsymbol{M}$.

A theory limits the interpretations of Σ-formulas to those from $\boldsymbol{M}$.

[1] Here, we use **bold** style for $\boldsymbol{M}$ to denote that it is *not a single* model, but a *collection* of them.

[2] *Class* is a generalization of a set.

Theory Examples

Example: Theory of Real Arithmetic $\mathcal{T}_{\mathrm{RA}} = \langle \Sigma_{\mathrm{RA}}, M_{\mathrm{RA}} \rangle$:

- $\Sigma^S_{\mathrm{RA}} = \{\texttt{Real}\}$
- $\Sigma^F_{\mathrm{RA}} = \{+, -, *, \leq\} \cup \{q \mid q \text{ is a decimal numeral}\}$
- All $\mathcal{I} \in M_{\mathrm{RA}}$ interpret `Real` as the set of *real numbers* $\mathbb{R}$, each q as the *decimal number* that it denotes, and the function symbols in the usual way.

Example: Theory of Ternary Strings $\mathcal{T}_{\mathrm{TS}} = \langle \Sigma_{\mathrm{TS}}, \boldsymbol{M}_{\mathrm{TS}} \rangle$:

- $\Sigma^S_{\mathrm{TS}} = \{\texttt{String}\}$
- $\Sigma^F_{\mathrm{TS}} = \{\cdot, <\} \cup \{\mathrm{a}, \mathrm{b}, \mathrm{c}\}$
- All $\mathcal{I} \in M_{\mathrm{TS}}$ interpret `String` as the set $\{\mathrm{a}, \mathrm{b}, \mathrm{c}\}^*$ of all finite strings over the characters {“a”, “b”, “c”}, symbol $\cdot$ as string concatenation (e.g., $\mathrm{a} \cdot \mathrm{b} = \mathrm{ab}$), and $<$ as lexicographic order.

$\mathcal{T}$-interpretations

Definition 3 (Reduct): Let Σ and Ω be two signatures over variables X, where $\Omega \supseteq \Sigma$, that is, $\Omega^S \supseteq \Sigma^S$ and $\Omega^F \supseteq \Sigma^F$.

Let $\mathcal{I}$ be an Ω-interpretation over X.

The *reduct* $\mathcal{I}^\Sigma$ of $\mathcal{I}$ to Σ is a Σ-interpretation obtained from $\mathcal{I}$ by resticting it to the symbols in Σ.

Definition 4 ($\mathcal{T}$-interpretation): Given a theory $\mathcal{T} = \langle \Sigma, \boldsymbol{M} \rangle$, a *$\mathcal{T}$-interpretation* is any Ω-interpretation $\mathcal{I}$ for some signature $\Omega \supseteq \Sigma$ such that $\mathcal{I}^\Sigma \in \boldsymbol{M}$.

Note: This definition allows us to consider the satisfiability in a theory $\mathcal{T} = \langle \Sigma, \boldsymbol{M} \rangle$ of formulas that contain sorts or function symbols not in Σ. These symbols are usually called *uninterpreted* (in $\mathcal{T}$).

$\mathcal{T}$-interpretations [2]

Example: Consider again the theory of real arithmetic $\mathcal{T}_{\mathrm{RA}} = \langle \Sigma_{\mathrm{RA}}, \boldsymbol{M}_{\mathrm{RA}} \rangle$.

All $\mathcal{I} \in \boldsymbol{M}_{\mathrm{RA}}$ interpret `Real` as $\mathbb{R}$ and function symbols as usual.

Which of the following interpretations are $\mathcal{T}_{\mathrm{RA}}$-interpretations?

1. $\texttt{Real}^{\mathcal{I}_1} = \mathbb{Q}$, symbols in Σ_{RA}^{F} interpreted as usual. ✗
2. $\texttt{Real}^{\mathcal{I}_2} = \mathbb{R}$, symbols in Σ_{RA}^{F} interpreted as usual, and $\texttt{String}^{\mathcal{I}_2} = \{0.5, 1.3\}$. ✓
3. $\texttt{Real}^{\mathcal{I}_3} = \mathbb{R}$, symbols in Σ_{RA}^{F} interpreted as usual, and $\log^{\mathcal{I}_3}$ is the successor function. ✓

$\mathcal{T}$-satisfiability, $\mathcal{T}$-entailment, $\mathcal{T}$-validity

Definition 5 ($\mathcal{T}$-satisfiability): A Σ-formula α is *satisfiable in $\mathcal{T}$*, or *$\mathcal{T}$-satisfiable*, if it is satisfied by *some* $\mathcal{T}$-interpretation $\mathcal{I}$.

Definition 6 ($\mathcal{T}$-entailment): A set Γ of formulas *$\mathcal{T}$-entails* a formula α, if every $\mathcal{T}$-interpretation that satisfies all formulas in Γ also satisfies α.

Definition 7 ($\mathcal{T}$-validity): A formula α is *$\mathcal{T}$-valid*, if it is satisfied by *all* $\mathcal{T}$-interpretations.

Note: A formula α is *$\mathcal{T}$-valid* iff $\emptyset \vDash \alpha$.

Example: Which of the following Σ_{RA}-formulas is satisfiable or valid in $\mathcal{T}_{\mathrm{RA}}$?

1. $(x_0 + x_1 \leq 0.5) \land (x_0 - x_1 \leq 2)$ — *satisfiable*, *falsifiable*
2. $\forall x_0.\,(x_0 + x_1 \leq 1.7) \to (x_1 \leq 1.7 - x_0)$ — *satisfiable*, *valid*
3. $\forall x_0.\forall x_1.\,(x_0 + x_1 \leq 1)$ — *unsatisfiable*, *falsifiable*

FOL vs Theory

For every signature Σ, entailment and validity in "pure" FOL can be seen as entailment and validity in the theory $\mathcal{T}_{\mathrm{FOL}} = \langle \Sigma, \boldsymbol{M}_{\mathrm{FOL}} \rangle$ where $\boldsymbol{M}_{\mathrm{FOL}}$ is the class of *all possible* Σ-interpretations.

- Pure first-order logic = reasoning over *all* possible interpretations.
- Reasoning modulo a theory = *restricting* interpretations with some domain constraints.
- Theories make automated reasoning *feasible* in many domains.

Axiomatization

Definition 8 (Axiomatic theory): A first-order *axiomatic theory* $\mathcal{T}$ is defined by a signature Σ and a set $\mathcal{A}$ of Σ-sentences, or *axioms*.

Definition 9 ($\mathcal{T}$-validity in axiomatic theory): An Ω-formula α is *valid* in an axiomatic theory $\mathcal{T}$ if it is entailed by the axioms of $\mathcal{T}$, that is, every Ω-interpretation $\mathcal{I}$ that satisfies $\mathcal{A}$ also satisfies α.

Note: Axiomatic theories are a *special case* of the general definition (via $\boldsymbol{M}$) of theories.

- Given an axiomatic theory $\mathcal{T}'$ defined by Σ and $\mathcal{A}$, we can define a theory $\mathcal{T} = \langle \Sigma, \boldsymbol{M} \rangle$ where $\boldsymbol{M}$ is the class of all Σ-interpretations that satisfy all axioms in $\mathcal{A}$.
- It is not hard to show that a formula α is valid in $\mathcal{T}$ *iff* it is valid in $\mathcal{T}'$.

Note: Not all theories are first-order axiomatizable.

Non-Axiomaticizable Theories

Note: Not all theories are first-order axiomatizable.

Example: Consider the theory $\mathcal{T}_{\texttt{Nat}}$ of the natural numbers, with signature Σ with $\Sigma^S = \{\texttt{Nat}\}$, $\Sigma^F = \{0, S, +, <\}$, and $M = \{\mathcal{I}\}$ where $\texttt{Nat}^{\mathcal{I}} = \mathbb{N}$ and Σ^F is interpreted as usual.

Any set of axioms (for example, *Peano axioms*) for this theory is satisfied by *non-standard models*, e.g., interpretations $\mathcal{I}'$ where $\texttt{Nat}^{\mathcal{I}'}$ includes other chains of elements besides the natural numbers.

However, these models *falsify* formulas that are *valid* in $\mathcal{T}_{\texttt{Nat}}$.

For example, "every number is either zero or a successor": $\forall x.\ (x \doteq 0) \lor \exists y.\ (x \doteq S(y))$.

- `true` in the *standard* model, i.e. $\texttt{Nat}^{\mathcal{I}} = \mathbb{N} = \{0, 1 := S(0), 2 := S(1), \ldots\}$.
- `false` in *non-standard* models, e.g., $\texttt{Nat}^{\mathcal{I}'} = \{0, 1, 2, \ldots\} \cup \{\omega, \omega + 1, \ldots\}$
  - Intuitively, ω is "an infinite element".
  - The successor function still applies: $S(\omega) = \omega + 1$, $S(\omega + 1) = \omega + 2$, etc.
  - Even the addition and multiplication still works: $\omega + 3 = S(S(S(\omega)))$, $\omega \times 2 = \omega + \omega$.
  - But ω is larger than all standard numbers: $\omega > 0, \omega > 1, \ldots$

Peano Arithmetic

Definition 10: *Peano arithmetic* $\mathcal{T}_{\text{PA}}$, or *first-order arithmetic*, is the axiomatic theory of natural numbers with signature $\Sigma_{\text{PA}}^{F} = \{0, S, +, \times, =\}$ and *Peano axioms*:

1. $\forall x.\,(S(x) \neq 0)$ (zero)
2. $\forall x. \forall y.\,(S(x) = S(y)) \rightarrow (x = y)$ (successor)
3. $F[0] \land (\forall x.\,F[x] \rightarrow F[x + 1]) \rightarrow \forall x.\,F[x]$ (induction)
4. $\forall x.\,(x + 0 = x)$ (plus zero)
5. $\forall x. \forall y.\,(x + S(y) = S(x + y))$ (plus successor)
6. $\forall x.\,(x \times 0 = 0)$ (times zero)
7. $\forall x. \forall y.\,(x \times S(y) = (x \times y) + x)$ (times successor)

Axiom (induction) is the *induction axiom schema*. It stands for an *infinite* set of axioms, one for each Σ_{PA}-formula F with one free variable. The notation $F[\alpha]$ means that F contains α as a sub-formula.

The *intended interpretation* (*standard models*) of $\mathcal{T}_{\text{PA}}$ have the domain $\mathbb{N}$ and the usual interpretations of the function symbols as $0_{\mathbb{N}}$, $S_{\mathbb{N}}$, $+_{\mathbb{N}}$, and $\times_{\mathbb{N}}$.

Presburger Arithmetic

Note: Satisfiability and validity in $\mathcal{T}_{\mathrm{PA}}$ is undecidable. Therefore, we need a more restricted theory of arithmetic that does not include multiplication.

Definition 11: *Presburger arithmetic* $\mathcal{T}_{\mathbb{N}}$ is the axiomatic theory of natural numbers with signature $\Sigma_{\mathbb{N}}^{F} = \{0, S, +, =\}$ and the *subset* of *Peano axioms*:

1. $\forall x.\, (S(x) \neq 0)$ (zero)
2. $\forall x. \forall y.\, (S(x) = S(y)) \rightarrow (x = y)$ (successor)
3. $F[0] \land (\forall x.\, F[x] \rightarrow F[x + 1]) \rightarrow \forall x.\, F[x]$ (induction)
4. $\forall x.\, (x + 0 = x)$ (plus zero)
5. $\forall x. \forall y.\, (x + S(y) = S(x + y))$ (plus successor)

Note: Presburger arithmetic is decidable.

Completeness of Theories

Definition 12: A Σ-theory $\mathcal{T}$ is *complete* if for every Σ-sentence α, either α or $\neg\alpha$ is valid in $\mathcal{T}$.

Note: In a complete Σ-theory, every Σ-sentence is either valid or unsatisfiable.

Example: Any theory $\mathcal{T} = \langle \Sigma, \boldsymbol{M} \rangle$ where all interpretations in M only differ in how they interpret the variables (e.g., $\mathcal{T}_{\mathrm{RA}}$) is *complete*.

Example: The axiomatic (mono-sorted) theory of *monoids* with $\Sigma^F = \{ \cdot, \varepsilon \}$ and axioms

$$\forall x. \forall y. \forall z. \, (x \cdot y) \cdot z \doteq x \cdot (y \cdot z) \qquad \forall x. \, (x \cdot \varepsilon \doteq x) \qquad \forall x. \, (\varepsilon \cdot x \doteq x)$$

is *incomplete*. For example, the sentence $\forall x. \forall y. \, (x \cdot y \doteq y \cdot x)$ is `true` in some monoids (e.g. the addition of integers *is* commutative) but `false` in others (e.g. the concatenation of strings *is not* commutative).

Completeness of Theories [2]

Example: The axiomatic (mono-sorted) theory of *dense linear orders without endpoints* with $\Sigma^F = \{\prec\}$ and the following axioms is *complete*.

$$\forall x.\forall y.(x \prec y) \to \exists z.\,((x \prec z) \land (z \prec y)) \qquad \text{(dense)}$$

$$\forall x.\forall y.\,((x \prec y) \lor (y \prec x) \lor (x \doteq y)) \qquad \text{(linear)}$$

$$\forall x.\,\neg(x \prec x) \quad \forall x.\forall y.\forall z.\,((x \prec y) \land (y \prec z) \to (x \prec z)) \qquad \text{(orders)}$$

$$\forall x.\exists y.\,(y \prec x) \quad \forall x.\exists y.\,(x \prec y) \qquad \text{(without endpoints)}$$

Decidability

Recall that a set A is *decidable* if there exists a *terminating* procedure that, given an input element a, returns (after *finite* time) either "yes" if $a \in A$ or "no" if $a \notin A$.

Definition 13: A theory $\mathcal{T} = \langle \Sigma, M \rangle$ is *decidable* if the set of all *$\mathcal{T}$-valid* Σ-formulas is decidable.

Definition 14: A *fragment* of $\mathcal{T}$ is a *syntactically-restricted subset* of $\mathcal{T}$-valid Σ-formulas.

Example: The *quantifier-free* fragment of $\mathcal{T}$ is the set of all $\mathcal{T}$-valid Σ-formulas *without quantifiers*.

Example: The *linear* fragment of $\mathcal{T}_{\mathrm{RA}}$ is the set of all $\mathcal{T}$-valid Σ_{RA}-formulas *without multiplication* ($\times$).

Axiomatizability

Definition 15: A theory $\mathcal{T} = \langle \Sigma, M \rangle$ is *recursively axiomatizable* if M is the class of all interpretations satisfying a *decidable set* of first-order axioms $\mathcal{A}$.

Theorem 1 (Lemma): Every recursively axiomatizable theory $\mathcal{T}$ admits a procedure $E_{\mathcal{T}}$ that *enumerates* all $\mathcal{T}$-valid formulas.

Theorem 2: For every *complete* and *recursively axiomatizable* theory $\mathcal{T}$, $\mathcal{T}$-validity is decidable.

Proof: Given a formula α, use $E_{\mathcal{T}}$ to enumerate all valid formulas. Since $\mathcal{T}$ is complete, either α or $\neg\alpha$ will eventually (after *finite* time) be produced by $E_{\mathcal{T}}$. $\square$

§2 Introduction to SMT

Common Theories in SMT

SMT traditionally focuses on theories with *decidable* quantifier-free *fragments*.

Recall: a formula α is *$\mathcal{T}$-valid* iff $\neg\alpha$ is *$\mathcal{T}$-unsatisfiable*.

Checking the (un)satisfiability of quantifier-free formulas in main background theories efficiently has a large number of applications in:

- hardware and software verification
- model checking
- symbolic execution
- compiler validation
- type checking
- planning and scheduling
- software synthesis
- cyber-security
- verifiable machine learning
- analysis of biological systems

Further, we are going to study:

- A few of those theories and their decision procedures.
- Proof systems to reason modulo theories automatically.

From QF to Conjunctions of Literals

Theorem 3: The satisfiability of *quantifier-free* formulas in a theory $\mathcal{T}$ is *decidable* iff the satisfiability in $\mathcal{T}$ of *conjunctions of literals* is decidable.

We will study a general extension of DPLL to SMT that uses decision procedures for *conjunctions of literals*.

Theory of Uninterpreted Functions

Given a signature Σ, the most general theory consists of the class of *all* Σ-interpretations.

In fact, this is a *family* of theories parameterized by the signature Σ.

It is known as the theory of *equality with uninterpreted functions* $\mathcal{T}_{\mathrm{EUF}}$, or the *empty theory*, since it is axiomatized by the empty set of axioms.

Validity, and so satisfiability, in $\mathcal{T}_{\mathrm{EUF}}$ is only *semi-decidable* (this is just a validity in FOL).

However, the satisfiability of *conjunctions $\mathcal{T}_{\mathrm{EUF}}$-literals* is *decidable*, in polynomial time, using the *congruence closure* algorithm.

Example: $(a \doteq b) \land (f(a) \doteq b) \land \neg(g(a) \doteq g(f(a)))$ Is this formula satisfiable in $\mathcal{T}_{\mathrm{EUF}}$?

Theory of Real Arithmetic

The theory of real arithmetic $\mathcal{T}_{\mathrm{RA}}$ is a theory of inequalities over the real numbers.

- $\Sigma^S = \{\texttt{Real}\}$
- $\Sigma^F = \{+, -, \times, <\} \cup \{q \mid q \text{ is a decimal numeral}\}$
- M is the class of interpretations that interpret `Real` as the set of *real numbers* $\mathbb{R}$, and the function symbols in the usual way.

Satisfiability in the full $\mathcal{T}_{\mathrm{RA}}$ is *decidable* (in worst-case doubly-exponential time).

Restricted fragments of $\mathcal{T}_{\mathrm{RA}}$ can be decided more efficiently.

Example: Quantifier-free linear real arithmetic (`QF_LRA`) is the theory of *linear* inequalities over the reals, where $\times$ can only be used in the form of *multiplication by constants (decimal numerals)*.

The satisfiability of conjunctions of literals in `QF_LRA` is *decidable* in *polynomial time*.

Theory of Integer Arithmetic

The theory of integer arithmetic $\mathcal{T}_{\text{IA}}$ is a theory of inequalities over the integers.

- $\Sigma^S = \{\texttt{Int}\}$
- $\Sigma^F = \{+, -, \times, <\} \cup \{n \mid n \text{ is an integer numeral}\}$
- M is the class of interpretations that interpret `Int` as the set of *integers* $\mathbb{Z}$, and the function symbols in the usual way.

Satisfiability in $\mathcal{T}_{\text{IA}}$ is *not even semi-decidable*!

Satisfiability of quantifier-free Σ-formulas in $\mathcal{T}_{\text{IA}}$ is *undecidable* as well.

Linear integer arithmetic (`LIA`, also known as *Presburger arithmetic*) is decidable, but not efficiently (in worst-case triply-exponential time).

Theory of Arrays with Extensionality

The theory of arrays $\mathcal{T}_{\mathrm{A}}$ is useful for modelling RAM or array data structures.

- $\Sigma^S = \{\mathtt{A}, \mathtt{I}, \mathtt{E}\}$ (arrays, indices, elements)
- $\Sigma^F = \{\mathrm{read}, \mathrm{write}\}$, where $\mathrm{rank}(\mathrm{read}) = \langle \mathtt{A}, \mathtt{I}, \mathtt{E} \rangle$ and $\mathrm{rank}(\mathrm{write}) = \langle \mathtt{A}, \mathtt{I}, \mathtt{E}, \mathtt{A} \rangle$

Let a be a variable of sort $\mathtt{A}$, variable i of sort $\mathtt{I}$, and variable v of sort $\mathtt{E}$.

- $\mathrm{read}(a, i)$ denotes the value stored in array a at index i.
- $\mathrm{write}(a, i, v)$ denotes the array that stores value v at index i and is otherwise identical to a.

Example: $\mathrm{read}(\mathrm{write}(a, i, v), i) \doteq_{\mathtt{E}} v$

- Is this formula *intuitively* valid/satisfiable/unsatisfiable in $\mathcal{T}_{\mathrm{A}}$?

Example: $\forall i.\, (\mathrm{read}(a, i) \doteq_{\mathtt{E}} \mathrm{read}(a', i)) \rightarrow (a \doteq_{\mathtt{A}} a')$

- Is this formula *intuitively* valid/satisfiable/unsatisfiable in $\mathcal{T}_{\mathrm{A}}$?

Theory of Arrays with Extensionality [2]

The theory of arrays $\mathcal{T}_{\mathrm{A}} = \langle \Sigma, M \rangle$ is finitely axiomatizable.

M is the class of interpretations that satisfy the following axioms:

1. $\forall a. \forall i. \forall v. \left(\mathrm{read}(\mathrm{write}(a, i, v), i) \doteq_{\mathsf{E}} v \right)$
2. $\forall a. \forall i. \forall j. \forall v. \neg (i \doteq_{\mathsf{I}} j) \rightarrow \left(\mathrm{read}(\mathrm{write}(a, i, v), j) \doteq_{\mathsf{E}} \mathrm{read}(a, j) \right)$
3. $\forall a. \forall b. \left(\forall i. \left(\mathrm{read}(a, i) \doteq_{\mathsf{E}} \mathrm{read}(b, i) \right) \right) \rightarrow (a \doteq_{\mathsf{A}} b)$

Note: The last axiom is called *extensionality* axiom. It states that two arrays are equal if they have the same values at all indices. It can be omitted to obtain a theory of arrays *without extensionality*.

Satisfiability in $\mathcal{T}_{\mathrm{A}}$ is *undecidable*.

There are several *decidable fragments* of $\mathcal{T}_{\mathrm{A}}$.

§3 Extra slides

Decidability and Complexity

Theory	Description	Full	QF	Full complexity	QFC complexity
PL	Propositional Logic	—	yes	NP-complete	$\Theta(n)$
$\mathcal{T}_{\mathrm{EUF}}$	Equality	no	yes	undecidable	$\mathcal{O}(n \log n)$
$\mathcal{T}_{\mathrm{PA}}$	Peano Arithmetic	no	no	undecidable	undecidable
$\mathcal{T}_{\mathbb{N}}$	Presburger Arithmetic	yes	yes	$\Omega(2^{2^n}), \mathcal{O}\left(2^{2^{2^{kn}}}\right)$	NP-complete
$\mathcal{T}_{\mathbb{Z}}$	Linear Integers	yes	yes	$\Omega(2^{2^n}), \mathcal{O}\left(2^{2^{2^{kn}}}\right)$	NP-complete
$\mathcal{T}_{\mathbb{R}}$	Reals (with $\times$)	yes	yes	$\mathcal{O}\left(2^{2^{kn}}\right)$	$\mathcal{O}\left(2^{2^{kn}}\right)$
$\mathcal{T}_{\mathbb{Q}}$	Rationals (without $\times$)	yes	yes	$\Omega(2^n), \mathcal{O}\left(2^{2^{kn}}\right)$	PTIME
$\mathcal{T}_{\mathrm{RDS}}$	Recursive Data Structures	no	yes	undecidable	$\mathcal{O}(n \log n)$
$\mathcal{T}^{+}_{\mathrm{RDS}}$	Acyclic RDS	yes	yes	not elementary recursive	$\Theta(n)$
$\mathcal{T}_{\mathrm{A}}$	Arrays	no	yes	undecidable	NP-complete
$\mathcal{T}^{=}_{\mathrm{A}}$	Arrays with Extensionality	no	yes	undecidable	NP-complete

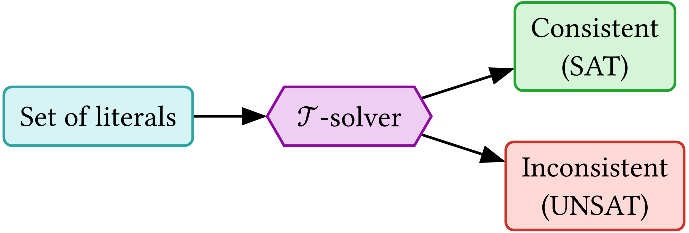
Decidability and Complexity [2]

- **“Full”** denotes the decidability of a complete theory *with* quantifiers.
- **“QF”** denotes the decidability of a *quantifier-free* theory.
- **“Full complexity”** denotes the complexity of the satisfiability in a complete theory *with* quantifiers.
- **“QFC complexity”** denotes the complexity of the satisfiability in a quantifier-free *conjunctive* fragment.
- For complexities, n is the size of the input formula, k is some positive integer.
- “*Not elementary recursive*” means the runtime cannot be bounded by a fixed-height stack of exponentials.

§4 Theory Solvers

Theory Solvers

Definition 16 ($\mathcal{T}$-solver): A *theory solver*, or *$\mathcal{T}$-solver*, is a specialized decision procedure for the satisfiability of conjunctions of literals in a theory $\mathcal{T}$.



Difference Logic

Definition 17: *Difference logic* is a fragment of linear integer arithmetic consisting of conjunctions of literals of the very restricted form:

$$x - y \bowtie c$$

where x and y are integer variables, c is a numeral, and $\bowtie \in \{=, <, \leq, >, \geq\}$.

A solver for difference logic consists of three steps:

1. Literals normalization.
2. Conversion to a graph.
3. Cycle detection.

Difference Logic [2]

Step 1: Rewrite each literal using $\leq$ by applying the following rules:

1. $(x - y = c) \longrightarrow (x - y \leq c) \land (x - y \geq c)$
2. $(x - y \geq c) \longrightarrow (y - x \leq -c)$
3. $(x - y > c) \longrightarrow (y - x < -c)$
4. $(x - y < c) \longrightarrow (x - y \leq c - 1)$

Step 2: Construct a weighted directed graph G with a vertex for each variable and an edge $x \xrightarrow{c} y$ for each literal $(x - y \leq c)$.

Step 3: Check for *negative cycles* in G.

- Use, for example, the Bellman-Ford algorithm.
- If G contains a negative cycle, the set of literals is *inconsistent* (UNSAT).
- Otherwise, the set of literals is *consistent* (SAT).

Difference Logic Example

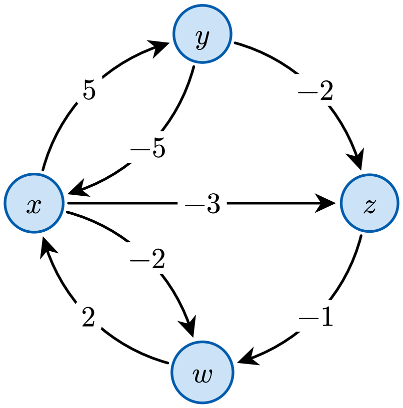
Consider the following set of difference logic literals:

$$(x - y = 5) \land (z - y \geq 2) \land (z - x > 2) \land (w - x = 2) \land (z - w < 0)$$

Normalize the literals:

- $(x - y = 5) \implies (x - y \leq 5) \land (y - x \leq -5)$
- $(z - y \geq 2) \implies (y - z \leq -2)$
- $(z - x > 2) \implies (x - z \leq -3)$
- $(w - x = 2) \implies (w - x \leq 2) \land (x - w \leq -2)$
- $(z - w < 0) \implies (z - w \leq -1)$

UNSAT because of the negative cycle: $x \xrightarrow{-3} z \xrightarrow{-1} w \xrightarrow{2} x$.



§5 Satisfiability Proof Systems

Flattening

Definition 18: A literal is *flat* if it is of the form:

- $x \doteq y$
- $\neg(x \doteq y)$
- $x \doteq f(\boldsymbol{z})$

where x and y are variables, f is a function symbol, and $\boldsymbol{z}$ is a tuple of 0 or more variables.

Note: Any set of literals can be converted to an equisatisfiable set of *flat* literals by introducing *new* variables and equating non-equational atoms to `true`.

Example: Consider the set of literals: $\{x + y > 0, y \doteq f(g(z))\}$.

We can convert it to an equisatisfiable set of flat literals by introducing fresh variables v_1, v_2, v_3 and v_4:

$$\{\, v_1 \doteq v_2 > v_3, \quad v_1 \doteq \texttt{true}, \quad v_2 \doteq x + y, \quad v_3 \doteq 0, \quad y \doteq f(v_4), \quad v_4 \doteq g(z) \,\}$$

Hereinafter, we will assume that all literals are *flat*.

Notation and Assumptions

- We abbreviate $\neg(s \doteq t)$ with $s \not\doteq t$.
- For tuples $\boldsymbol{u} = \langle u_1, \ldots, u_n \rangle$ and $\boldsymbol{v} = \langle v_1, \ldots, v_n \rangle$, we abbreviate $(u_1 \doteq v_1) \wedge \ldots \wedge (u_n \doteq v_n)$ with $\boldsymbol{u} = \boldsymbol{v}$.
- Γ is used to refer to the "current" proof state in rule premises.
- $\Gamma, s \doteq t$ is an abbreviation for $\Gamma \cup \{s \doteq t\}$.
- If applying a rule R does not change Γ, then R *is not applicable* to Γ, that is, Γ is *irreducible* w.r.t. R.

Theory of Equiality with Uninterpreted Functions

Definition 19: The theory of equality with uninterpreted functions $\mathcal{T}_{\mathrm{EUF}}$ is defined by the signature $\Sigma^F = \{\doteq, f, g, h, \ldots\}$ (*interpreted* equiality and *uninterpreted* functions) and the following axioms:

- $\forall x.\, x \doteq x$ (reflexivity)
- $\forall x. \forall y.\, (x \doteq y) \rightarrow (y \doteq x)$ (symmetry)
- $\forall x. \forall y. \forall z.\, (x \doteq y) \land (y \doteq z) \rightarrow (x \doteq z)$ (transitivity)
- $\forall \boldsymbol{x}. \forall \boldsymbol{y}.\, \left(\bigwedge_{i=1}^{n} x_i \doteq y_i\right) \rightarrow (f(\boldsymbol{x}) \doteq f(\boldsymbol{y}))$ (function congruence)

A Satisfiability Proof System for `QF_UF`

Let `QF_UF` be the quantifier-free fragment of FOL over some signature Σ.

Below is a simple satisfiability proof system R_{UF} for `QF_UF`:

$$\textsc{Refl}\ \frac{x \text{ occurs in } \Gamma}{\Gamma := \Gamma, x \doteq x} \qquad \textsc{Symm}\ \frac{x \not\doteq y \in \Gamma}{\Gamma := \Gamma, y \doteq x}$$

$$\textsc{Trans}\ \frac{x \not\doteq y \in \Gamma \quad y \doteq z \in \Gamma}{\Gamma := \Gamma, x \doteq z} \qquad \textsc{Cong}\ \frac{x \doteq f(\boldsymbol{u}) \in \Gamma \quad y \doteq f(\boldsymbol{v}) \in \Gamma \quad \boldsymbol{u} = \boldsymbol{v} \in \Gamma}{\Gamma := \Gamma, x \doteq y}$$

$$\textsc{Contr}\ \frac{x \doteq y \in \Gamma \quad x \not\doteq y \in \Gamma}{\text{UNSAT}} \qquad \textbf{SAT}\ \frac{\text{No other rules apply}}{\text{SAT}}$$

Is R_{UF} *sound*?

Is R_{UF} *terminating*?

Example Derivation in R_{UF}

$$\textsc{Refl}\ \frac{x \text{ occurs in } \Gamma}{\Gamma := \Gamma, x \doteq x} \qquad \textsc{Contr}\ \frac{x \doteq y \in \Gamma \qquad x \not\doteq y \in \Gamma}{\text{UNSAT}} \qquad \textsc{Trans}\ \frac{x \doteq y \in \Gamma \qquad y \doteq z \in \Gamma}{\Gamma := \Gamma, x \doteq z}$$

$$\textsc{Symm}\ \frac{x \doteq y \in \Gamma}{\Gamma := \Gamma, y \doteq x} \qquad \textsc{Cong}\ \frac{x \doteq f(\boldsymbol{u}) \in \Gamma \qquad y \doteq f(\boldsymbol{v}) \in \Gamma \qquad \boldsymbol{u} = \boldsymbol{v} \in \Gamma}{\Gamma := \Gamma, x \doteq y} \qquad \textbf{SAT}\ \frac{\text{No other rules apply}}{\text{SAT}}$$

Example: Determine the satisfiability of the following set of literals: $a \doteq f(f(a))$, $a \doteq f(f(f(a)))$, $g(a, f(a)) \not\doteq g(f(a), a)$. Flatten the literals and construct the following proof:

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{a \doteq f(a_1), a_1 \doteq f(a), a \doteq f(a_2), a_2 \doteq f(a_1), a_3 \not\doteq a_4, a_3 \doteq g(a, a_1), a_4 \doteq g(a_1, a)}{a_1 \doteq a_1}\ \textsc{Refl}}{a \doteq a_2}\ \textsc{Cong} \text{ applied to } a \doteq f(a_1), a_2 \doteq f(a_1), a_1 \doteq a_1}{a_1 \doteq a}\ \textsc{Cong} \text{ applied to } a_1 \doteq f(a), a \doteq f(a_2), a \doteq a_2}{a \doteq a_1}\ \textsc{Symm}}{a_3 \doteq a_4}\ \textsc{Cong} \text{ applied to } a_3 \doteq g(a, a_1), a_4 \doteq g(a_1, a), a \doteq a_1, a_1 \doteq a}{\text{UNSAT}}\ \textsc{Contr} \text{ applied to } a_3 \doteq a_4, a_3 \not\doteq a_4$$

Soundness of R_{UF}

Theorem 4 (Refutation soundness): A literal set Γ_0 is unsatisfiable if R_{UF} derives UNSAT from it.

Proof: All rules except SAT are satisfiability-preserving.

If a derivation from Γ_0 ends with UNSAT, then Γ_0 must be unsatisfiable. $\square$

Theorem 5 (Solution soundness): A literal set Γ_0 is satisfiable if R_{UF} derives SAT from it.

Proof: Let Γ be a proof state to which SAT applies. From Γ, we can construct an interpretation $\mathcal{I}$ that satisfies Γ_0. Let $s \sim t$ iff $(s \doteq t) \in \Gamma$. One can show that $\sim$ is an equivalence relation.

Let the domain of $\mathcal{I}$ be the equivalence classes $E_1, \ldots, E_k$ of $\sim$.
- For every variable or a constant t, let $t^{\mathcal{I}} = E_i$ if $t \in E_i$ for some i. Otherwise, let $t^{\mathcal{I}} = E_1$.
- For every unary function symbol f, and equivalence class E_i, let $f^{\mathcal{I}}$ be such that $f^{\mathcal{I}}(E_i) = E_j$ if $f(t) \in E_j$ for some $t \in E_i$. Otherwise, let $f^{\mathcal{I}}(E_i) = E_1$. Define $f^{\mathcal{I}}$ for non-unary f similarly.

We can show that $\mathcal{I} \models \Gamma$. This means that $\mathcal{I}$ models Γ_0 as well since $\Gamma_0 \subseteq \Gamma$. $\square$

Termination in R_{UF}

Theorem 6: Every derivation strategy for R_{UF} terminates.

Proof: R_{UF} adds to the current state Γ only equalities between variables of Γ_0.

So, at some point it will run out of new equalities to add. $\square$

Completeness of R_{UF}

Theorem 7 (Refutation completeness): Every derivation strategy applied to an unsatisfiable state Γ_0 ends with UNSAT.

Proof: Let Γ_0 be an unsatisfiable state. Suppose there was a derivation from Γ_0 that did not end with UNSAT. Then, by the termination theorem, it would have to end with SAT. But then R_{UF} would be not be solution sound. □

Theorem 8 (Solution completeness): Every derivation strategy applied to a satisfiable state Γ_0 ends with SAT.

Proof: Let Γ_0 be a satisfiable state. Suppose there was a derivation from Γ_0 that did not end with SAT. Then, by the termination theorem, it would have to end with UNSAT. But then R_{UF} would be not be refutation sound. □

TODO

- theory of arrays $\mathcal{T}_{\mathrm{A}}$
- satisfiability proof system for $\mathcal{T}_{\mathrm{A}}$
- soundness, termination, completeness
- LRA, Linear programming, Simplex algorithm
- Strings solver