Lecture 8

First-order logic

Syntax and semantics

Introduction to Logic for Computer Science

Prof Hongseok Yang
KAIST

These slides are minor variants of those made by Prof Worrell and Prof Haase for their logic course at Oxford.

Limitations of propositional logic

- Can only reason about true or false.
- Atomic formulas have no internal structure.
- Impossible to express “real” mathematical statements.

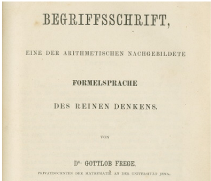
Limitations of propositional logic

- Can only reason about true or false.
- Atomic formulas have no internal structure.
- Impossible to express “real” mathematical statements.

Example

Every natural number x is either odd or even.

Frege’s Begriffsschrift (Concept script in English)


BEGRIFFSSCHRIFT,

EINE DER ARITHMETISCHEN NACHGEBILDETE

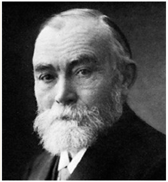
FORMELSPRACHE

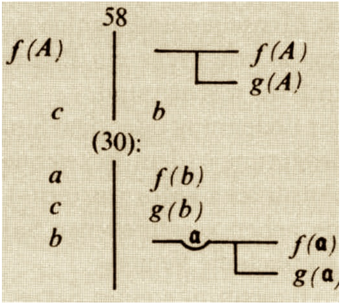
DES REINEN DENKENS.

VON

DR. GOTTLOB FREGE.

PRIVATDOCENTEN DER MATHEMATIK AN DER UNIVERSITÄT JENA.





Introduced a classical second-order logic with equality.

Fermat’s last theorem in first-order logic

$$\forall n \left(n > 2 \to \neg \left(\exists x \, \exists y \, \exists z \, (x^n + y^n = z^n) \right) \right).$$

Fermat's last theorem in first-order logic

$$\forall n \left(n > 2 \to \neg \left(\exists x \, \exists y \, \exists z \, (x^n + y^n = z^n) \right) \right).$$

Uses quantification.

Fermat's last theorem in first-order logic

$$\forall n \left(n > 2 \rightarrow \neg \left(\exists x \, \exists y \, \exists z \, (x^n + y^n = z^n) \right) \right).$$

Uses quantification.

Includes atomic formulas with internal structures.

Fermat's last theorem in first-order logic

$$\forall n \left(n > 2 \to \neg \left(\exists x \, \exists y \, \exists z \, (x^n + y^n = z^n) \right) \right).$$

Uses quantification.

Includes atomic formulas with internal structures.

What atomic formulas can we use?

Signatures – Overview

Parameterises the syntax of first-order logic.

Determines the atomic formulas that we can write.

Signatures – Overview

Parameterises the syntax of first-order logic.

Determines the atomic formulas that we can write.

Example: $\sigma_N = \langle 0, 1, 2, +, \times, \mathit{pow}, <, = \rangle$.

Signatures – Overview

Parameterises the syntax of first-order logic.

Determines the atomic formulas that we can write.

Example: $\sigma_N = \langle 0, 1, 2, +, \times, \mathit{pow}, <, = \rangle$.

Signatures – Overview

Parameterises the syntax of first-order logic.

Determines the atomic formulas that we can write.

Example: $\sigma_N = \langle 0, 1, 2, +, \times, pow, <, = \rangle$.

Signatures – Overview

Parameterises the syntax of first-order logic.

Determines the atomic formulas that we can write.

Example: $\sigma_N = \langle 0, 1, 2, +, \times, pow, <, = \rangle$.

Signatures

Definition

A **signature** σ is a tuple consisting of

- a set of **constant symbols** (denoted c, d),
- a set of **function symbols** (denoted f, g), and
- a set of **predicate symbols** (denoted P, Q, R).

Each function and predicate symbol has an **arity** $k > 0$.

Signatures

Definition

A **signature** σ is a tuple consisting of

- a set of **constant symbols** (denoted c, d),
- a set of **function symbols** (denoted f, g), and
- a set of **predicate symbols** (denoted P, Q, R).

Each function and predicate symbol has an **arity** $k > 0$.

Independent of signature, we assume that a countably infinite number of **variables** $\mathcal{X} = \{x_1, x_2, \ldots\}$ is given.

Signatures

Definition

A **signature** σ is a tuple consisting of

- a set of **constant symbols** (denoted c, d),
- a set of **function symbols** (denoted f, g), and
- a set of **predicate symbols** (denoted P, Q, R).

Each function and predicate symbol has an **arity** $k > 0$.

Independent of signature, we assume that a countably infinite number of **variables** $\mathcal{X} = \{x_1, x_2, \ldots\}$ is given.

A signature of number theory:

$$\sigma_N = \langle 0, 1, 2, +, \times, pow, <, = \rangle.$$

where $0, 1, 2$ are constant symbols, $+, \times, pow$ are function symbols of arity two, and $<, =$ are predicate symbols of arity two.

Signatures

Definition

A **signature** σ is a tuple consisting of

- a set of **constant symbols** (denoted c, d),
- a set of **function symbols** (denoted f, g), and
- a set of **predicate symbols** (denoted P, Q, R).

Each function and predicate symbol has an **arity** $k > 0$.

Independent of signature, we assume that a countably infinite number of **variables** $\mathcal{X} = \{x_1, x_2, \ldots\}$ is given.

A signature of number theory:

$$\sigma_N = \langle 0, 1, 2, +, \times, pow, <, = \rangle.$$

where $0, 1, 2$ are constant symbols, $+, \times, pow$ are function symbols of arity two, and $<, =$ are predicate symbols of arity two.

Ex: Define a signature σ_P for a propositional formulas over variables p, q, r. Include a predicate symbol for logical entailment.

σ-Terms

Definition

Given a signature σ, σ-**terms** or **terms** are defined by structural induction:

- Each variable x is a term.
- Each constant symbol c is a term.
- If $t_1, \ldots, t_k$ are terms and f is a k-ary function symbol, then $f(t_1, \ldots, t_k)$ is a term.

σ-Terms

Definition

Given a signature σ, σ-**terms** or **terms** are defined by structural induction:

- Each variable x is a term.
- Each constant symbol c is a term.
- If $t_1, \ldots, t_k$ are terms and f is a k-ary function symbol, then $f(t_1, \ldots, t_k)$ is a term.

Expressions denoting objects, such as natural numbers.

Example: Given the signature σ_N of number theory,

$$\times(+(1, 1), x)$$

is a term.

σ-Terms

Definition

Given a signature σ, σ-**terms** or **terms** are defined by structural induction:

- Each variable x is a term.
- Each constant symbol c is a term.
- If $t_1, \ldots, t_k$ are terms and f is a k-ary function symbol, then $f(t_1, \ldots, t_k)$ is a term.

Expressions denoting objects, such as natural numbers.

Example: Given the signature σ_N of number theory,

$$\times(+(1, 1), x)$$

is a term. We often use **infix** notation and write $(1 + 1) \times x$ instead.

Formulas

Definition

The set of **formulas** given a signature σ is defined inductively:

- $P(t_1, \ldots, t_k)$ is a formula for any k-ary predicate symbol P in σ and any σ-terms $t_1, \ldots, t_k$.
- *true* and *false* are formulas.
- For each formula F, $\neg F$ is a formula.
- For formulas F, G, $(F \vee G)$ and $(F \wedge G)$ are both formulas.
- If F is a formula and x is a variable, then $\exists x\, F$ and $\forall x\, F$ are both formulas (**existential** and **universal quantifiers**).

Formulas

Definition

The set of **formulas** given a signature σ is defined inductively:

- $P(t_1, \ldots, t_k)$ is a formula for any k-ary predicate symbol P in σ and any σ-terms $t_1, \ldots, t_k$.
- *true* and *false* are formulas.
- For each formula F, $\neg F$ is a formula.
- For formulas F, G, $(F \vee G)$ and $(F \wedge G)$ are both formulas.
- If F is a formula and x is a variable, then $\exists x\, F$ and $\forall x\, F$ are both formulas (**existential** and **universal quantifiers**).

Formulas from the first two cases are called **atomic formulas**.

Formulas

Definition

The set of **formulas** given a signature σ is defined inductively:

- $P(t_1, \ldots, t_k)$ is a formula for any k-ary predicate symbol P in σ and any σ-terms $t_1, \ldots, t_k$.
- *true* and *false* are formulas.
- For each formula F, $\neg F$ is a formula.
- For formulas F, G, $(F \vee G)$ and $(F \wedge G)$ are both formulas.
- If F is a formula and x is a variable, then $\exists x\, F$ and $\forall x\, F$ are both formulas (**existential** and **universal quantifiers**).

Formulas from the first two cases are called **atomic formulas**.

Describe properties of objects.

Formulas

Definition

The set of **formulas** given a signature σ is defined inductively:

- $P(t_1, \ldots, t_k)$ is a formula for any k-ary predicate symbol P in σ and any σ-terms $t_1, \ldots, t_k$.
- *true* and *false* are formulas.
- For each formula F, $\neg F$ is a formula.
- For formulas F, G, $(F \vee G)$ and $(F \wedge G)$ are both formulas.
- If F is a formula and x is a variable, then $\exists x\, F$ and $\forall x\, F$ are both formulas (**existential** and **universal quantifiers**).

Formulas from the first two cases are called **atomic formulas**.

Describe properties of objects.

Ex1: Assume σ_N of number theory. Express that there are infinitely many prime numbers.

Ex2: Assume σ_P for propositional logic. Express de Morgan's laws.

Quantifier depth and free/bound variables

Inductive structure of formulas enables structural induction:

Definition

Quantifier depth is defined as follows:

$$\begin{aligned}
\mathrm{qd}(P(t_1,\ldots,t_k)) = \mathrm{qd}(\mathit{true}) = \mathrm{qd}(\mathit{false}) &:= 0\\
\mathrm{qd}(\neg F) &:= \mathrm{qd}(F)\\
\mathrm{qd}(F \wedge G) = \mathrm{qd}(F \vee G) &:= \max(\mathrm{qd}(F), \mathrm{qd}(G))\\
\mathrm{qd}(\exists x\, F) = \mathrm{qd}(\forall x\, F) &:= \mathrm{qd}(F) + 1.
\end{aligned}$$

Quantifier depth and free/bound variables

Inductive structure of formulas enables structural induction:

Definition

Quantifier depth is defined as follows:

$$\begin{aligned}
\mathrm{qd}(P(t_1,\ldots,t_k)) = \mathrm{qd}(\mathit{true}) = \mathrm{qd}(\mathit{false}) &:= 0 \\
\mathrm{qd}(\neg F) &:= \mathrm{qd}(F) \\
\mathrm{qd}(F \wedge G) = \mathrm{qd}(F \vee G) &:= \max(\mathrm{qd}(F), \mathrm{qd}(G)) \\
\mathrm{qd}(\exists x\, F) = \mathrm{qd}(\forall x\, F) &:= \mathrm{qd}(F) + 1.
\end{aligned}$$

Ex1: Define $\mathrm{free}(F)$ that computes the set of free variables in F.

Ex2: Define $\mathrm{bound}(F)$ that computes the set of bound variables in F.

Scope, free/bound variables, and sentences

For a formula $\exists x\, F$, if S is a subformula of F, we say S is in the **scope** of the quantifier $\exists x$. Likewise for $\forall x\, F$ and a term t appearing in F.

Scope, free/bound variables, and sentences

For a formula $\exists x\, F$, if S is a subformula of F, we say S is in the **scope** of the quantifier $\exists x$. Likewise for $\forall x\, F$ and a term t appearing in F.

An occurrence of a variable x in F is **bound** if it is in the scope of $\exists x$ or $\forall x$. Otherwise, the occurrence is **free**.

Ex1: Can an occurrence of x be both free and bound in F?

Scope, free/bound variables, and sentences

For a formula $\exists x\, F$, if S is a subformula of F, we say S is in the **scope** of the quantifier $\exists x$. Likewise for $\forall x\, F$ and a term t appearing in F.

An occurrence of a variable x in F is **bound** if it is in the scope of $\exists x$ or $\forall x$. Otherwise, the occurrence is **free**.

Ex1: Can an occurrence of x be both free and bound in F?

A variable x in F is **bound** if F contains a quantifier over x.

A variable x in F is **free** if it has a free occurrence in F.

Formulas with no free variables are called **closed formulas** or **sentences**.

Ex2: Can a variable x be both free and bound in F?

Semantics of first-order logic

Definition

Given a signature σ, a **σ-structure** (or **assignment**) $\mathcal{A}$ consists of:

- a non-empty set $U_\mathcal{A}$ called the **universe** of the structure;
- for each constant symbol c, an element $c_\mathcal{A}$ of $U_\mathcal{A}$;
- for each k-ary function symbol f in σ, a k-ary function,

$$f_\mathcal{A} \colon \underbrace{U_\mathcal{A} \times \cdots \times U_\mathcal{A}}_{k} \to U_\mathcal{A};$$

- for each k-ary predicate symbol P in σ, a k-ary relation

$$P_\mathcal{A} \subseteq \underbrace{U_\mathcal{A} \times \cdots \times U_\mathcal{A}}_{k};$$

- for each variable x, an element $x_\mathcal{A}$ of $U_\mathcal{A}$.

Often a σ-structure or structure just means the first four.

Semantics of first-order logic

Definition

Given a signature σ, a σ-**structure** (or **assignment**) $\mathcal{A}$ consists of:

- a non-empty set $U_\mathcal{A}$ called the **universe** of the structure;
- for each constant symbol c, an element $c_\mathcal{A}$ of $U_\mathcal{A}$;
- for each k-ary function symbol f in σ, a k-ary function,

$$f_\mathcal{A} \colon \underbrace{U_\mathcal{A} \times \cdots \times U_\mathcal{A}}_{k} \to U_\mathcal{A};$$

- for each k-ary predicate symbol P in σ, a k-ary relation

$$P_\mathcal{A} \subseteq \underbrace{U_\mathcal{A} \times \cdots \times U_\mathcal{A}}_{k};$$

- for each variable x, an element $x_\mathcal{A}$ of $U_\mathcal{A}$.

Often a σ-structure or structure just means the first four.

Example

Let σ_N be the signature of number theory. The natural σ_N-structure $\mathcal{A}$ is:

- $U_{\mathcal{A}} := \mathbb{N} = \{0, 1, \ldots\}$.
- $0_{\mathcal{A}} := 0$, $1_{\mathcal{A}} := 1$, $2_{\mathcal{A}} := 2$.
- $+_{\mathcal{A}} := (m, n) \mapsto m + n$.
- $\times_{\mathcal{A}} := (m, n) \mapsto m \cdot n$.
- $pow_{\mathcal{A}} := (m, n) \mapsto m^n$.
- $(<_{\mathcal{A}}) := \{(n, m) : n, m \in \mathbb{N}, n < m\}$ and $(=_{\mathcal{A}}) := \{(n, n) : n \in \mathbb{N}\}$.

Example

Let σ_N be the signature of number theory. The natural σ_N-structure $\mathcal{A}$ is:

- $U_{\mathcal{A}} := \mathbb{N} = \{0, 1, \ldots\}$.
- $0_{\mathcal{A}} := 0$, $1_{\mathcal{A}} := 1$, $2_{\mathcal{A}} := 2$.
- $+_{\mathcal{A}} := (m, n) \mapsto m + n$.
- $\times_{\mathcal{A}} := (m, n) \mapsto m \cdot n$.
- $pow_{\mathcal{A}} := (m, n) \mapsto m^n$.
- $(<_{\mathcal{A}}) := \{(n, m) : n, m \in \mathbb{N}, n < m\}$ and $(=_{\mathcal{A}}) := \{(n, n) : n \in \mathbb{N}\}$.

The following $\mathcal{B}$ is also a σ_N-structure:

- $U_{\mathcal{A}} := \{A, B, 5\}$.
- $0_{\mathcal{A}} := A$, $1_{\mathcal{A}} := 5$, $2_{\mathcal{A}} := B$.
- $+_{\mathcal{A}} := (m, n) \mapsto 5$.
- $\times_{\mathcal{A}} = pow_{\mathcal{A}} := (m, n) \mapsto A$.
- $(<_{\mathcal{A}}) = (=_{\mathcal{A}}) := \{(A, B), (B, B)\}$.

Ex: Define a structure for the signature σ_P for propositional logic.

Semantics of first-order logic – Overview

Parameterised by a structure $\mathcal{A}$ of a signature σ.

First, semantics of terms t (using $\mathcal{A}(t) \in U_{\mathcal{A}}$).

Then, semantics of formulas F (using $\mathcal{A} \models F$).

Semantics of first-order logic – Terms

Definition

The **value** $\mathcal{A}(t) \in U_{\mathcal{A}}$ of term t is defined inductively as follows:

- For a constant symbol c, $\mathcal{A}(c) := c_{\mathcal{A}}$.
- For a variable x, $\mathcal{A}(x) := x_{\mathcal{A}}$.
- For a term $f(t_1, \ldots, t_k)$, where f is a k-ary function symbol and $t_1, \ldots, t_k$ are terms,

$$\mathcal{A}(f(t_1, \ldots, t_k)) := f_{\mathcal{A}}(\mathcal{A}(t_1), \ldots, \mathcal{A}(t_k)).$$

Semantics of first-order logic – Formulas

Definition

Define the **satisfaction relation** $\mathcal{A} \models F$ ($\mathcal{A}$ **satisfies** F, or $\mathcal{A}$ **models** F) by structural induction:

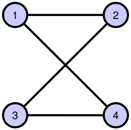
- $\mathcal{A} \models P(t_1, \ldots, t_k)$ if and only if $(\mathcal{A}(t_1), \ldots, \mathcal{A}(t_k)) \in P_{\mathcal{A}}$.
- $\mathcal{A} \models \mathit{true}$ always holds.
- $\mathcal{A} \models \mathit{false}$ never holds.
- $\mathcal{A} \models (F \wedge G)$ if and only if $\mathcal{A} \models F$ and $\mathcal{A} \models G$.
- $\mathcal{A} \models (F \vee G)$ if and only if $\mathcal{A} \models F$ or $\mathcal{A} \models G$.
- $\mathcal{A} \models \neg F$ if and only if $\mathcal{A} \not\models F$.
- $\mathcal{A} \models \exists x\, F$ if and only if there exists $a \in U_{\mathcal{A}}$ such that $\mathcal{A}_{[x \mapsto a]} \models F$.
- $\mathcal{A} \models \forall x\, F$ if and only if $\mathcal{A}_{[x \mapsto a]} \models F$ for all $a \in U_{\mathcal{A}}$.

Semantics of first-order logic – Example

Ex: Let $\mathcal{A}$ be the natural σ-structure of number theory. Then, does the following satisfaction relation hold?

$$\mathcal{A} \models \forall x \, \exists y((x = (1+1) \times y) \vee (x = 1 + (1+1) \times y)).$$

Semantics of first-order logic – Example



Undirected graph as a σ-structure with one binary relation symbol E interpreted as the edge relation.

The above graph represented by the structure $\mathcal{A}$ with universe $U_{\mathcal{A}} = \{1, 2, 3, 4\}$ and irreflexive symmetric binary relation

$$E_{\mathcal{A}} = \{(1,2),(2,3),(3,4),(4,1),(2,1),(3,2),(4,3),(1,4)\}\,.$$

Ex1: Does the following satisfaction relation hold?

$$\mathcal{A} \models \forall x\,\neg E(x,x) \wedge \forall x\,\forall y\,(E(x,y) \rightarrow E(y,x)).$$

Ex2: Does the following satisfaction relation hold?

$$\mathcal{A} \models \forall x\,\forall y\,\exists z_1\,\exists z_2\,(E(x,z_1) \wedge E(z_1,z_2) \wedge E(z_2,y)).$$

The relevance lemma

Lemma

Let $\mathcal{A}$ and $\mathcal{A}'$ be σ-structures, and F be a formula over σ. If

1. *$\mathcal{A}$ and $\mathcal{A}'$ use the same universe U;*
2. *they have identical interpretations of the constant, function, and predicate symbols in σ; and*
3. *they give the same interpretation to each variable occurring free in F,*

then $\mathcal{A} \models F$ if and only if $\mathcal{A}' \models F$.

The relevance lemma

Lemma

Let $\mathcal{A}$ and $\mathcal{A}'$ be σ-structures, and F be a formula over σ. If

1. *$\mathcal{A}$ and $\mathcal{A}'$ use the same universe U;*
2. *they have identical interpretations of the constant, function, and predicate symbols in σ; and*
3. *they give the same interpretation to each variable occurring free in F,*

then $\mathcal{A} \models F$ if and only if $\mathcal{A}' \models F$.

Ex: Prove the lemma. Use structural induction on terms and formulas.