Automated Deduction

Laura Kovács
TU Wien

Outline

Theories

When analysing programs one has to deal with logical and mathematical concepts beyond propositional logic. For example, statements about program properties can include assertions about various data types, for example

- Equality: $f(a) = a$;
- Arithmetic: $x + x = 2 \cdot x$;
- Arrays: $read(write(a, i, 4), i) = 4$;
- Records: $pair(x_1, y_2) = pair(x_2, y_2) \rightarrow x_1 = x_2 \wedge y_1 = y_2$;
- ...

FOL: Differences and Similarities

These statements are written in (a fragment of) First-Order Logic (FOL).

FOL: Differences and Similarities

These statements are written in (a fragment of) First-Order Logic (FOL).
Consider a system of equations over integers:

$$x + 2y = 0$$
$$y + 2z = 0$$

which we can also write down as a formula

$$x + 2y = 0 \wedge y + 2z = 0$$

FOL: Differences and Similarities

These statements are written in (a fragment of) First-Order Logic (FOL).
Consider a system of equations over integers:

$$x + 2y = 0$$
$$y + 2z = 0$$

which we can also write down as a formula

$$x + 2y = 0 \land y + 2z = 0$$

1. connectives ($\land$);

FOL: Differences and Similarities

These statements are written in (a fragment of) First-Order Logic (FOL). Consider a system of equations over integers:

$$x + 2y = 0$$
$$y + 2z = 0$$

which we can also write down as a formula

$$x + 2y = 0 \wedge y + 2z = 0$$

1. connectives ($\wedge$);
2. variables (x, y, z);

FOL: Differences and Similarities

These statements are written in (a fragment of) First-Order Logic (FOL).
Consider a system of equations over integers:

$$x + 2y = 0$$
$$y + 2z = 0$$

which we can also write down as a formula

$$x + 2y = 0 \land y + 2z = 0$$

1. connectives ($\land$);
2. variables (x, y, z);
3. atomic formulas are complex;

FOL: Differences and Similarities

These statements are written in (a fragment of) First-Order Logic (FOL).
Consider a system of equations over integers:

$$x + 2y = 0$$
$$y + 2z = 0$$

which we can also write down as a formula

$$x + 2y = 0 \wedge y + 2z = 0$$

1. connectives ($\wedge$);
2. variables (x, y, z);
3. atomic formulas are complex;
4. variables range over infinite sets of values;

FOL: Differences and Similarities

These statements are written in (a fragment of) First-Order Logic (FOL).
Consider a system of equations over integers:

$$x + 2y = 0$$
$$y + 2z = 0$$

which we can also write down as a formula

$$x + 2y = 0 \wedge y + 2z = 0$$

1. connectives ($\wedge$);
2. variables (x, y, z);
3. atomic formulas are complex;
4. variables range over infinite sets of values;
5. solutions/interpretations assign values to variables;

FOL: Differences and Similarities

These statements are written in (a fragment of) First-Order Logic (FOL).
Consider a system of equations over integers:

$$\begin{aligned} x + 2y = 0 \\ y + 2z = 0 \end{aligned}$$

which we can also write down as a formula

$$x + 2y = 0 \wedge y + 2z = 0$$

1. connectives ($\wedge$);
2. variables (x, y, z);
3. atomic formulas are complex;
4. variables range over infinite sets of values;
5. solutions/interpretations assign values to variables;
6. infinite number of solutions;

FOL: Differences and Similarities

These statements are written in (a fragment of) First-Order Logic (FOL).
Consider a system of equations over integers:

$$x + 2y = 0$$
$$y + 2z = 0$$

which we can also write down as a formula

$$x + 2y = 0 \wedge y + 2z = 0$$

1. connectives ($\wedge$);
2. variables (x, y, z);
3. atomic formulas are complex;
4. variables range over infinite sets of values;
5. solutions/interpretations assign values to variables;
6. infinite number of solutions;
7. constants ($0, 2$);

FOL: Differences and Similarities

These statements are written in (a fragment of) First-Order Logic (FOL).
Consider a system of equations over integers:

$$x + 2y = 0$$
$$y + 2z = 0$$

which we can also write down as a formula

$$x + 2y = 0 \land y + 2z = 0$$

1. connectives ($\land$);
2. variables (x, y, z);
3. atomic formulas are complex;
4. variables range over infinite sets of values;
5. solutions/interpretations assign values to variables;
6. infinite number of solutions;
7. constants ($0, 2$);
8. functions ($+, \cdot$)

Satisfiability Modulo Theories (SMT): an Example

$$x + 2 = y \rightarrow f(read(write(a, x, 3), y - 2)) = f(y - x + 1)$$

Satisfiability Modulo Theories (SMT): an Example

$$x + 2 = y \rightarrow f(read(write(a, x, 3), y - 2)) = f(y - x + 1)$$

- arithmetic;

Satisfiability Modulo Theories (SMT): an Example

$$x + 2 = y \rightarrow f(read(write(a, x, 3), y - 2)) = f(y - x + 1)$$

- arithmetic;
- arrays;

Satisfiability Modulo Theories (SMT): an Example

$$x + 2 = y \rightarrow f(read(write(a, x, 3), y - 2)) = f(y - x + 1)$$

- arithmetic;
- arrays;
- equality and uninterpreted functions.

Satisfiability Modulo Theories (SMT): an Example

$$x + 2 = y \rightarrow f(read(write(a, x, 3), y - 2)) = f(y - x + 1)$$

- arithmetic;
- arrays;
- equality and uninterpreted functions.

- Some symbols are interpreted: $2, 3, +, -, read, write$.

Satisfiability Modulo Theories (SMT): an Example

$$x + 2 = y \rightarrow f(read(write(a, x, 3), y - 2)) = f(y - x + 1)$$

- arithmetic;
- arrays;
- equality and uninterpreted functions.

- Some symbols are interpreted: $2, 3, +, -, read, write$.
- Some symbols are uninterpreted: f, x, y, a.

Satisfiability Modulo Theories (SMT): an Example

$$x + 2 = y \rightarrow f(read(write(a, x, 3), y - 2)) = f(y - x + 1)$$

- arithmetic;
- arrays;
- equality and uninterpreted functions.

- Some symbols are interpreted: $2, 3, +, -, read, write$.
- Some symbols are uninterpreted: f, x, y, a.
- Sorts:
  - $x, y : \mathbb{Z}$;
  - $a : Array(\mathbb{Z})$;
  - $f : \mathbb{Z} \rightarrow ???$.

Semantics: Interpretation

$$x + 2 = y \rightarrow f(read(write(a, x, 3), y - 2)) = f(y - x + 1)$$

- Interpretation maps uninterpreted symbols to values;
- Interpretation must respect sorts, for example, an integer variable is mapped to an integer value;
- Interpreted symbols are interpreted in the respective theory, for example $+$ is always interpreted as the integer addition.

First-Order Logic

- Language: variables, function and predicate (relation) symbols. Each symbol has an associated arity, denoting the number of arguments. A constant symbol a function symbol of arity 0.

First-Order Logic

- Language: variables, function and predicate (relation) symbols. Each symbol has an associated arity, denoting the number of arguments. A constant symbol a function symbol of arity 0.
- Terms: variables, constants, and expressions $f(t_1, \ldots, t_n)$, where f is a function symbol of arity n and $t_1, \ldots, t_n$ are terms.

First-Order Logic

- Language: variables, function and predicate (relation) symbols. Each symbol has an associated arity, denoting the number of arguments. A constant symbol a function symbol of arity 0.
- Terms: variables, constants, and expressions $f(t_1, \ldots, t_n)$, where f is a function symbol of arity n and $t_1, \ldots, t_n$ are terms. Terms denote domain (universe) elements (objects).

First-Order Logic

- Language: variables, function and predicate (relation) symbols. Each symbol has an associated arity, denoting the number of arguments. A constant symbol a function symbol of arity 0.
- Terms: variables, constants, and expressions $f(t_1, \ldots, t_n)$, where f is a function symbol of arity n and $t_1, \ldots, t_n$ are terms. Terms denote domain (universe) elements (objects).
- Atomic formula: expression $p(t_1, \ldots, t_n)$, where p is a predicate symbol of arity n and $t_1, \ldots, t_n$ are terms.

First-Order Logic

- Language: variables, function and predicate (relation) symbols. Each symbol has an associated arity, denoting the number of arguments. A constant symbol a function symbol of arity 0.
- Terms: variables, constants, and expressions $f(t_1, \ldots, t_n)$, where f is a function symbol of arity n and $t_1, \ldots, t_n$ are terms. Terms denote domain (universe) elements (objects).
- Atomic formula: expression $p(t_1, \ldots, t_n)$, where p is a predicate symbol of arity n and $t_1, \ldots, t_n$ are terms. Formulas denote properties of domain elements.

First-Order Logic

- Language: variables, function and predicate (relation) symbols. Each symbol has an associated arity, denoting the number of arguments. A constant symbol a function symbol of arity 0.
- Terms: variables, constants, and expressions $f(t_1, \ldots, t_n)$, where f is a function symbol of arity n and $t_1, \ldots, t_n$ are terms. Terms denote domain (universe) elements (objects).
- Atomic formula: expression $p(t_1, \ldots, t_n)$, where p is a predicate symbol of arity n and $t_1, \ldots, t_n$ are terms. Formulas denote properties of domain elements.
- Some symbols are interpreted, such as $=$ or $>$, other symbols uninterpreted.

First-Order Logic: Syntax

A signature consisting of

- A set of sorts (for example, integers, rationals, arrays of integers). Sorts will be denoted by α, β.

First-Order Logic: Syntax

A signature consisting of

- A set of sorts (for example, integers, rationals, arrays of integers). Sorts will be denoted by α, β.
- Constants, denoted by a, b, c. Each constant c has a sort α, written $c : \alpha$.

First-Order Logic: Syntax

A signature consisting of

- A set of sorts (for example, integers, rationals, arrays of integers). Sorts will be denoted by α, β.
- Constants, denoted by a, b, c. Each constant c has a sort α, written $c : \alpha$.
- Function symbols, denoted by f, g, each function symbol f has a type $\alpha_1 \times \ldots \times \alpha_n \to \alpha$.

First-Order Logic: Syntax

A signature consisting of

- A set of sorts (for example, integers, rationals, arrays of integers). Sorts will be denoted by α, β.
- Constants, denoted by a, b, c. Each constant c has a sort α, written $c : \alpha$.
- Function symbols, denoted by f, g, each function symbol f has a type $\alpha_1 \times \ldots \times \alpha_n \to \alpha$.
- Predicate symbols, denoted by p, q, each predicate symbol p has a type $\alpha_1 \times \ldots \times \alpha_n$.

First-Order Logic: Syntax

A signature consisting of

- A set of sorts (for example, integers, rationals, arrays of integers). Sorts will be denoted by α, β.
- Constants, denoted by a, b, c. Each constant c has a sort α, written $c : \alpha$.
- Function symbols, denoted by f, g, each function symbol f has a type $\alpha_1 \times \ldots \times \alpha_n \to \alpha$.
- Predicate symbols, denoted by p, q, each predicate symbol p has a type $\alpha_1 \times \ldots \times \alpha_n$.

Variables are not part of the signature.

First-Order Logic: Syntax

A signature consisting of

- A set of sorts (for example, integers, rationals, arrays of integers). Sorts will be denoted by α, β.
- Constants, denoted by a, b, c. Each constant c has a sort α, written $c : \alpha$.
- Function symbols, denoted by f, g, each function symbol f has a type $\alpha_1 \times \ldots \times \alpha_n \to \alpha$.
- Predicate symbols, denoted by p, q, each predicate symbol p has a type $\alpha_1 \times \ldots \times \alpha_n$.

Variables are not part of the signature.

Variables have sorts.

First-Order Logic: Semantics

An interpretation I maps

- Each sort to a non-empty set, called the *domain* for this sort.

First-Order Logic: Semantics

An interpretation I maps

- Each sort to a non-empty set, called the *domain* for this sort.
- Each constant $c : \alpha$ to an element $c^I \in I(\alpha)$.
- Each variable $x : \alpha$ to an element $x^I \in I(\alpha)$.

First-Order Logic: Semantics

An interpretation I maps

- Each sort to a non-empty set, called the *domain* for this sort.
- Each constant $c : \alpha$ to an element $c^I \in I(\alpha)$.
- Each variable $x : \alpha$ to an element $x^I \in I(\alpha)$.
- Each function symbol $f : \alpha_1 \times \ldots \times \alpha_n \to \alpha$ to a function $f^I : I(\alpha_1) \times \ldots \times I(\alpha_n) \to I(\alpha)$;

First-Order Logic: Semantics

An interpretation I maps

- Each sort to a non-empty set, called the *domain* for this sort.
- Each constant $c : \alpha$ to an element $c^I \in I(\alpha)$.
- Each variable $x : \alpha$ to an element $x^I \in I(\alpha)$.
- Each function symbol $f : \alpha_1 \times \ldots \times \alpha_n \rightarrow \alpha$ to a function $f^I : I(\alpha_1) \times \ldots \times I(\alpha_n) \rightarrow I(\alpha)$;
- Each predicate symbol $p : \alpha_1 \times \ldots \times \alpha_n$ to a relation p^I on $I(\alpha_1) \times \ldots \times I(\alpha_n)$;

First-Order Logic: Syntax

Terms:

- a constant $c : \alpha$ or a variable $x : \alpha$ is a term of the sort α;

First-Order Logic: Syntax

Terms:

- a constant $c : \alpha$ or a variable $x : \alpha$ is a term of the sort α;
- If $t_1, \ldots, t_n$ are terms of the sorts $\alpha_1, \ldots, \alpha_n$ and $f : \alpha_1 \times \ldots \times \alpha_n \to \alpha$, then $f(t_1, \ldots, t_n)$ is a term of the sort α;

First-Order Logic: Syntax

Terms:

- a constant $c : \alpha$ or a variable $x : \alpha$ is a term of the sort α;
- If $t_1, \ldots, t_n$ are terms of the sorts $\alpha_1, \ldots, \alpha_n$ and $f : \alpha_1 \times \ldots \times \alpha_n \to \alpha$, then $f(t_1, \ldots, t_n)$ is a term of the sort α;

Atomic formulas:

- If $t_1, \ldots, t_n$ are terms of the sorts $\alpha_1, \ldots, \alpha_n$ and $p : \alpha_1 \times \ldots \times \alpha_n$, then $p(t_1, \ldots, t_n)$ is an atomic formula.

First-Order Logic: Syntax

Terms:

- a constant $c : \alpha$ or a variable $x : \alpha$ is a term of the sort α;
- If $t_1, \ldots, t_n$ are terms of the sorts $\alpha_1, \ldots, \alpha_n$ and $f : \alpha_1 \times \ldots \times \alpha_n \rightarrow \alpha$, then $f(t_1, \ldots, t_n)$ is a term of the sort α;

Atomic formulas:

- If $t_1, \ldots, t_n$ are terms of the sorts $\alpha_1, \ldots, \alpha_n$ and $p : \alpha_1 \times \ldots \times \alpha_n$, then $p(t_1, \ldots, t_n)$ is an atomic formula.

We can build non-atomic formulas using connectives, as before. For example, if A and B are formulas, then $A \rightarrow B$ is a formula.

Quantifiers

- Let A be a formula and x a variable. Then $\forall x A$ and $\exists x A$ are formulas.

Quantifiers

- Let A be a formula and x a variable. Then $\forall x A$ and $\exists x A$ are formulas.
- The symbols $\forall$ and $\exists$ are called quantifiers.

Quantifiers

- Let A be a formula and x a variable. Then $\forall x A$ and $\exists x A$ are formulas.
- The symbols $\forall$ and $\exists$ are called quantifiers.
- A variable occurrence x in a formula A is called bound, if it is in the scope of a quantifier $\forall x$ or $\exists x$, otherwise it is called free.

Quantifiers

- Let A be a formula and x a variable. Then $\forall x A$ and $\exists x A$ are formulas.
- The symbols $\forall$ and $\exists$ are called quantifiers.
- A variable occurrence x in a formula A is called bound, if it is in the scope of a quantifier $\forall x$ or $\exists x$, otherwise it is called free.
- A formula is called quantifier-free if it contains no occurrences of quantifiers.

First-Order Logic: Semantics of Terms

We will now extend interpretations to terms and formulas.

Let I be an interpretation and t a term of a sort α. Define an element $t^I \in I(\alpha)$ as follows.

First-Order Logic: Semantics of Terms

We will now extend interpretations to terms and formulas.

Let I be an interpretation and t a term of a sort α. Define an element $t^I \in I(\alpha)$ as follows.

- for constants $c : \alpha$ and variables $x : \alpha$ we have $c^I \stackrel{\text{def}}{\Leftrightarrow} I(c)$ and $x^I \stackrel{\text{def}}{\Leftrightarrow} I(x)$.
- $f(t_1, \ldots, t_n)^I \stackrel{\text{def}}{\Leftrightarrow} f^I(t_1^I, \ldots, t_n^I)$.

First-Order Logic: Semantics of Formulas

Likewise, for every formula A define a boolean value A^I as follows.

For an atomic formula $p(t_1, \ldots, t_n)$ we have

- $p(t_1, \ldots, t_n)^I = 1 \stackrel{\text{def}}{\Leftrightarrow} (t_1^I, \ldots, t_n^I) \in p^I$.

First-Order Logic: Semantics of Formulas

Likewise, for every formula A define a boolean value A^I as follows.

For an atomic formula $p(t_1, \dots, t_n)$ we have

- $p(t_1, \dots, t_n)^I = 1 \stackrel{\text{def}}{\Leftrightarrow} (t_1^I, \dots, t_n^I) \in p^I$.

For connectives formulas the definition is as usual, for example

- $(A \to B)^I = 1$ if either $A^I = 0$ or $B^I = 1$.

Semantics of Quantifiers

Let $\bar{x}$ be a sequence of variables. We say that two interpretations of the same signature Σ are $\bar{x}$-variants is they coincide on all symbols and all variables not occurring in $\bar{x}$.

Semantics of Quantifiers

Let $\bar{x}$ be a sequence of variables. We say that two interpretations of the same signature Σ are $\bar{x}$-variants is they coincide on all symbols and all variables not occurring in $\bar{x}$.

- $(\forall x A)^I = 1$ if for all $\bar{x}$-variants I' or I we have $(A)^{I'} = 1$

Semantics of Quantifiers

Let $\bar{x}$ be a sequence of variables. We say that two interpretations of the same signature Σ are $\bar{x}$-variants is they coincide on all symbols and all variables not occurring in $\bar{x}$.

- $(\forall x A)^I = 1$ if for all $\bar{x}$-variants I' or I we have $(A)^{I'} = 1$
- $(\exists x A)^I = 1$ if for some $\bar{x}$-variant I' or I we have $(A)^{I'} = 1$

Satisfiability

- A formula A with free variables $\bar{x}$ is said to be satisfiable in an interpretation I if for some $\bar{x}$-variant I' of A we have $I' \models A$.
- A is satisfiable if it is satisfiable in some interpretation.

Validity

- A formula A with free variables $\bar{x}$ is said to be valid in an interpretation I if for every $\bar{x}$-variant I' of A we have $I' \models A$.

Validity

- A formula A with free variables $\bar{x}$ is said to be valid in an interpretation I if for every $\bar{x}$-variant I' of A we have $I' \models A$.
- A formula A is said to be valid if it is valid in every interpretation.

Validity

- A formula A with free variables $\bar{x}$ is said to be valid in an interpretation I if for every $\bar{x}$-variant I' of A we have $I' \models A$.
- A formula A is said to be valid if it is valid in every interpretation.
- A formula A is valid if and only if $\neg A$ is unsatisfiable.

Validity and Satisfiability

The formula

$$x + 2 = y \rightarrow f(\mathit{read}(\mathit{write}(a, x, 3), y - 2)) = f(y - x + 1)$$

is valid if and only if the following set of two unit clauses is unsatisfiable:

$$x + 2 = y$$
$$\neg f(\mathit{read}(\mathit{write}(a, x, 3), y - 2)) = f(y - x + 1)$$

We will write a negation of equality as an inequality:

$$x + 2 = y$$
$$f(\mathit{read}(\mathit{write}(a, x, 3), y - 2)) \neq f(y - x + 1)$$

Interpretation: an example

$$x + 2 = y$$
$$f(read(write(a, x, 3), y - 2)) \neq f(y - x + 1)$$

Take an interpretation I in which

$$x^I = 0$$
$$y^I = 2$$
$$a^I[0] = 0, \ldots$$
$$f^I(3) = \texttt{student@tuwien.ac.at}, \ldots$$

Interpretation: an example

$$x + 2 = y$$
$$f(read(write(a, x, 3), y - 2)) \neq f(y - x + 1)$$

Take an interpretation I in which

$$x^I = 0$$
$$y^I = 2$$
$$a^I[0] = 0, \ldots$$
$$f^I(3) = \texttt{student@tuwien.ac.at}, \ldots$$

- Does this interpretation satisfy the set of clauses?

Interpretation: an example

$$x + 2 = y$$
$$f(read(write(a, x, 3), y - 2)) \neq f(y - x + 1)$$

Take an interpretation I in which

$$x^I = 0$$
$$y^I = 2$$
$$a^I[0] = 0, \ldots$$
$$f^I(3) = \texttt{student@tuwien.ac.at}, \ldots$$

- Does this interpretation satisfy the set of clauses?
- In fact, this set of clauses is unsatisfiable.

Interpretation: an example

$$x + 2 = y$$
$$f(read(write(a, x, 3), y - 2)) \neq f(y - x + 1)$$

Take an interpretation I in which

$$x^I = 0$$
$$y^I = 2$$
$$a^I[0] = 0, \ldots$$
$$f^I(3) = \texttt{student@tuwien.ac.at}, \ldots$$

- Does this interpretation satisfy the set of clauses?
- In fact, this set of clauses is unsatisfiable.
- How can we check for (un)satisfiability automatically?

Theories

Normally, when we discuss satisfiability, we are interested not in arbitrary interpretations, but only in interpretations of a special form. There are two ways of dealing with such a class.

Theories

Normally, when we discuss satisfiability, we are interested not in arbitrary interpretations, but only in interpretations of a special form. There are two ways of dealing with such a class.

Axiomatic: when we write a collection of formulas (called axioms) and say we restrict ourselves to the interpretations that make these formulas valid. For example, the theory of equality for a signature Σ uses the following axioms:

- reflexivity, symmetry and transitivity:

$$
\begin{aligned}
&x = x \\
&x = y \rightarrow y = x \\
&x = y \land y = z \rightarrow x = z
\end{aligned}
$$

- congruence axioms for each function symbol f of Σ:

$$x_1 = y_1 \land \ldots \land x_n = y_n \rightarrow f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n).$$

- congruence axioms for each predicate symbol p of Σ:

$$x_1 = y_1 \land \ldots \land x_n = y_n \land p(x_1, \ldots, x_n) \rightarrow p(y_1, \ldots, y_n).$$

Theories

The second way is to describe the class of interpretations we are interested in. For example, we can say that we are interested in those interpretations in which $=$ is the equality relation. Or in those interpretations where the symbols of arithmetic have the intended meaning.

Theories

The second way is to describe the class of interpretations we are interested in. For example, we can say that we are interested in those interpretations in which $=$ is the equality relation. Or in those interpretations where the symbols of arithmetic have the intended meaning.

The second way is more flexible since not every set of interpretations has an appropriate axiomatisation.

Small Exercise

Example 1

Consider the class of all interpretations I such that

1. I interprets $+$ and 1 in the standard way over the integers.

Is the formula $f(x) = f(x + 1) \wedge f(a) \neq f(b)$ satisfiable for this class of interpretations? If yes, give an interpretation that satisfies this formula.

Small Exercise

Example 1

Consider the class of all interpretations I such that

1. I interprets $+$ and 1 in the standard way over the integers.

Is the formula $f(x) = f(x + 1) \wedge f(a) \neq f(b)$ satisfiable for this class of interpretations? If yes, give an interpretation that satisfies this formula.

Example 2

Consider the class of all interpretations I such that

1. I interprets $+$ and 1 in the standard way over the integers;
2. the formula $f(x) = f(x + 1)$ is valid in I;

Is the formula $f(a) \neq f(b)$ satisfiable for this class of interpretations? If yes, give an interpretation that satisfies this formula.

Outline

First-Order Logic

Theories

Theories – summary

When we discuss satisfiability in a theory, we are interested not in arbitrary interpretations, but only in interpretations of a special form.

Theories – summary

When we discuss satisfiability in a theory, we are interested not in arbitrary interpretations, but only in interpretations of a special form.

One way to deal with such a class is the axiomatic way: we write a collection of formulas (called axioms) and we restrict ourselves to the interpretations that make these formulas valid. For example, the theory of equality for a signature Σ uses the following axioms:

- reflexivity, symmetry and transitivity:

$$\begin{array}{l} x = x \\ x = y \to y = x \\ x = y \wedge y = z \to x = z \end{array}$$

- congruence axioms for each function symbol f of Σ:

$$x_1 = y_1 \wedge \ldots \wedge x_n = y_n \to f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n).$$

- congruence axioms for each predicate symbol p of Σ:

$$x_1 = y_1 \wedge \ldots \wedge x_n = y_n \wedge p(x_1, \ldots, x_n) \to p(y_1, \ldots, y_n).$$

Theories – axiomatizable theories

A theory $\mathcal{T}$ is thus defined by

- a signature $\Sigma_{\mathcal{T}}$
- a set of formulas $A_{\mathcal{T}}$ over $\Sigma_{\mathcal{T}}$, called the axioms of $\mathcal{T}$.

Theories – axiomatizable theories

A theory $\mathcal{T}$ is thus defined by

- a signature $\Sigma_{\mathcal{T}}$
- a set of formulas $A_{\mathcal{T}}$ over $\Sigma_{\mathcal{T}}$, called the axioms of $\mathcal{T}$.

From now on, when we speak about theories, we speak about:

QUANTIFIER-FREE THEORIES!

(Unless stated otherwise.)

Theories – axiomatizable theories

A theory $\mathcal{T}$ is thus defined by

- a signature $\Sigma_{\mathcal{T}}$
- a set of formulas $A_{\mathcal{T}}$ over $\Sigma_{\mathcal{T}}$, called the axioms of $\mathcal{T}$.

From now on, when we speak about theories, we speak about:

QUANTIFIER-FREE THEORIES!

(Unless stated otherwise.)

Examples

- Theory of equality, denoted by $\mathcal{T}_E$;

  Also called as *the theory of equality and uninterpreted functions*.

- Theory of arrays, denoted by $\mathcal{T}_A$;
- Theory of rational numbers, denoted by $\mathcal{T}_Q$.

Example: Theory of Equality

The theory of equality $\mathcal{T}_E$ is defined by

- a signature $\Sigma_E = \{a, b, \ldots, f, g, \ldots, =, p, \ldots\}$
- the previously given five axioms, that is:

$x = x$	(reflexivity)
$x = y \rightarrow y = x$	(symmetry)
$x = y \land y = z \rightarrow x = z$	(transitivity)
$x_1 = y_1 \land \ldots \land x_n = y_n \rightarrow f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)$	(function congruence)
$x_1 = y_1 \land \ldots \land x_n = y_n \land p(x_1, \ldots, x_n) \rightarrow p(y_1, \ldots, y_n)$	(predicate congruence)

Satisfiability Modulo Theory (SMT)

- An interpretation I which which makes all axioms of $\mathcal{T}$ valid, that is $I \models A_{\mathcal{T}}$, is called a $\mathcal{T}$-interpretation.
- A formula F is valid in $\mathcal{T}$ (or $\mathcal{T}$-valid) if F is valid in every $\mathcal{T}$-interpretation, written $\mathcal{T} \models F$.
- A formula F is satisfiable in $\mathcal{T}$ (or $\mathcal{T}$-satisfiable) if there exists a $\mathcal{T}$-interpretation which satisfies F.
- A theory $\mathcal{T}$ is decidable if the set of all F such that $\mathcal{T} \models F$ is decidable, that is there exists a procedure that decides whether $\mathcal{T} \models F$.

Related problems

- Deciding a conjunction of literals: How can we check whether a set of *literals* is satisfiable?

Related problems

- Deciding a conjunction of literals: How can we check whether a set of *literals* is satisfiable?

  In particular, in $\mathcal{T}_E$, $\mathcal{T}_A$, and $\mathcal{T}_Q$.

Related problems

- Deciding a conjunction of literals: How can we check whether a set of *literals* is satisfiable?

  In particular, in $\mathcal{T}_E$, $\mathcal{T}_A$, and $\mathcal{T}_Q$.

- How can we put theory reasoning and SAT solving together?

Related problems

- Deciding a conjunction of literals: How can we check whether a set of *literals* is satisfiable?

  In particular, in $\mathcal{T}_E$, $\mathcal{T}_A$, and $\mathcal{T}_Q$.

- How can we put theory reasoning and SAT solving together?

- Combination of theories: Given decision procedures for theories, how can we build a decision procedure for formulas using several theories?