Theoretical Computer Science II (ACS II)

3. First-order logic

Bernhard Nebel   Christian Becker-Asano

Albert-Ludwigs-Universität Freiburg

November 14th, 2011

Theoretical Computer Science II (ACS II)

November 14th, 2011 — 3. First-order logic

Motivation

Propositional logic does not allow talking about structured objects.

A famous syllogism

- All men are mortal.
- Socrates is a man.
- Therefore, Socrates is mortal.

It is impossible to formulate this in propositional logic.
$\rightsquigarrow$ first-order logic (predicate logic)

Elements of logic (recap)

The same questions as before:

- Which elements are well-formed? $\rightsquigarrow$ syntax
- What does it mean for a formula to be true? $\rightsquigarrow$ semantics
- When does one formula follow from another? $\rightsquigarrow$ inference

We will now discuss these questions for first-order logic
(but only touching the topic of inference briefly).

Building blocks of first-order logic

In propositional logic, we can only talk about formulae (propositions). An interpretation tells us which formulae are true (or false).

In first-order logic, there are two different kinds of elements under discussion:

- terms identify the object under discussion
  - "Socrates"
  - "the square root of 5"
- formulae state properties of the objects under discussion
  - "All men are mortal."
  - "The square root of 5 is greater than 2."

An interpretation tells us which object is denoted by a term, and which formulae are true (or false).

Syntax of first-order logic: signatures

Definition (signature)

A (first-order) signature is a 4-tuple $\mathcal{S} = \langle \mathcal{V}, \mathcal{C}, \mathcal{F}, \mathcal{R} \rangle$ consisting of the following four (disjoint) parts:

- a finite or countable set $\mathcal{V}$ of variable symbols,
- a finite or countable set $\mathcal{C}$ of constant symbols,
- a finite or countable set $\mathcal{F}$ of function symbols,
- a finite or countable set $\mathcal{R}$ of relation symbols (also called predicate symbols)

Each function symbol $\mathsf{f} \in \mathcal{F}$ and relation symbol $\mathsf{R} \in \mathcal{R}$ has an associated arity (number of arguments) $\mathit{arity}(\mathsf{f}), \mathit{arity}(\mathsf{R}) \in \mathbb{N}_1$.

Terminology: A k-ary (function or relation) symbol is a symbol s with $\mathit{arity}(\mathsf{s}) = k$.

Also: unary, binary, ternary

Signatures: examples

Example: arithmetic

- $\mathcal{V} = \{x, y, z, x_1, x_2, x_3, \ldots\}$
- $\mathcal{C} = \{\mathrm{zero}, \mathrm{one}\}$
- $\mathcal{F} = \{\mathrm{sum}, \mathrm{product}\}$
- $\mathcal{R} = \{\mathrm{Positive}, \mathrm{PerfectSquare}\}$

$\mathit{arity}(\mathrm{sum}) = \mathit{arity}(\mathrm{product}) = 2$, $\mathit{arity}(\mathrm{Positive}) = \mathit{arity}(\mathrm{PerfectSquare}) = 1$

Conventions:

- variable symbols are typeset in *italics*, other symbols in an upright typeface
- relation symbols begin with upper-case letters, other symbols with lower-case letters

Signatures: examples

Example: genealogy

- $\mathcal{V} = \{x, y, z, x_1, x_2, x_3, \ldots\}$
- $\mathcal{C} = \{\text{queen-elizabeth}, \text{donald-duck}\}$
- $\mathcal{F} = \emptyset$
- $\mathcal{R} = \{\text{Female}, \text{Male}, \text{Parent}\}$

$\mathit{arity}(\text{Female}) = \mathit{arity}(\text{Male}) = 1$, $\mathit{arity}(\text{Parent}) = 2$

Conventions:

- variable symbols are typeset in *italics*, other symbols in an upright typeface
- relation symbols begin with upper-case letters, other symbols with lower-case letters

Syntax of first-order logic: terms

Definition (term)

Let $\mathcal{S} = \langle \mathcal{V}, \mathcal{C}, \mathcal{F}, \mathcal{R} \rangle$ be a signature.
A term (over $\mathcal{S}$) is inductively constructed according to the following rules:

- Each variable symbol $v \in \mathcal{V}$ is a term.
- Each constant symbol $\mathsf{c} \in \mathcal{C}$ is a term.
- If $t_1, \dots, t_k$ are terms and $\mathsf{f} \in \mathcal{F}$ is a function symbol with arity k, then $\mathsf{f}(t_1, \dots, t_k)$ is a term.

Examples:

- x_4
- donald-duck
- $\mathsf{sum}(x_3, \mathsf{product}(\mathsf{one}, x_5))$

Syntax of first-order logic: formulae

Definition (formula)

Let $\mathcal{S} = \langle \mathcal{V}, \mathcal{C}, \mathcal{F}, \mathcal{R} \rangle$ be a signature.
A formula (over $\mathcal{S}$) is inductively constructed as follows:

- $\mathsf{R}(t_1, \ldots, t_k)$ (atomic formula; atom)
  where $\mathsf{R} \in \mathcal{R}$ is a k-ary relation symbol
  and $t_1, \ldots, t_k$ are terms (over $\mathcal{S}$)
- $t_1 = t_2$ (equality; also an atomic formula)
  where t_1 and t_2 are terms (over $\mathcal{S}$)
- $\forall x \, \varphi$ (universal quantification)
- $\exists x \, \varphi$ (existential quantification)
  where $x \in \mathcal{V}$ is a variable symbol and φ is a formula over $\mathcal{S}$
- ...

Syntax of first-order logic: formulae

Definition (formula)

- ...
- $\top$ (truth)
- $\bot$ (falseness)
- $\neg\varphi$ (negation)
  where φ is a formula over $\mathcal{S}$
- $(\varphi \wedge \psi)$ (conjunction)
- $(\varphi \vee \psi)$ (disjunction)
- $(\varphi \rightarrow \psi)$ (material conditional)
- $(\varphi \leftrightarrow \psi)$ (biconditional)
  where φ and ψ are formulae over $\mathcal{S}$

Syntax: examples

Example: arithmetic and genealogy

- $\mathrm{Positive}(x_2)$
- $\forall x\, \mathrm{PerfectSquare}(x) \rightarrow \mathrm{Positive}(x)$
- $\exists x_3\, \mathrm{PerfectSquare}(x_3) \wedge \neg \mathrm{Positive}(x_3)$
- $\forall x\, (x = y)$
- $\forall x\, (\mathrm{sum}(x, x) = \mathrm{product}(x, \mathrm{one}))$
- $\forall x \exists y\, (\mathrm{sum}(x, y) = \mathrm{zero})$
- $\forall x \exists y\, \mathrm{Parent}(y, x) \wedge \mathrm{Female}(y)$

Conventions: When we omit parentheses, $\forall$ and $\exists$ bind less tightly than anything else.

$\leadsto$ $\forall x\, P(x) \rightarrow Q(x)$ is read as $\forall x\, (P(x) \rightarrow Q(x))$, not as $(\forall x\, P(x)) \rightarrow Q(x)$.

Terminology and notation

- ground term: term that contains no variable symbol
  examples: zero, $\mathsf{sum}(\mathsf{one}, \mathsf{one})$, donald-duck
  counterexamples: x_4, $\mathsf{product}(x, \mathsf{zero})$
- similarly: ground atom, ground formula
  example: $\mathsf{PerfectSquare}(\mathsf{zero}) \vee \mathsf{one} = \mathsf{zero}$
  counterexample: $\exists x\, \mathsf{one} = x$

Abbreviation:

sequences of quantifiers of the same kind can be collapsed

- $\forall x \forall y \forall z\, \varphi \rightsquigarrow \forall xyz\, \varphi$
- $\forall x_3 \forall x_1 \exists x_2 \exists x_5\, \varphi \rightsquigarrow \forall x_3 x_1 \exists x_2 x_5\, \varphi$

Sometimes commas and/or colons are used:

- $\forall x, y, z \colon \varphi$
- $\forall x_3, x_1 \exists x_2, x_5\, \varphi$

Semantics of first-order logic: motivation

- In propositional logic, an interpretation was given by assigning to the atomic propositions.
- In first-order logic, there are no proposition variables; instead we need to interpret the meaning of constant, function and relation symbols.
- Variable symbols also need to be given meaning.
- However, this is not done through the interpretation itself, but through a separate variable assignment.

Interpretations and variable assignments

Let $\mathcal{S} = \langle \mathcal{V}, \mathcal{C}, \mathcal{F}, \mathcal{R} \rangle$ be a signature.

Definition (interpretation, variable assignment)

An interpretation (for $\mathcal{S}$) is a pair $\mathcal{I} = \langle D, \cdot^{\mathcal{I}} \rangle$ consisting of

- a nonempty set D called the domain (or universe) and
- a function $\cdot^{\mathcal{I}}$ that assigns a meaning to constant, function and relation symbols:
  - $\mathsf{c}^{\mathcal{I}} \in D$ for constant symbols $\mathsf{c} \in \mathcal{C}$
  - $\mathsf{f}^{\mathcal{I}} : D^k \to D$ for k-ary function symbols $\mathsf{f} \in \mathcal{F}$
  - $\mathsf{R}^{\mathcal{I}} \subseteq D^k$ for k-ary relation symbols $\mathsf{R} \in \mathcal{R}$

A variable assignment (for $\mathcal{S}$ and domain D)
is a function $\alpha : \mathcal{V} \to D$.

Idea: extend $\mathcal{I}$ and α to general terms, then to atoms,
then to arbitrary formulae

Semantics of first-order logic: informally

Example: $(\forall x \mathsf{Block}(x) \rightarrow \mathsf{Red}(x)) \land \mathsf{Block}(\mathsf{a})$
"For all objects x: if x is a block, then x is red.
Also, the object denoted by a is a block."

- Terms are interpreted as objects.
- Unary predicates denote properties of objects (being a block, being red, . . .)
- General predicates denote relations between objects (being the child of someone, having a common multiple, . . .)
- Universally quantified formulae ("$\forall$") are true if they hold for all objects in the domain.
- Existentially quantified formulae ("$\exists$") are true if they hold for at least one object in the domain.

Interpreting terms in first-order logic

Let $\mathcal{S} = \langle \mathcal{V}, \mathcal{C}, \mathcal{F}, \mathcal{R} \rangle$ be a signature.

Definition (interpretation of a term)

Let $\mathcal{I} = \langle D, \cdot^{\mathcal{I}} \rangle$ be an interpretation for $\mathcal{S}$,
and let α be a variable assignment for $\mathcal{S}$ and domain D.

Let t be a term over $\mathcal{S}$.
The interpretation of t under $\mathcal{I}$ and α, in symbols $t^{\mathcal{I},\alpha}$ is an element of the domain D defined as follows:

- If $t = x$ with $x \in \mathcal{V}$ (t is a variable term):
  $x^{\mathcal{I},\alpha} = \alpha(x)$
- If $t = \mathsf{c}$ with $\mathsf{c} \in \mathcal{C}$ (t is a constant term):
  $\mathsf{c}^{\mathcal{I},\alpha} = \mathsf{c}^{\mathcal{I}}$
- If $t = \mathsf{f}(t_1, \ldots, t_k)$ (t is a function term):
  $(\mathsf{f}(t_1, \ldots, t_k))^{\mathcal{I},\alpha} = \mathsf{f}^{\mathcal{I}}(t_1^{\mathcal{I},\alpha}, \ldots, t_k^{\mathcal{I},\alpha})$

Interpreting terms: example

Example

Signature: $\mathcal{S} = \langle \mathcal{V}, \mathcal{C}, \mathcal{F}, \mathcal{R} \rangle$
with $\mathcal{V} = \{x, y, z\}$, $\mathcal{C} = \{\text{zero}, \text{one}\}$ $\mathcal{F} = \{\text{sum}, \text{product}\}$,
$\mathit{arity}(\text{sum}) = \mathit{arity}(\text{product}) = 2$

$\mathcal{I} = \langle D, \cdot^{\mathcal{I}} \rangle$ with

- $D = \{d_0, d_1, d_2, d_3, d_4, d_5, d_6\}$
- $\text{zero}^{\mathcal{I}} = d_0$
- $\text{one}^{\mathcal{I}} = d_1$
- $\text{sum}^{\mathcal{I}}(d_i, d_j) = d_{(i+j) \bmod 7}$ for all $i, j \in \{0, \dots, 6\}$
- $\text{product}^{\mathcal{I}}(d_i, d_j) = d_{(i \cdot j) \bmod 7}$ for all $i, j \in \{0, \dots, 6\}$

$\alpha = \{x \mapsto d_5, y \mapsto d_5, z \mapsto d_0\}$

Interpreting terms: example (ctd.)

Example (ctd.)

- $\mathsf{zero}^{\mathcal{I},\alpha} =$
- $y^{\mathcal{I},\alpha} =$
- $\mathsf{sum}(x, y)^{\mathcal{I},\alpha} =$
- $\mathsf{product}(\mathsf{one}, \mathsf{sum}(x, \mathsf{zero}))^{\mathcal{I},\alpha} =$

Satisfaction/truth in first-order logic

Let $\mathcal{S} = \langle \mathcal{V}, \mathcal{C}, \mathcal{F}, \mathcal{R} \rangle$ be a signature.

Definition (satisfaction/truth of a formula)

Let $\mathcal{I} = \langle D, \cdot^{\mathcal{I}} \rangle$ be an interpretation for $\mathcal{S}$,
and let α be a variable assignment for $\mathcal{S}$ and domain D.
We say that $\mathcal{I}$ and α satisfy a first-order logic formula φ
(also: φ is true under $\mathcal{I}$ and α), in symbols: $\mathcal{I}, \alpha \models \varphi$,
according to the following inductive rules:

$$\begin{aligned} \mathcal{I}, \alpha \models \mathsf{R}(t_1, \dots, t_k) \quad & \text{iff } \langle t_1^{\mathcal{I},\alpha}, \dots, t_k^{\mathcal{I},\alpha} \rangle \in \mathsf{R}^{\mathcal{I}} \\ \mathcal{I}, \alpha \models t_1 = t_2 \quad & \text{iff } t_1^{\mathcal{I},\alpha} = t_2^{\mathcal{I},\alpha} \end{aligned}$$

...

Satisfaction/truth in first-order logic

Let $\mathcal{S} = \langle \mathcal{V}, \mathcal{C}, \mathcal{F}, \mathcal{R} \rangle$ be a signature.

Definition (satisfaction/truth of a formula)

...

$$\begin{aligned} \mathcal{I}, \alpha \models \forall x \varphi & \quad \text{iff } \mathcal{I}, \alpha[x := d] \models \varphi \text{ for all } d \in D \\ \mathcal{I}, \alpha \models \exists x \varphi & \quad \text{iff } \mathcal{I}, \alpha[x := d] \models \varphi \text{ for at least one } d \in D \end{aligned}$$

where $\alpha[x := d]$ is the variable assignment
which is the same as α except for x, where it assigns d. Formally:
$(\alpha[x := d])(z) = \begin{cases} d & \text{if } z = x \\ \alpha(z) & \text{if } z \neq x \end{cases}$
...

Satisfaction/truth in first-order logic

Let $\mathcal{S} = \langle \mathcal{V}, \mathcal{C}, \mathcal{F}, \mathcal{R} \rangle$ be a signature.

Definition (satisfaction/truth of a formula)

...

$$
\begin{array}{rl}
\mathcal{I}, \alpha \models \top & \text{always (i. e., for all } \mathcal{I}, \alpha) \\
\mathcal{I}, \alpha \models \bot & \text{never (i. e., for no } \mathcal{I}, \alpha) \\
\mathcal{I}, \alpha \models \neg\varphi & \text{iff } \mathcal{I}, \alpha \not\models \varphi \\
\mathcal{I}, \alpha \models \varphi \wedge \psi & \text{iff } \mathcal{I}, \alpha \models \varphi \text{ and } \mathcal{I}, \alpha \models \psi \\
\mathcal{I}, \alpha \models \varphi \vee \psi & \text{iff } \mathcal{I}, \alpha \models \varphi \text{ or } \mathcal{I}, \alpha \models \psi \\
\mathcal{I}, \alpha \models \varphi \rightarrow \psi & \text{iff } \mathcal{I}, \alpha \not\models \varphi \text{ or } \mathcal{I}, \alpha \models \psi \\
\mathcal{I}, \alpha \models \varphi \leftrightarrow \psi & \text{iff } (\mathcal{I}, \alpha \models \varphi \text{ and } \mathcal{I}, \alpha \models \psi) \text{ or} \\
 & \quad (\mathcal{I}, \alpha \not\models \varphi \text{ and } \mathcal{I}, \alpha \not\models \psi)
\end{array}
$$

Semantics of first-order logic: example

Example

Signature: $\mathcal{S} = \langle \mathcal{V}, \mathcal{C}, \mathcal{F}, \mathcal{R} \rangle$
with $\mathcal{V} = \{x, y, z\}$, $\mathcal{C} = \{\mathsf{a}, \mathsf{b}\}$, $\mathcal{F} = \emptyset$, $\mathcal{R} = \{\mathsf{Block}, \mathsf{Red}\}$,
$\mathit{arity}(\mathsf{Block}) = \mathit{arity}(\mathsf{Red}) = 1$.

$\mathcal{I} = \langle D, \cdot^{\mathcal{I}} \rangle$ with

- $D = \{d_1, d_2, d_3, d_4, d_5\}$
- $\mathsf{a}^{\mathcal{I}} = d_1$
- $\mathsf{b}^{\mathcal{I}} = d_3$
- $\mathsf{Block}^{\mathcal{I}} = \{d_1, d_2\}$
- $\mathsf{Red}^{\mathcal{I}} = \{d_1, d_2, d_3, d_5\}$

$\alpha = \{x \mapsto d_1, y \mapsto d_2, z \mapsto d_1\}$

Semantics of first-order logic: example (ctd.)

Example (ctd.)

Questions:

- $\mathcal{I}, \alpha \models \mathsf{Block}(\mathsf{b}) \lor \neg\mathsf{Block}(\mathsf{b})$?
- $\mathcal{I}, \alpha \models \mathsf{Block}(x) \rightarrow (\mathsf{Block}(x) \lor \neg\mathsf{Block}(y))$?
- $\mathcal{I}, \alpha \models \mathsf{Block}(\mathsf{a}) \land \mathsf{Block}(\mathsf{b})$?
- $\mathcal{I}, \alpha \models \forall x(\mathsf{Block}(x) \rightarrow \mathsf{Red}(x))$?

Satisfaction/truth of sets of formulae

Definition (satisfaction/truth of a set of formulae)

Consider a signature $\mathcal{S}$, a set of formulae Φ over $\mathcal{S}$,
an interpretation $\mathcal{I}$ for $\mathcal{S}$, and a variable assignment α for $\mathcal{S}$ and the domain of $\mathcal{I}$.

We say that $\mathcal{I}$ and α satisfy Φ (also: Φ is true under $\mathcal{I}$ and α), in symbols: $\mathcal{I}, \alpha \models \Phi$, if $\mathcal{I}, \alpha \models \varphi$ for all $\varphi \in \Phi$.

Free and bound variables: motivation

Question:

- Consider a signature with variable symbols $\{x_1, x_2, x_3, \dots\}$, and consider any interpretation $\mathcal{I}$.
- To decide if $\mathcal{I}, \alpha \models (\forall x_4(\mathsf{R}(x_4, x_2) \lor \mathsf{f}(x_3) = x_4)) \lor \exists x_3 \mathsf{S}(x_3, x_2)$, which parts of the definition of α matter?
- $\alpha(x_1)$, $\alpha(x_5)$, $\alpha(x_6)$, $\alpha(x_7)$, ... do not matter because these variable symbols do not occur in the formula
- $\alpha(x_4)$ does not matter either: it occurs in the formula, but all its occurrences are bound by a surrounding quantifier
- $\rightsquigarrow$ only the assignments to the free variables x_2 and x_3 matter

Variables of a term

Definition (variables of a term)

Let t be a term. The set of variables occurring in t, written $vars(t)$, is defined as follows:

- $vars(x) = \{x\}$ for variable symbols x
- $vars(\mathrm{c}) = \emptyset$ for constant symbols c
- $vars(\mathrm{f}(t_1, \dots, t_k)) = vars(t_1) \cup \dots \cup vars(t_k)$ for function terms

Example: $vars(\mathrm{product}(x, \mathrm{sum}(\mathrm{c}, y))) =$

Free and bound variables of a formula

Definition (free variables)

Let φ be a logical formula. The set of free variables of φ, written $\mathit{free}(\alpha)$, is defined as follows:

- $\mathit{free}(\mathrm{R}(t_1, \ldots, t_k)) = \mathit{vars}(t_1) \cup \cdots \cup \mathit{vars}(t_k)$
- $\mathit{free}(t_1 = t_2) = \mathit{vars}(t_1) \cup \mathit{vars}(t_2)$
- $\mathit{free}(\top) = \mathit{free}(\bot) = \emptyset$
- $\mathit{free}(\neg\varphi) = \mathit{free}(\varphi)$
- $\mathit{free}(\varphi \wedge \psi) = \mathit{free}(\varphi \vee \psi) = \mathit{free}(\varphi \rightarrow \psi)$ $= \mathit{free}(\varphi \leftrightarrow \psi) = \mathit{free}(\varphi) \cup \mathit{free}(\psi)$
- $\mathit{free}(\forall x\, \varphi) = \mathit{free}(\exists x\, \varphi) = \mathit{free}(\varphi) \setminus \{x\}$

Example: $\mathit{free}((\forall x_4(\mathrm{R}(x_4, x_2) \vee \mathrm{f}(x_3) = x_4)) \vee \exists x_3 \mathrm{S}(x_3, x_2))$
$=$

Closed formulae/sentences

Remark: Let φ be a formula, and let α and β be variable assignments such that $\alpha(x) = \beta(x)$ for all free variables of φ.

Then $\mathcal{I}, \alpha \models \varphi$ iff $\mathcal{I}, \beta \models \varphi$.

In particular, if $\mathit{free}(\varphi) = \emptyset$, then α does not matter at all.

Definition (closed formulae/sentences)

A formula φ with no free variables (i. e., $\mathit{free}(\varphi) = \emptyset$) is called a closed formula or sentence.

If φ is a sentence, we often use the notation $\mathcal{I} \models \varphi$ instead of $\mathcal{I}, \alpha \models \varphi$ because the definition of α does not affect whether or not φ is true under $\mathcal{I}$ and α.

Formulae with at least one free variable are called open.

Closed formulae: examples

Question: Which of the following formulae are sentences?

- $\text{Block(b)} \vee \neg\text{Block(b)}$
- $\text{Block}(x) \rightarrow (\text{Block}(x) \vee \neg\text{Block}(y))$
- $\text{Block(a)} \wedge \text{Block(b)}$
- $\forall x(\text{Block}(x) \rightarrow \text{Red}(x))$

Omitting signatures and domains

For convenience, from now on we implicitly assume that we use matching signatures and that variable assignments are defined for the correct domain.

Example: Instead of

> *Consider a signature $\mathcal{S}$, a set of formulae Φ over $\mathcal{S}$, an interpretation $\mathcal{I}$ for $\mathcal{S}$, and a variable assignment α for $\mathcal{S}$ and the domain of $\mathcal{I}$.*

we write:

> *Consider a set of formulae Φ, an interpretation $\mathcal{I}$ and a variable assignment α.*

More logic terminology

The terminology we introduced for propositional logic can be reused for first-order logic:

- interpretation $\mathcal{I}$ and variable assignment α form a model of formula φ if $\mathcal{I}, \alpha \models \varphi$.
- formula φ is satisfiable if $\mathcal{I}, \alpha \models \varphi$ for at least one $\mathcal{I}, \alpha$ (i. e., if it has a model)
- formula φ is falsifiable if $\mathcal{I}, \alpha \not\models \varphi$ for at least one $\mathcal{I}, \alpha$
- formula φ is valid if $\mathcal{I}, \alpha \models \varphi$ for all $\mathcal{I}, \alpha$
- formula φ is unsatisfiable if $\mathcal{I}, \alpha \not\models \varphi$ for all $\mathcal{I}, \alpha$
- formula φ entails (also: implies) formula ψ, written $\varphi \models \psi$, if all models of φ are models of ψ
- formulae φ and ψ are logically equivalent, written $\varphi \equiv \psi$, if they have the same models (equivalently: if $\varphi \models \psi$ and $\psi \models \varphi$)

Terminology for formula sets and sentences

- All concepts from the previous slide also apply to sets of formulae instead of single formulae.
  Examples:
  - formula set Φ is satisfiable if $\mathcal{I}, \alpha \models \Phi$ for at least one $\mathcal{I}, \alpha$
  - formula set Φ entails formula ψ, written $\Phi \models \psi$, if all models of Φ are models of ψ
  - formula set Φ entails formula set Ψ, written $\Phi \models \Psi$, if all models of Φ are models of Ψ
- All concepts apply to sentences (or sets of sentences) as a special case. In this case, we usually omit α.
  Examples:
  - interpretation $\mathcal{I}$ is a model of a sentence φ if $\mathcal{I} \models \varphi$
  - sentence φ is unsatisfiable if $\mathcal{I} \not\models \varphi$ for all $\mathcal{I}$

Going further

Using these definitions, we could discuss the same topics as for propositional logic, such as:

- important logical equivalences
- normal forms
- entailment theorems (deduction theorem etc.)
- proof calculi
- (first-order) resolution

We will mention a few basic results on these topics, but we do not cover them in detail.

Logical equivalences

- All propositional logic equivalences also apply to first-order logic (e. g., $\varphi \lor \psi \equiv \psi \lor \varphi$).
- Additionally, here are some equivalences and entailments involving quantifiers:

$(\forall x \varphi) \land (\forall x \psi)$	$\equiv \forall x(\varphi \land \psi)$	
$(\forall x \varphi) \lor (\forall x \psi)$	$\models \forall x(\varphi \lor \psi)$	but not vice versa
$(\forall x \varphi) \land \psi$	$\equiv \forall x(\varphi \land \psi)$	if $x \notin \mathit{free}(\psi)$
$(\forall x \varphi) \lor \psi$	$\equiv \forall x(\varphi \lor \psi)$	if $x \notin \mathit{free}(\psi)$
$\neg \forall x \varphi$	$\equiv \exists x \neg \varphi$	
$\exists x(\varphi \lor \psi)$	$\equiv (\exists x \varphi) \lor (\exists x \psi)$	
$\exists x(\varphi \land \psi)$	$\models (\exists x \varphi) \land (\exists x \psi)$	but not vice versa
$(\exists x \varphi) \lor \psi$	$\equiv \exists x(\varphi \lor \psi)$	if $x \notin \mathit{free}(\psi)$
$(\exists x \varphi) \land \psi$	$\equiv \exists x(\varphi \land \psi)$	if $x \notin \mathit{free}(\psi)$
$\neg \exists x \varphi$	$\equiv \forall x \neg \varphi$	

Normal forms

Similar to DNF and CNF for propositional logic, there are some important normal forms for first-order logic, such as:

- negation normal form (NNF):
  negation symbols may only occur in front of atoms
- prenex normal form:
  quantifiers must be the outermost parts of the formula
- Skolem normal form:
  prenex normal form with no existential quantifiers

Polynomial-time procedures transform formula φ

- into an equivalent formula in negation normal form,
- into an equivalent formula in prenex normal form, or
- into an equisatisfiable formula in Skolem normal form.

Entailment, proof systems, resolution...

- The deduction theorem, contraposition theorem and contradiction theorem also hold for first-order logic.
  (The same proofs can be used.)
- Sound and complete proof systems (calculi) exist for first-order logic (just like for propositional logic).
- Resolution can be generalized to first-order logic by using the concept of unification.
- This first-order resolution is refutation-complete, and hence with the contradiction theorem gives a general reasoning algorithm for first-order logic.
- However, the algorithm does not terminate on all inputs.

Summary

- First-order logic is a richer logic than propositional logic and allows us to reason about objects and their properties.
- Objects are denoted by terms built from variables, constants and function symbols.
- Properties are denoted by formulae built from predicates, quantification, and the usual logical operators such as negation, disjunction and conjunction.
- As with all logics, we analyze
  - syntax: what is a formula?
  - semantics: how do we interpret a formula?
  - reasoning methods: how can we prove logical consequences of a knowledge base?

We only scratched the surface. Further topics are discussed in the courses mentioned at the end of the previous chapter.