Theoretical Computer Science II (ACS II)

3. First-order logic

Bernhard Nebel    Christian Becker-Asano

Albert-Ludwigs-Universität Freiburg

November 14th, 2011
Introduction

Syntax

Semantics

Further topics

Wrap-up
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.

\(~\) first-order logic (predicate logic)
Elements of logic (recap)

The same questions as before:

- Which elements are well-formed? \(\leadsto\) syntax
- What does it mean for a formula to be true? \(\leadsto\) semantics
- When does one formula follow from another? \(\leadsto\) 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 $S = \langle \mathcal{V}, \mathcal{C}, \mathcal{F}, \mathcal{R} \rangle$ 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 $f \in \mathcal{F}$ and relation symbol $R \in \mathcal{R}$ has an associated arity (number of arguments) $\text{arity}(f), \text{arity}(R) \in \mathbb{N}_1$.

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

Also: unary, binary, ternary
Signatures: examples

Example: arithmetic

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

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

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

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

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

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

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

Examples:

- \( x_4 \)
- donald-duck
- \( \text{sum}(x_3, \text{product}(\text{one}, x_5)) \)
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:

- $R(t_1, \ldots, t_k)$ (atomic formula; atom)
  where $R \in \mathcal{R}$ 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 $\varphi$ is a formula over $\mathcal{S}$

- $\ldots$
Syntax of first-order logic: formulae

Definition (formula)

- ... 
- \( \top \) (truth)
- \( \bot \) (falseness)
- \( \neg \varphi \) (negation) where \( \varphi \) is a formula over \( S \)
- \( \varphi \land \psi \) (conjunction)
- \( \varphi \lor \psi \) (disjunction)
- \( \varphi \rightarrow \psi \) (material conditional)
- \( \varphi \leftrightarrow \psi \) (biconditional)

where \( \varphi \) and \( \psi \) are formulae over \( S \)
Syntax: examples

Example: arithmetic and genealogy

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

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

$\forall x P(x) \rightarrow Q(x)$ 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, sum(one, one), donald-duck
  - counterexamples: $x_4$, product($x$, zero)

- **similarly:** ground atom, ground formula
  - example: $\text{PerfectSquare}(\text{zero}) \lor \text{one} = \text{zero}$
  - counterexample: $\exists x \text{ one} = x$

**Abbreviation:**
sequences of quantifiers of the same kind can be collapsed

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

Sometimes commas and/or colons are used:

- $\forall x, y, z: \varphi$
- $\forall x_3, x_1 \exists x_2, x_5 \varphi$
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 $S = \langle \mathcal{V}, \mathcal{C}, \mathcal{F}, \mathcal{R} \rangle$ be a signature.

**Definition (interpretation, variable assignment)**

An interpretation (for $S$) 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:
  - $c^\mathcal{I} \in D$ for constant symbols $c \in \mathcal{C}$
  - $f^{\mathcal{I}} : D^k \to D$ for $k$-ary function symbols $f \in \mathcal{F}$
  - $R^{\mathcal{I}} \subseteq D^k$ for $k$-ary relation symbols $R \in \mathcal{R}$

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

**Idea:** extend $\mathcal{I}$ and $\alpha$ to general terms, then to atoms, then to arbitrary formulae
Semantics of first-order logic: informally

Example: \((\forall x \text{Block}(x) \rightarrow \text{Red}(x)) \land \text{Block}(a)\)

“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 \( S = \langle \mathcal{V}, \mathcal{C}, \mathcal{F}, \mathcal{R} \rangle \) be a signature.

**Definition (interpretation of a term)**

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

Let \( t \) be a term over \( S \).

The *interpretation of \( t \) under \( \mathcal{I} \) and \( \alpha \), 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 \( t = c \) with \( c \in \mathcal{C} \) (\( t \) is a constant term):
  \[ c^{\mathcal{I},\alpha} = c^\mathcal{I} \]

- If \( t = f(t_1, \ldots, t_k) \) (\( t \) is a function term):
  \[ (f(t_1, \ldots, t_k))^{\mathcal{I},\alpha} = f^{\mathcal{I}}(t_1^{\mathcal{I},\alpha}, \ldots, t_k^{\mathcal{I},\alpha}) \]
Interpreting terms: example

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

\( \mathcal{I} = \langle D, \cdot^{\mathcal{I}} \rangle \) with
- \( D = \{ d_0, d_1, d_2, d_3, d_4, d_5, d_6 \} \)
- \( \cdot^{\mathcal{I}} \text{zero} = d_0 \)
- \( \cdot^{\mathcal{I}} \text{one} = d_1 \)
- \( \cdot^{\mathcal{I}} (d_i, d_j) = d_{(i+j) \mod 7} \) for all \( i, j \in \{ 0, \ldots, 6 \} \)
- \( \cdot^{\mathcal{I}} (d_i, d_j) = d_{(i \cdot j) \mod 7} \) for all \( i, j \in \{ 0, \ldots, 6 \} \)

\( \alpha = \{ x \mapsto d_5, y \mapsto d_5, z \mapsto d_0 \} \)
Interpreting terms: example (ctd.)

Example (ctd.)

- $\text{zero}^{\mathcal{I}, \alpha} = \text{sum}(x, \text{zero})^{\mathcal{I}, \alpha} = \text{product(one, sum(x, zero))}^{\mathcal{I}, \alpha} = \frac{19}{38}$
Satisfaction/truth in first-order logic

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

**Definition (satisfaction/truth of a formula)**

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

$I, \alpha \models R(t_1, \ldots, t_k)$ iff $\langle t_1^{I,\alpha}, \ldots, t_k^{I,\alpha} \rangle \in R^I$

$I, \alpha \models t_1 = t_2$ iff $t_1^{I,\alpha} = t_2^{I,\alpha}$

...
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)**

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

where \( \alpha[x := d] \) is the variable assignment which is the same as \( \alpha \) 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)**

\[
\begin{align*}
\mathcal{I}, \alpha & \models \top \quad \text{always (i.e., for all } \mathcal{I}, \alpha) \\
\mathcal{I}, \alpha & \models \bot \quad \text{never (i.e., for no } \mathcal{I}, \alpha) \\
\mathcal{I}, \alpha & \models \neg \varphi \quad \text{iff } \mathcal{I}, \alpha \not\models \varphi \\
\mathcal{I}, \alpha & \models \varphi \land \psi \quad \text{iff } \mathcal{I}, \alpha \models \varphi \text{ and } \mathcal{I}, \alpha \models \psi \\
\mathcal{I}, \alpha & \models \varphi \lor \psi \quad \text{iff } \mathcal{I}, \alpha \not\models \varphi \text{ or } \mathcal{I}, \alpha \models \psi \\
\mathcal{I}, \alpha & \models \varphi \rightarrow \psi \quad \text{iff } \mathcal{I}, \alpha \not\models \varphi \text{ or } \mathcal{I}, \alpha \models \psi \\
\mathcal{I}, \alpha & \models \varphi \leftrightarrow \psi \quad \text{iff } (\mathcal{I}, \alpha \models \varphi \text{ and } \mathcal{I}, \alpha \models \psi) \text{ or } (\mathcal{I}, \alpha \not\models \varphi \text{ and } \mathcal{I}, \alpha \not\models \psi)
\end{align*}
\]
Example

Signature: $S = \langle \mathcal{V}, \mathcal{C}, \mathcal{F}, \mathcal{R} \rangle$

with $\mathcal{V} = \{x, y, z\}$, $\mathcal{C} = \{a, b\}$, $\mathcal{F} = \emptyset$, $\mathcal{R} = \{\text{Block, Red}\}$,

$\text{arity}(\text{Block}) = \text{arity}(\text{Red}) = 1$.

$I = \langle D, \cdot I \rangle$ with

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

$\alpha = \{x \mapsto d_1, y \mapsto d_2, z \mapsto d_1\}$
Semantics of first-order logic: example (ctd.)

Example (ctd.)

Questions:

- $\mathcal{I}, \alpha \models \text{Block}(b) \lor \neg \text{Block}(b)$?
- $\mathcal{I}, \alpha \models \text{Block}(x) \rightarrow (\text{Block}(x) \lor \neg \text{Block}(y))$?
- $\mathcal{I}, \alpha \models \text{Block}(a) \land \text{Block}(b)$?
- $\mathcal{I}, \alpha \models \forall x (\text{Block}(x) \rightarrow \text{Red}(x))$?
Satisfaction/truth of sets of formulae

Definition (satisfaction/truth of a set of formulae)
Consider a signature $S$, a set of formulae $\Phi$ over $S$, an interpretation $\mathcal{I}$ for $S$, and a variable assignment $\alpha$ for $S$ and the domain of $\mathcal{I}$.

We say that $\mathcal{I}$ and $\alpha$ satisfy $\Phi$ (also: $\Phi$ is true under $\mathcal{I}$ and $\alpha$), 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 \( \{ x_1, x_2, x_3, \ldots \} \), and consider any interpretation \( \mathcal{I} \).
- To decide if \( \mathcal{I}, \alpha \models (\forall x_4 (R(x_4, x_2) \lor f(x_3) = x_4)) \lor \exists x_3 S(x_3, x_2) \), which parts of the definition of \( \alpha \) matter?
- \( \alpha(x_1), \alpha(x_5), \alpha(x_6), \alpha(x_7), \ldots \) 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
- \( \leadsto \) 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 $\text{vars}(t)$, is defined as follows:

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

Example: $\text{vars}(\text{product}(x, \text{sum}(c, y))) =$
Free and bound variables of a formula

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

- $\text{free}(\text{R}(t_1, \ldots, t_k)) = \text{vars}(t_1) \cup \cdots \cup \text{vars}(t_k)$
- $\text{free}(t_1 = t_2) = \text{vars}(t_1) \cup \text{vars}(t_2)$
- $\text{free}(\top) = \text{free}(\bot) = \emptyset$
- $\text{free}(\neg \varphi) = \text{free}(\varphi)$
- $\text{free}(\varphi \land \psi) = \text{free}(\varphi \lor \psi) = \text{free}(\varphi \to \psi) = \text{free}(\varphi \leftrightarrow \psi) = \text{free}(\varphi) \cup \text{free}(\psi)$
- $\text{free}(\forall x \varphi) = \text{free}(\exists x \varphi) = \text{free}(\varphi) \setminus \{x\}$

Example: $\text{free}((\forall x_4 (\text{R}(x_4, x_2) \lor f(x_3) = x_4)) \lor \exists x_3 S(x_3, x_2))$
Closed formulae/sentences

Remark: Let $\varphi$ be a formula, and let $\alpha$ and $\beta$ be variable assignments such that $\alpha(x) = \beta(x)$ for all free variables of $\varphi$.
Then $I, \alpha \models \varphi$ iff $I, \beta \models \varphi$.

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

Definition (closed formulae/sentences)
A formula $\varphi$ with no free variables (i.e., $\text{free}(\varphi) = \emptyset$) is called a closed formula or sentence.

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

Formulae with at least one free variable are called open.
Closed formulae: examples

**Question:** Which of the following formulae are sentences?

- \( \text{Block}(b) \lor \neg \text{Block}(b) \)
- \( \text{Block}(x) \rightarrow (\text{Block}(x) \lor \neg \text{Block}(y)) \)
- \( \text{Block}(a) \land \text{Block}(b) \)
- \( \forall x (\text{Block}(x) \rightarrow \text{Red}(x)) \)
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 $S$, a set of formulae $\Phi$ over $S$, an interpretation $\mathcal{I}$ for $S$, and a variable assignment $\alpha$ for $S$ and the domain of $\mathcal{I}$.

we write:

Consider a set of formulae $\Phi$, an interpretation $\mathcal{I}$ and a variable assignment $\alpha$. 
More logic terminology

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

- interpretation $\mathcal{I}$ and variable assignment $\alpha$ form a **model** of formula $\varphi$ if $\mathcal{I}, \alpha \models \varphi$.
- formula $\varphi$ is **satisfiable** if $\mathcal{I}, \alpha \models \varphi$ for at least one $\mathcal{I}, \alpha$ (i.e., if it has a model)
- formula $\varphi$ is **falsifiable** if $\mathcal{I}, \alpha \not\models \varphi$ for at least one $\mathcal{I}, \alpha$
- formula $\varphi$ is **valid** if $\mathcal{I}, \alpha \models \varphi$ for all $\mathcal{I}, \alpha$
- formula $\varphi$ is **unsatisfiable** if $\mathcal{I}, \alpha \not\models \varphi$ for all $\mathcal{I}, \alpha$
- formula $\varphi$ **entails** (also: **implies**) formula $\psi$, written $\varphi \models \psi$, if all models of $\varphi$ are models of $\psi$
- formulae $\varphi$ and $\psi$ 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 $\Phi$ is satisfiable if $I, \alpha \models \Phi$ for at least one $I, \alpha$
  - formula set $\Phi$ entails formula $\psi$, written $\Phi \models \psi$,
    if all models of $\Phi$ are models of $\psi$
  - formula set $\Phi$ entails formula set $\Psi$, written $\Phi \models \Psi$,
    if all models of $\Phi$ are models of $\Psi$

- All concepts apply to sentences (or sets of sentences) as a special case. In this case, we usually omit $\alpha$.

Examples:
  - interpretation $I$ is a model of a sentence $\varphi$ if $I \models \varphi$
  - sentence $\varphi$ is unsatisfiable if $I \not\models \varphi$ for all $I$
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:

\[
\begin{align*}
(\forall x \varphi) \land (\forall x \psi) & \equiv \forall x (\varphi \land \psi) \\
(\forall x \varphi) \lor (\forall x \psi) & \models \forall x (\varphi \lor \psi) \\
(\forall x \varphi) \land \psi & \equiv \forall x (\varphi \land \psi) \\
(\forall x \varphi) \lor \psi & \equiv \forall x (\varphi \lor \psi) \\
\neg \forall x \varphi & \equiv \exists x \neg \varphi \\
\exists x (\varphi \lor \psi) & \equiv (\exists x \varphi) \lor (\exists x \psi) \\
\exists x (\varphi \land \psi) & \models (\exists x \varphi) \land (\exists x \psi) \\
(\exists x \varphi) \lor \psi & \equiv \exists x (\varphi \lor \psi) \\
(\exists x \varphi) \land \psi & \equiv \exists x (\varphi \land \psi) \\
\neg \exists x \varphi & \equiv \forall x \neg \varphi
\end{align*}
\]
Further topics

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 $\varphi$:

- 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.