# EC4219: Software Engineering

## Lecture 6 — First-Order Theories

Sunbeom So

2024 Spring

# Review: Syntax of First-Order Logic

FOL is an extension of PL with quantifiers and nonlogical symbols (constant-, function-, and predicate symbols). A FOL formula $F$ is defined by the grammar:

$$
\begin{array}{rcll}
F & \to & \bot \mid \top \mid p(t_1, \cdots, t_n) & \text{atom} \\
 & \mid & \neg F & \text{negation ("not")} \\
 & \mid & F_1 \land F_2 & \text{conjunction ("and")} \\
 & \mid & F_1 \lor F_2 & \text{disjunction ("or")} \\
 & \mid & F_1 \to F_2 & \text{implication ("implies")} \\
 & \mid & F_1 \leftrightarrow F_2 & \text{iff ("if and only if")} \\
 & \mid & \exists x. F[x] & \text{existential quantification} \\
 & \mid & \forall x. F[x] & \text{universal quantification}
\end{array}
$$

where a term $t$ is defined by the grammar:

$$t \to x \mid c \mid f(t_1, \cdots, t_n)$$

# Review: Semantics of First-Order Logic

The semantics is determined by an interpretation $I : (D_I, \alpha_I)$.

- A **domain** $D_I$ is a nonempty set of values. An **assignment** $\alpha_I$ is a mapping for free variables and non-logical symbols.

$\boxed{\text{Base cases}}$

$$I \models \top$$
$$I \not\models \bot$$
$$I \models p(t_1, \cdots, t_n) \quad \text{iff} \quad \alpha_I[p(t_1, \cdots, t_n)] = true$$

$\boxed{\text{Inductive cases}}$

$$
\begin{array}{lll}
I \models \neg F & \text{iff} & I \not\models F \\
I \models F_1 \wedge F_2 & \text{iff} & I \models F_1 \text{ and } I \models F_2 \\
I \models F_1 \vee F_2 & \text{iff} & I \models F_1 \text{ or } I \models F_2 \\
I \models F_1 \rightarrow F_2 & \text{iff} & I \not\models F_1 \text{ or } I \models F_2 \\
I \models F_1 \leftrightarrow F_2 & \text{iff} & (I \models F_1 \text{ and } I \models F_2) \text{ or } (I \not\models F_1 \text{ and } I \not\models F_2) \\
I \models \forall x.F & \text{iff} & \text{for all } v \in D_I, I \triangleleft \{x \mapsto v\} \models F \\
I \models \exists x.F & \text{iff} & \text{there exists } v \in D_I \text{ such that } I \triangleleft \{x \mapsto v\} \models F
\end{array}
$$

# Review: Satisfiability of FOL Formulas

Q. Is the following formula $\boldsymbol{true}$ or $\boldsymbol{false}$?

$$\exists x. x + \mathbf{0} = \mathbf{1}$$

# Review: Satisfiability of FOL Formulas

Q. Is the following formula $\boldsymbol{true}$ or $\boldsymbol{false}$?

$$\exists x. x + 0 = 1$$

- $\boldsymbol{true}$ under the conventional interpretation $I : \{\mathbb{Z}, \alpha_I\}$ where

$$\alpha_I : \{+ \mapsto +_{\mathbb{Z}}, 0 \mapsto 0_{\mathbb{Z}}, 1 \mapsto 1_{\mathbb{Z}}, = \mapsto =_{\mathbb{Z}}\}$$

- $\boldsymbol{false}$ under the following interpretation $I : \{\mathbb{Z}, \alpha_I\}$ where

$$\alpha_I : \{+ \mapsto *_{\mathbb{Z}}, 0 \mapsto 0_{\mathbb{Z}}, 1 \mapsto 1_{\mathbb{Z}}, = \mapsto =_{\mathbb{Z}}\}$$

In FOL formulas, non-logical symbols are **uninterpreted!**
(i.e., can be assigned any meaning)

# Necessaity of First-Order Theories

- In practice, we are interested in a specific class of interpretations. That is, we have **fixed meanings** for some non-logical symbols!
  - Given $F : \exists x. x + 0 = 1$, we expect $+$ is treated as $+_{\mathbb{Z}}$.
- First-order logic is rather a general framework for building specific logic, called **First-order theories**, by imposing some restrictions (i.e., giving fixed meaning to non-logical symbols).
  - In the theory of integers ($T_{\mathbb{Z}}$), $+$ in $F$ is always treated as $+_{\mathbb{Z}}$.
- Q.How to restrict interpretations?
  A. By providing a set of axioms. That is, we consider interpretations that satisfy the axioms only.

# First-Order Theories

A first-order theory $T$ is defined by the two components.

- **Signature:** A set of nonlogical symbols (constant-, function-, predicate symbols). Given a signature $\Sigma$, a $\Sigma$-formula is the formula constructed from non-logical symbols of $\Sigma$.
- **Axioms:** A set of closed FOL formulas whose nonlogical symbols are from $\Sigma$.

Signature restricts the syntax, and axioms restrict the interpretations.

# Basic Terminologies

- An interpretation $I$, which satisfies all axioms $\mathcal{A}$ of $T$, is called a $T$-**interpretation**.

$$I \models A \ \text{ for every } A \in \mathcal{A}$$

- A $\Sigma$-formula $F$ is **satisfiable in $T$** or $T$-**satisfiable**, if there is a $T$-interpretation that satisfies $F$.

$$I \models F \ \text{ for some } T\text{-interpretation } I$$

- A $\Sigma$-formula $F$ is **valid in $T$** or $T$-**valid**, if every $T$-interpretation satisfies $F$.

$$I \models F \ \text{ for every } T\text{-interpretation } I \text{ (can be written as } T \models F\text{)}$$

- A theory $T$ is **complete**, if for every closed $\Sigma$-formula $F$, $T \models F$ or $T \models \neg F$.
- A theory $T$ is **decidable** if $T \models F$ (checking $T$-validity) is decidable for every $\Sigma$-formula $F$.
  - There is an algorithm that always terminates with "yes" if $F$ is $T$-valid or with "no" if $F$ is $T$-invalid.

# Terminologies (cont'd)

A theory restricts only the nonlogical symbols. Restrictions on the logical symbols or the grammar are done by defining **fragments** of the logic. Two popular fragments:

- **Quantifier-free fragment:** the set of $\Sigma$-formulas without quantifiers.
- **Conjunctive fragment:** the set of formulas where the only boolean connective that is allowed is conjunction.

Many first-order theories are undecidable while their quantifier-free fragments are decidable. In practice, we are mostly interested in the satisfiability problem of the quantifier-free fragment of first-order theories.

# Plan

In the remainder of this lecture, we will explore commonly-used first-order theories.

- The theory of equality $T_E$
- Peano Arithmetic $T_{PA}$
- Presburger Arithmetic $T_{\mathbb{N}}$
- The theory of Reals $T_{\mathbb{R}}$ and Rationals $T_{\mathbb{Q}}$.
- The theory of Arrays $T_A$

# Theory of Equality

The theory of equality $T_E$ is the simplest and most widely-used first-order theory. Its signature

$$\Sigma_E : \{=, a, b, c, \cdots, f, g, h, \cdots, p, q, r, \cdots\}$$

consists of

- $=$ (equality), a binary predicate, and
- all constants, function- and predicate symbols.

Equality $=$ is an **interpreted** predicate symbol; its meaning will be defined via the axioms. The others are **uninterpreted** since functions, predicates, and constants are left unspecified.

# Axioms of the Theory of Equality

1. Reflexivity: $\forall x.x = x$
2. Symmetry: $\forall x, y.x = y \implies y = x$
3. Transivity: $\forall x, y, z.x = y \wedge y = z \implies x = z$
4. Function congruence (consistency): for each positive integer $n$ and $n$-ary function symbol $f$,

   $$\forall \vec{x}, \vec{y}.(\bigwedge_{i=1}^{n} x_i = y_i) \rightarrow f(\vec{x}) = f(\vec{y}).$$

   where $\vec{x} = x_1 \cdots x_n$ and $\vec{y} = y_1 \cdots y_n$.
5. Predicate congruence (consistency): for each positive integer $n$ and $n$-ary predicate symbol $p$,

   $$\forall \vec{x}, \vec{y}.(\bigwedge_{i=1}^{n} x_i = y_i) \rightarrow p(\vec{x}) = p(\vec{y}).$$

cf) 4 and 5 are axiom schemata; $f$ and $p$ should be instantiated to concrete function- and predicate symbols.

# Example: Theory of Equality

To prove that

$$F : a = b \wedge b = c \to g(f(a), b) = g(f(c), a)$$

is $T_E$-valid, assume otherwise to derive a contradiction.

| | | |
|---|---|---|
| **1.** | $I \not\models F$ | assumption |
| **2.** | $I \models a = b \wedge b = c$ | $1, \to$ |
| **3.** | $I \not\models g(f(a), b) = g(f(c), a)$ | $1, \to$ |
| **4.** | $I \models a = b$ | $2, \wedge$ |
| **5.** | $I \models b = c$ | $3, \wedge$ |
| **6.** | $I \models a = c$ | $4, 5,$ transitivity |
| **7.** | $I \models f(a) = f(c)$ | $6,$ function congruence |
| **8.** | $I \models b = a$ | $4,$ symmetry |
| **9.** | $I \models g(f(a), b) = g(f(c), a)$ | $7, 8,$ function congruence |
| **10.** | $I \models \bot$ | $3, 9$ |

# Decidability

Like the full first-order logic, $T_E$-validity is undecidable. However, there exists an efficient decision procedure for its quantifier-free fragment. [1]

[1] see Chap.9 in "The Calculus of Computation: Decision Procedures with Applications to Verification"

# Uninterpreted Functions

- In $T_E$, function symbols are uninterpreted since the axioms do not assign meaning to them other than in the context of equality.
- The only thing we know about them is that they are functions.

# Use of Uninterpreted Functions

**A main application of uninterpreted functions is to abstract complex formulas** that are otherwise difficult to automatically reason about.

- Given a formula $F$, treating a function symbol $f$ as uninterpreted makes the formula weaker; we ignore the semantics of $f$ except for congruence with respect to equality.
- Let $\varphi^{UF}$ be the formula derived from $\varphi$ by replacing some interpreted functions with uninterpreted ones. Then,

$$\models \varphi^{UF} \implies \models \varphi.$$

  Note that the converse is not true!
- $\varphi^{UF}$ is an approximation of $\varphi$ such that if $\varphi^{UF}$ is valid so is $\varphi$. But $\varphi^{UF}$ may fail to be valid even though $\varphi$ is.

Uninterpreted functions simplify proofs. Uninterpreted functions enable to reason about systems while ignoring the semantics of irrelevant parts.

# Example: Use of Uninterpreted Functions

Consider the task of proving the two C functions behave the same.

```
int power3 (int in) {
  int i, out;
  out = in;
  for (i=0; i<2; i++)
    out = out * in;
  return out;
}
```

```
int mypower3 (int in) {
  int out;
  out = (in * in) * in;
  return out;
}
```

We can prove the equivalence by translating the programs into formulas

$$\varphi_a : out_0 = in \land out_1 = out_0 * in \land out_2 = out_1 * in$$
$$\varphi_b : out = (in * in) * in$$

and checking the validity of the following formula:

$$\varphi_a \land \varphi_b \to out_2 = out$$

# Example: Use of Uninterpreted Functions (cont'd)

Deciding formulas with multiplication is generally hard. Replacing the multiplication symbol with uninterpreted functions can aid the problem.

$$\varphi_a^{UF} : out_0 = in \land out_1 = G(out_0, in) \land out_2 = G(out_1, in)$$
$$\varphi_b^{UF} : out = G(G(in, in), in)$$

The following abstract formula is valid and so is the original formula.

$$\varphi_a^{UF} \land \varphi_b^{UF} \rightarrow out_2 = out$$

# Theory of Peano Arithmetic

A theory for natural numbers. The theory of Peano arithmetic $\boldsymbol{T_{PA}}$ has the signature

$$\boldsymbol{\Sigma_{PA} : \{0, 1, +, \cdot, =\}}$$

where

- $\mathbf{0}$ and $\mathbf{1}$ are constants,
- $+$ (addition) and $\cdot$ (multiplication) are binary functions, and
- $=$ (equality) is a binary predicate.

# Axioms: Theory of Peano Arithmetic

The axioms of $T_{PA}$:

1. Zero: $\forall x.\ \neg(x + 1 = 0)$
2. Successor: $\forall x, y.\ x + 1 = y + 1 \rightarrow x = y$
3. Plus zero: $\forall x.\ x + 0 = x$
4. Plus successor: $\forall x, y.\ x + (y + 1) = (x + y) + 1$
5. Times zero: $\forall x.\ x \cdot 0 = 0$
6. Times successor: $\forall x, y.\ x \cdot (y + 1) = x \cdot y + x$
7. Induction (axiom schema):
   $F[0] \wedge (\forall x. F[x] \rightarrow F[x + 1]) \rightarrow \forall x. F[x]$ (for every $\Sigma_{PA}$-formula $F$ with exactly one free variable)

# Example: $T_{PA}$ formulas

- The formula $3x + 5 = 2y$ can be written as

$$(1 + 1 + 1) \cdot x + 1 + 1 + 1 + 1 + 1 = (1 + 1) \cdot y$$

- The inequality $3x + 5 > 2y$ can be expressed by

$$\exists z. z \neq 0 \land 3x + 5 = 2y + z$$

  where $\equiv \neg(z = 0) z \neq 0$.

- Every formula of the set

$$\{\forall x, y, z. x \neq 0 \land y \neq 0 \land z \neq 0 \rightarrow x^n + y^n \neq z^n \mid n > 2 \land n \in \mathbb{Z}\}$$

  is $T_{PA}$-valid (Fermat's Last Theorem).

# Decidability and Completeness of $T_{PA}$

- $T_{PA}$ is neither complete nor decidable.
- Even undecidable is its quantifier-free fragment.
- A fragment of $T_{PA}$, called Presburger arithmetic, is both complete and decidable.

# Axioms: Theory of Presburger Arithmetic

A restriction that does not allow multiplication. The theory has a signature

$$\Sigma_{\mathbb{N}} : \{0, 1, +, =\}$$

and axioms:

1. Zero: $\forall x. \neg(x + 1 = 0)$
2. Successor: $\forall x, y. x + 1 = y + 1 \rightarrow x = y$
3. Plus zero: $\forall x. x + 0 = x$
4. Plus successor: $\forall x, y. x + (y + 1) = (x + y) + 1$
5. Induction: $F[0] \wedge (\forall x. F[x] \rightarrow F[x + 1]) \rightarrow \forall x. F[x]$

# Theory of Integers

- Although integer reasoning can be done with natural numbers, it is convenient to have a theory of integers.
- The theory of integers $\boldsymbol{T_{\mathbb{Z}}}$ (with linear arithmetic) has signatures

$$\boldsymbol{\Sigma_{\mathbb{Z}} : \{\cdots, -1, 0, 1, \cdots, -3\cdot, -2\cdot, 2\cdot, 3\cdot, \cdots, +, -, =, >\}}$$

- $\boldsymbol{T_{\mathbb{Z}}}$ is no more expressive but more convenient than Presburger arithmetic ($\boldsymbol{T_{\mathbb{N}}}$).
- $\boldsymbol{T_{\mathbb{Z}}}$ is both complete and decidable, and one of the most widely used theories.

# cf) Integer Reasoning with Natural Number

- Integer reasoning can be performed with natural number reasoning: formulas over all integers $\mathbb{Z} = \{\cdots, -1, 0, 1, \cdots\}$ can be encoded as $\Sigma_{\mathbb{N}}$-formulas.
- Idea: replace integer variables with the difference of variables of natural-numbers. For example, consider the formula

$$F_0 : \forall w, x. \exists y, z. x + 2y - z > -3w$$

1. Introduce two variables, $v_p$ and $v_n$, for each variable $v$ of $F_0$:

$$F_1 : \forall w_p, w_n, x_p, x_n. \exists y_p, y_n, z_p, z_n. \\ (x_p - x_n) + 2(y_p - y_n) - (z_p - z_n) > -3(w_p - w_n)$$

2. Move negated terms to the other side of the inequality.

$$F_2 : \forall w_p, w_n, x_p, x_n. \exists y_p, y_n, z_p, z_n. \\ x_p + 2y_p + z_n + 3w_p > x_n + 2y_n + z_p + 3w_n$$

$F_2$ is $T_{\mathbb{N}}$-valid precisely when $F_0$ is valid in the integer interpretation.

# Theories of Reals and Rationals

The theory of reals $\boldsymbol{T}_{\mathbb{R}}$ has the signature

$$\boldsymbol{\Sigma}_{\mathbb{R}} : \{0, 1, +, -, \cdot, =, \geq\}.$$

The theory of rationals $\boldsymbol{T}_{\mathbb{Q}}$ has the signature

$$\boldsymbol{\Sigma}_{\mathbb{Q}} : \{0, 1, +, -, =, \geq\}.$$

$\boldsymbol{T}_{\mathbb{R}}$ and $\boldsymbol{T}_{\mathbb{Q}}$ have complex axioms (see Chapter 3 of the textbook).

# Theory of Arrays

The theory of arrays $T_A$ has the signature

$$\Sigma_A : \{\cdot[\cdot], \cdot\langle \cdot \triangleleft \cdot \rangle, =\}$$

where

- $a[i]$ represents the value of array $a$ at position $i$ (binary function).
- $a\langle i \triangleleft v \rangle$ represents the modified array $a$ in which position $i$ has the value $v$ (ternary function).
- $=$ is the equality predicate.

The axioms of $T_A$:

1. the axioms of reflexivity, symmetry, and transitivity of $T_E$
2. (array congruence) $\forall a, i, j. i = j \rightarrow a[i] = a[j]$
3. (read-over-write 1) $\forall a, i, j. i = j \rightarrow a\langle i \triangleleft v \rangle[j] = v$
4. (read-over-write 2) $\forall a, i, j. i \neq j \rightarrow a\langle i \triangleleft v \rangle[j] = a[j]$

# Example: Theory of Arrays

Determine the validity of the formula:

$$F : a[i] = e \rightarrow \forall j. a\langle i \triangleleft e\rangle[j] = a[j]$$

| | | |
|---|---|---|
| **1.** | $I \not\models F$ | assumption |
| **2.** | $I \models a[i] = e$ | $1, \rightarrow$ |
| **3.** | $I \not\models \forall j. a\langle i \triangleleft e\rangle[j] = a[j]$ | $1, \rightarrow$ |
| **4.** | $I_1 : I \triangleleft \{j \mapsto v\} \not\models a\langle i \triangleleft e\rangle[j] = a[j]$ | $3, \forall,$ for some $v \in D$ |
| **5.** | $I_1 \models a\langle i \triangleleft e\rangle[j] \neq a[j]$ | $4, \neg$ |
| **6.** | $I_1 \models i = j$ | $5,$ read-over-write 2 |
| **7.** | $I_1 \models a[i] = a[j]$ | $6,$ array congruence |
| **8.** | $I_1 \models a\langle i \triangleleft e\rangle[j] = e$ | $6,$ read-over-write 1 |
| **9.** | $I_1 \models a\langle i \triangleleft e\rangle[j] = a[j]$ | $2, 7, 8,$ transitivity |
| **10.** | $I_1 \models \bot$ | |

# Decidability of First-order Theories

| Theory | Description | Full | QFF |
|---|---|---|---|
| $T_E$ | equality | no | yes |
| $T_{PA}$ | Peano arithmetic | no | no |
| $T_{\mathbb{N}}$ | Presburger arithmetic | yes | yes |
| $T_{\mathbb{Z}}$ | linear integers | yes | yes |
| $T_{\mathbb{R}}$ | reals (with $\cdot$) | yes | yes |
| $T_{\mathbb{Q}}$ | rationals (without $\cdot$) | yes | yes |
| $T_{RDS}$ | recursive data structures | no | yes |
| $T_{RDS}$ | arrays | no | yes |
| $T_A^=$ | arrays with extentionality | no | yes |

# Combining Theories

- In practice, the formulas we check for satisfiability or validity span multiple theories.
  - For example, in program verification, we want to prove properties about a list of integers or an array of integers.
- Nelson and Oppen presented a general method for combining quantifier-free fragments of first-order theories.
- Suppose we are given $T_1$ and $T_2$ such that $\Sigma_1 \cap \Sigma_2 = \{=\}$, the combined theory $T_1 \cup T_2$ has the signature $\Sigma_1 \cup \Sigma_2$ and axioms $A_1 \cup A_2$. Nelson and Oppen showed that if
  - satisfiability in the quantifier-free fragments of $T_1$ is decidable
  - satisfiability in the quantifier-free fragments of $T_2$ is decidable
  - and certain conditions are met

  then satisfiability in the quantifier-free fragment of $T_1 \cup T_2$ is decidable.
- Furthermore, if the decision procedures for $T_1$ and $T_2$ are in P (resp., NP), then the combined decision procedure for $T_1 \cup T_2$ is in P (resp., NP).

# Summary

- FOL is an extension of PL with quantifiers and nonlogical symbols (constant-, function-, and predicate symbols).
- In FOL formulas, non-logical symbols are uninterpreted!
  - $\exists x. x + 0 = 1$ can be either $true$ or $false$.
- In practice, we are interested in a specific class of interpretations.
- The specific logic, called **First-order theories**, is built by imposing some restrictions.