FIRST-ORDER LOGIC

Semantics

Wolfgang Schreiner and Wolfgang Windsteiger
Wolfgang.(Schreiner|Windsteiger)@risc.jku.at

Research Institute for Symbolic Computation (RISC)
Johannes Kepler University (JKU), Linz, Austria
http://www.risc.jku.at
The Semantics of First-Order Logic

In first-order logic, the semantics (meaning) depends on a structure and an assignment.

- **A structure** \((D, I)\) consists of a **domain** \(D\) and an **interpretation** \(I\) on \(D\):
  - A **domain** is a non-empty collection of objects (e.g., a set \(D \neq \emptyset\)).
    - The “universe” about which a first-order logic formula talks.
  - An **interpretation** maps every constant and function/predicate symbol to its meaning:
    - **Constant** \(c \in C\): \(I(c)\) is an object in \(D (I(c) \in D)\).
    - **Function symbol** \(f \in F\) of arity \(n\): \(I(f)\) is an \(n\)-ary function on \(D (I(f) : D^n \to D)\).
    - **Predicate symbol** \(p \in P\) of arity \(n\): \(I(p)\) is an \(n\)-ary predicate/relation on \(D (I(p) \subseteq D^n)\).

- **An assignment** \(a\) maps every variable to its meaning:
  - **Variable** \(v \in V\): \(a(v)\) is an object in \(D (a(v) \in D)\).

\[
\begin{align*}
D &= \mathbb{N} \\
I &= [0 \mapsto \text{zero}, + \mapsto \text{add}, \ < \mapsto \text{less-than}, \ldots] \\
a &= [x \mapsto \text{one}, y \mapsto \text{zero}, z \mapsto \text{three}, \ldots]
\end{align*}
\]
Informal Semantics

Terms The meaning of a term is an object in $D$.
- The meaning of a variable $v$ is the object assigned to it by $a$, i.e., $a(v)$.
- The meaning of a constant $c$ is its interpretation in $I$, i.e., $I(c)$.
- The meaning of a function application $f(t_1,\ldots,t_n)$ is the result of applying its interpretation $I(f)$ to the meanings of $t_1,\ldots,t_n$.

Formulas The meaning of a formula is "true" or "false".
- The meaning of an atomic formula $p(t_1,\ldots,t_n)$ is the result of applying its interpretation $I(p)$ to the meanings of $t_1,\ldots,t_n$.
  - An equality $t_1 = t_2$ is "true", if $t_1$ has the same meaning as $t_2$.
- The meaning of the propositional constructions is as already known.
- $(\forall x: F)$ is true if $F$ is true for all possible objects assigned to $x$ in $a$.
- $(\exists x: F)$ is true if $F$ is true for some possible object assigned to $x$ in $a$. 
The Formal Semantics of Terms

\[(D, I) \xrightarrow{a} \llbracket t \rrbracket \xrightarrow{\quad d \in D \quad} \]

- **Term semantics** \( \llbracket t \rrbracket_{a}^{D, I} \in D \)
  - Given structure \((D, I)\) and assignment \(a\), the semantics of term \(t\) is an object in \(D\).
    \[
    t ::= v \mid c \mid f(t_1, \ldots, t_n)
    \]
  - The meaning of a **variable** is the value given by the assignment:
    \[
    \llbracket v \rrbracket_{a}^{D, I} := a(v)
    \]
  - The meaning of a **constant** is the value given by the interpretation:
    \[
    \llbracket c \rrbracket_{a}^{D, I} := I(c)
    \]
  - The meaning of a **function application** is the result of the interpretation of the function symbol applied to the values of the argument terms:
    \[
    \llbracket f(t_1, \ldots, t_n) \rrbracket_{a}^{D, I} := I(f)(\llbracket t_1 \rrbracket_{a}^{D, I}, \ldots, \llbracket t_n \rrbracket_{a}^{D, I})
    \]

The recursive definition of a function evaluating a term.
Example

\[ D = \mathbb{N} = \{ \text{zero, one, two, three, \ldots} \} \]

\[ I = [0 \mapsto \text{zero}, + \mapsto \text{add}, \ldots] \]

\[ a = [x \mapsto \text{one}, y \mapsto \text{two}, \ldots] \]

\[
\begin{align*}
\lfloor x + (y + 0) \rfloor_a^{D,I} &= \text{add}(\lfloor x \rfloor_a^{D,I}, \lfloor y + 0 \rfloor_a^{D,I}) \\
&= \text{add}(a(x), \lfloor y + 0 \rfloor_a^{D,I}) \\
&= \text{add}(\text{one}, \lfloor y + 0 \rfloor_a^{D,I}) \\
&= \text{add}(\text{one}, \text{add}(\lfloor y \rfloor_a^{D,I}, \lfloor 0 \rfloor_a^{D,I})) \\
&= \text{add}(\text{one}, \text{add}(a(y), I(0))) \\
&= \text{add}(\text{one}, \text{add}(\text{two}, \text{zero})) \\
&= \text{add}(\text{one}, \text{two}) \\
&= \text{three}.
\end{align*}
\]

The meaning of the term with the “usual” interpretation.
Example

\[ D = \mathcal{P}(\mathbb{N}) = \{\emptyset, \{\text{zero}\}, \{\text{one}\}, \{\text{two}\}, \ldots, \{\text{zero, one}\}, \ldots\} \]

\[ I = [0 \mapsto \emptyset, + \mapsto \text{union}, \ldots] \]

\[ a = [x \mapsto \{\text{one}\}, y \mapsto \{\text{two}\}, \ldots] \]

\[ \left\lfloor x + (y + 0) \right\rfloor_a^{D,I} = \text{union}(\left\lfloor x \right\rfloor_a^{D,I}, \left\lfloor y + 0 \right\rfloor_a^{D,I}) \]
\[ = \text{union}(a(x), \left\lfloor y + 0 \right\rfloor_a^{D,I}) \]
\[ = \text{union}(\{\text{one}\}, \left\lfloor y + 0 \right\rfloor_a^{D,I}) \]
\[ = \text{union}(\{\text{one}\}, \text{union}(\left\lfloor y \right\rfloor_a^{D,I}, \left\lfloor 0 \right\rfloor_a^{D,I})) \]
\[ = \text{union}(\{\text{one}\}, \text{union}(a(y), I(0))) \]
\[ = \text{union}(\{\text{one}\}, \text{union}(\{\text{two}\}, \emptyset)) \]
\[ = \text{union}(\{\text{one}\}, \{\text{two}\}) \]
\[ = \{\text{one, two}\} \]

The meaning of the term with another interpretation.
The Formal Semantics of Formulas

![Diagram](image)

**Formula semantics** $[[F]]^D_I_a \in \{true, false\}$

- Given structure $(D, I)$ and assignment $a$, the semantics of formula $F$ is a truth value.

$$F ::= p(t_1, \ldots, t_n) \mid \top \mid \bot \mid \ldots \mid (\forall v : F) \mid (\exists v : F)$$

- The meaning of an atomic formula is the result of the interpretation of the predicate symbol applied to the values of the argument terms (fixed interpretation of equality).

$$[[p(t_1, \ldots, t_n)]]^D_I_a := I(p)([[t_1]]^D_I_a, \ldots, [[t_n]]^D_I_a)$$

$$[[t_1 = t_2]]^D_I_a := \begin{cases} true & \text{if } [[t_1]]^D_I_a = [[t_2]]^D_I_a \\ false & \text{else} \end{cases}$$

- The meaning of the logical constants:

$$[[\top]]^D_I_a := true \quad [[\bot]]^D_I_a := false$$

The meaning of the basic formulas.
The Semantics of Propositional Formulas

The meaning of the logical connectives:

\[
\begin{align*}
\lbrack \neg F \rbrack_a^D, I & :=
\begin{cases}
    \text{true} & \text{if } \lbrack F \rbrack_a^D = \text{false} \\
    \text{false} & \text{else}
\end{cases} \\
\lbrack F_1 \land F_2 \rbrack_a^D, I & :=
\begin{cases}
    \text{true} & \text{if } \lbrack F_1 \rbrack_a^D = \lbrack F_2 \rbrack_a^D = \text{true} \\
    \text{false} & \text{else}
\end{cases} \\
\lbrack F_1 \lor F_2 \rbrack_a^D, I & :=
\begin{cases}
    \text{false} & \text{if } \lbrack F_1 \rbrack_a^D = \lbrack F_2 \rbrack_a^D = \text{false} \\
    \text{true} & \text{else}
\end{cases} \\
\lbrack F_1 \rightarrow F_2 \rbrack_a^D, I & :=
\begin{cases}
    \text{false} & \text{if } \lbrack F_1 \rbrack_a^D = \text{true and } \lbrack F_2 \rbrack_a^D = \text{false} \\
    \text{true} & \text{else}
\end{cases} \\
\lbrack F_1 \leftrightarrow F_2 \rbrack_a^D, I & :=
\begin{cases}
    \text{true} & \text{if } \lbrack F_1 \rbrack_a^D = \lbrack F_2 \rbrack_a^D \\
    \text{false} & \text{else}
\end{cases}
\end{align*}
\]

An embedding of the semantics of propositional logic into first-order logic.
The Semantics of Quantified Formulas

- \( (\forall x : F) \) is true, if \( F \) is true for every possible object \( d \) assigned to variable \( x \):

\[
\llbracket \forall x : F \rrbracket_{a}^{D,I} := \begin{cases} 
\text{true} & \text{if } \llbracket F \rrbracket_{a[x\mapsto d]}^{D,I} = \text{true for all } d \text{ in } D \\
\text{false} & \text{else}
\end{cases}
\]

- \( (\exists x : F) \) is true, if \( F \) is true for at least one possible object \( d \) assigned to variable \( x \):

\[
\llbracket \exists x : F \rrbracket_{a}^{D,I} := \begin{cases} 
\text{true} & \text{if } \llbracket F \rrbracket_{a[x\mapsto d]}^{D,I} = \text{true for some } d \text{ in } D \\
\text{false} & \text{else}
\end{cases}
\]

- Assignment \( a \) updated by the assignment of object \( d \) to variable \( x \):

\[
a[x \mapsto d](y) = \begin{cases} 
d & \text{if } x = y \\
a(y) & \text{else}
\end{cases}
\]

The core of the semantics of first-order logic.
Example

\[ D = \mathbb{N}_3 = \{\text{zero, one, two}\} \quad I = [0 \mapsto \text{zero, } + \mapsto \text{add, } \ldots] \quad a = [x \mapsto \text{one, } y \mapsto \text{two, } z \mapsto \text{two, } \ldots] \]

\[
\forall x: \exists y: x + y = z \]_{D,I}^a = ?

- \[ \exists y: x + y = z \]_{D,I}^a[x \mapsto \text{zero}] = true
  - \[ [x + y = z]_{D,I}^a[x \mapsto \text{zero}, y \mapsto \text{zero}] = \text{false} \]
  - \[ [x + y = z]_{D,I}^a[x \mapsto \text{zero}, y \mapsto \text{one}] = \text{false} \]
  - \[ [x + y = z]_{D,I}^a[x \mapsto \text{zero}, y \mapsto \text{two}] = \text{true} \]

- \[ \exists y: x + y = z \]_{D,I}^a[x \mapsto \text{one}] = true
  - \[ [x + y = z]_{D,I}^a[x \mapsto \text{one}, y \mapsto \text{zero}] = \text{false} \]
  - \[ [x + y = z]_{D,I}^a[x \mapsto \text{one}, y \mapsto \text{one}] = \text{true} \]
  - \[ [x + y = z]_{D,I}^a[x \mapsto \text{one}, y \mapsto \text{two}] = \text{false} \]

- \[ \exists y: x + y = z \]_{D,I}^a[x \mapsto \text{two}] = true
  - \[ [x + y = z]_{D,I}^a[x \mapsto \text{two}, y \mapsto \text{zero}] = \text{true} \]
  - \[ [x + y = z]_{D,I}^a[x \mapsto \text{two}, y \mapsto \text{one}] = \text{false} \]
  - \[ [x + y = z]_{D,I}^a[x \mapsto \text{two}, y \mapsto \text{two}] = \text{false} \]

\[ \forall x: \exists y: x + y = z \]_{D,I}^a = true.
Semantics: Structures and Assignments

- **∀n: R(n,n)**
  - The domain of natural numbers with R interpreted as the divisibility relation.
  - “Every natural number is divisible by itself”: true (for every assignment).

- **∀n: R(n,n)**
  - The domain of natural numbers with R interpreted as the less-than relation.
  - “Every natural number is less than itself”: false (for every assignment).

- **∃x: R(y,x) ∧ R(x,z)**
  - The domain of natural numbers with R interpreted as the less-than relation.
  - “There exists a natural number x with y < x and x < z”.
  - Assignment [y ↦ 2, z ↦ 4]: true (there is the value x = 3 with 2 < x and x < 4).
  - Assignment [y ↦ 2, z ↦ 3]: false (there is no value for x with 2 < x and x < 3).

The truth value of a formula depends on the structure and the assignment.
Semantics: Nested Quantifiers

Consider the domain of natural numbers with the usual interpretation of $<$. 

- $\left( \forall x : \exists y : x < y \right)$: true.
  - “For every natural number $x$ there exists some $y$ such that $x$ is less than $y$”.
  - For every natural number $x$, there is indeed such a $y$, namely $y := x + 1$.

- $\left( \exists y : \forall x : x < y \right)$: false
  - “There exists some natural number $y$ such that every $x$ is less than $y$.”
  - We assume that the formula is true and derive a contradiction. Because of the assumption, there exists some natural number $y$ such that $\left( \forall x : x < y \right)$ is true. But then, since $x < y$ is true for every value of $x$, it is also true for $x := y$. Thus $y < y$ is true, which we know to be false.

The order of nested quantifiers matters.
Semantic Notions: Satisfiability and Validity

Let $F$ denote a formula, $M = (D, I)$ a structure, $a$ an assignment.

**Satisfiability** Formula $F$ is *satisfiable*, if there exists some structure $M$ and assignment $a$ such that $\models M^a_a = true$.

- Example: $p(0, x)$ is satisfiable; $q(x) \land \neg q(x)$ is not.

**Model** Structure $M$ is a *model* of formula $F$, written as $M \models F$, if for every assignment $a$, we have $\models M^a_a = true$.

- Example: $(\mathbb{N}, [0 \mapsto \text{zero}, p \mapsto \text{less-equal}]) \models p(0, x)$

**Validity** Formula $F$ is *valid*, written as $\models F$, if every structure $M$ is a model of $F$, i.e., for every structure $M$ we have $M \models F$.

- Example: $\models p(x) \land (p(x) \to q(x)) \to q(x)$
Semantic Notions: Logical Consequence and Equivalence

**Logical Consequence** Formula $F_2$ is a logical consequence of formula $F_1$, written as $F_1 \models F_2$, if for every structure $M$ and assignment $a$, the following is true:
If $\llbracket F_1 \rrbracket_a^M = \text{true}$, then also $\llbracket F_2 \rrbracket_a^M = \text{true}$.

■ Example: $p(x) \land (p(x) \rightarrow q(x)) \models q(x)$

**Logical Consequence Generalized** Formula $F$ is a logical consequence of formulas $F_1, \ldots, F_n$, written $F_1, \ldots, F_n \models F$, if for every $M$ and $a$ the following is true:
If for every formula $F_i$ we have $\llbracket F_i \rrbracket_a^M = \text{true}$, then $\llbracket F \rrbracket_a^M = \text{true}$.

■ Example: $p(x), q(x) \models p(x) \land q(x)$

**Logical Equivalence** Formulas $F_1$ and $F_2$ are logically equivalent, written as $F_1 \iff F_2$, if and only if $F_1$ is a logical consequence of $F_2$ and vice versa, i.e., $F_1 \models F_2$ and $F_2 \models F_1$.

■ Example: $p(x) \rightarrow q(x) \iff \neg p(x) \lor q(x)$
Semantic Notions: Propositions

Satisfiability and Validity

- $F$ is satisfiable, if $\neg F$ is not valid.
- $F$ is valid, if $\neg F$ is not satisfiable.

Logical Consequence and Equivalence

- Formula $F_2$ is a logical consequence of formula $F_1$ (i.e., $F_1 \models F_2$) if and only if the formula $(F_1 \rightarrow F_2)$ is valid.
- Formula $F$ is a logical consequence of formulas $F_1, \ldots, F_n$ (i.e., $F_1, \ldots, F_n \models F$) if and only if the formula $(F_1 \wedge \ldots \wedge F_n \rightarrow F)$ is valid.
- Formula $F_1$ and formula $F_2$ are logically equivalent (i.e., $F_1 \iff F_2$) if and only if the formula $(F_1 \iff F_2)$ is valid.

Logical consequence/equivalence reduced to validity of an implication/equivalence.
Logical Equivalence: Formula Substitutions

Assume $F \iff F'$ and $G \iff G'$. Then we have the following equivalences:

- $\neg F \iff \neg F'$
- $F \land G \iff F' \land G'$
- $F \lor G \iff F' \lor G'$
- $F \rightarrow G \iff F' \rightarrow G'$
- $F \leftrightarrow G \iff F' \leftrightarrow G'$
- $\forall x: F \iff \forall x: F'$
- $\exists x: F \iff \exists x: F'$

Logically equivalent formulas can be substituted in any context.
Logical Equivalence: Rules

In addition to the logical equivalences for connectives in propositional logic:

\[ \neg \forall x: F \iff \exists x: \neg F \]  
\[ \neg \exists x: F \iff \forall x: \neg F \]  
\[ \forall x: (F_1 \land F_2) \iff (\forall x: F_1) \land (\forall x: F_2) \]  
\[ \exists x: (F_1 \lor F_2) \iff (\exists x: F_1) \lor (\exists x: F_2) \]  
\[ \forall x: (F_1 \lor F_2) \iff F_1 \lor (\forall x: F_2) \] if \( x \) is not free in \( F_1 \)  
\[ \exists x: (F_1 \land F_2) \iff F_1 \land (\exists x: F_2) \] if \( x \) is not free in \( F_1 \)  

For a finite domain \( \{v_1, \ldots, v_n\} \):

\[ \forall x: F \iff F[v_1/x] \land \ldots \land F[v_n/x] \]  
\[ \exists x: F \iff F[v_1/x] \lor \ldots \lor F[v_n/x] \]
Logical Equivalence: Examples

Push negations from the outside to the inside:

\[ \neg (\forall x: p(x) \rightarrow \exists y: q(x,y)) \iff \exists x: \neg (p(x) \rightarrow \exists y: q(x,y)) \]

\[ \iff \exists x: \neg ((\neg p(x)) \lor \exists y: q(x,y)) \]

\[ \iff \exists x: ((\neg \neg p(x)) \land \neg \exists y: q(x,y)) \]

\[ \iff \exists x: (p(x) \land \neg \exists y: q(x,y)) \]

\[ \iff \exists x: (p(x) \land \forall y: \neg q(x,y)) \]

Reduce the scope of quantifiers:

\[ \forall x, y: (p(x) \rightarrow q(x,y)) \iff \forall x, y: (\neg p(x) \lor q(x,y)) \]

\[ \iff \forall x: (\neg p(x) \lor \forall y: q(x,y)) \]

\[ \iff \forall x: (p(x) \rightarrow \forall y: q(x,y)) \]

Replace quantification in a finite domain \( D = \{0, 1, 2\} \):

\[ \forall x: p(x) \iff p(0) \land p(1) \land p(2) \]