2.1 Syntax of Predicate Logic

2.1 Syntax of Predicate Logic

Syntax: determines which symbols constitute the words of a language and in which order these symbols may occur.

First: define alphabet for formulas of predicate logic

Def 2.1.1 (Signature)

A signature \((\Sigma, \Delta)\) is a pair with \(\Sigma = \bigcup_{n \in \mathbb{N}} \Sigma_n\) and \(\Delta = \bigcup_{n \in \mathbb{N}} \Delta_n\). The sets \(\Sigma_n\) and \(\Delta_n\) are pairwise disjoint. Every \(f \in \Sigma_n\) is a function symbol of arity \(n\), every \(p \in \Delta_n\) is a predicate symbol of arity \(n\). The elements of \(\Sigma_0\) are also called constants. We always require \(\Sigma_0 \neq \emptyset\).

Ex. 2.1.2: Signature \((\Sigma, \Delta)\) for the logic prog. from Chapter 1. Here \(\Sigma = \Sigma_0 \cup \Sigma_3\), \(\Delta = \Delta_1 \cup \Delta_2\).

date is an additional fact. Symbol of arity 3

corresponds to dates consisting of day, month, year.

Fct symbols create objects (terms).
Def 2.13 (Terms)
Let $(\Sigma, \Delta)$ be a signature, let $\mathcal{V}$ be a set of variables with $\Sigma \cap \mathcal{V} = \emptyset$. Then $\mathcal{J}(\Sigma, \mathcal{V})$ is the set of all terms over $\Sigma$ and $\mathcal{V}$. $\mathcal{J}(\Sigma, \mathcal{V})$ is the smallest set such that:
- $\mathcal{V} \subseteq \mathcal{J}(\Sigma, \mathcal{V})$
- $f(t_1, \ldots, t_n) \in \mathcal{J}(\Sigma, \mathcal{V})$ if $f \in \Sigma_n$ and $t_1, \ldots, t_n \in \mathcal{J}(\Sigma, \mathcal{V})$ for some $n \in \mathbb{N}$.

$\mathcal{J}(\Sigma)$ stands for $\mathcal{J}(\Sigma, \emptyset)$, i.e., the set of ground terms (terms without variables).

For any term $t$, let $\mathcal{V}(t)$ be the set of all variables in $t$.

Ex 2.14 Let $\Sigma$ be as in Ex 2.1.2, let $\mathcal{V} = \{X, Y, Z, \ldots\}$.
Terms in $\mathcal{J}(\Sigma, \mathcal{V})$: $X, \text{monika}, 42, \text{date}(10, 4, 2015), \text{date}(X, \text{monika}, \text{date}(10, 4, 2015)), \ldots$

Def 2.15 (Formulas)
Let $(\Sigma, \Delta)$ be a signature and $\mathcal{V}$ be a set of variables.
The set of atomic formulas over $(\Sigma, \Delta)$ and $\mathcal{V}$ is defined as $\mathcal{A}(\Sigma, \Delta, \mathcal{V}) = \{ p(t_1, \ldots, t_n) | p \in \Delta_n, t_1, \ldots, t_n \in \mathcal{J}(\Sigma, \mathcal{V}) \}$.
$\mathcal{F}(\Sigma, \Delta, \mathcal{V})$ is the set of all formulas over $(\Sigma, \Delta)$ and $\mathcal{V}$. $\mathcal{F}(\Sigma, \Delta, \mathcal{V})$ is the smallest set such that:
- $\mathcal{A}(\Sigma, \Delta, \mathcal{V}) \subseteq \mathcal{F}(\Sigma, \Delta, \mathcal{V})$
- if $\varphi \in \mathcal{F}(\Sigma, \Delta, \mathcal{V})$, then $\neg \varphi \in \mathcal{F}(\Sigma, \Delta, \mathcal{V})$
- if $\varphi \in \mathcal{F}(\Sigma, \Delta, \mathcal{V})$, then $\varphi \Rightarrow \psi \in \mathcal{F}(\Sigma, \Delta, \mathcal{V})$
- $\varphi$ or $\psi$
\[ \text{if } \varphi \in \mathcal{F}(\Sigma, \Delta, \psi) \text{ then } \overrightarrow{\varphi} \in \mathcal{F}(\Sigma, \Delta, \psi) \]

or

\[ \text{if } \varphi_1, \varphi_2 \in \mathcal{F}(\Sigma, \Delta, \psi) \text{ then } (\varphi_1 \land \varphi_2), (\varphi_1 \lor \varphi_2), (\varphi_1 \rightarrow \varphi_2), \]

and

\[ (\varphi_1 \leftarrow \varphi_2) \in \mathcal{F}(\Sigma, \Delta, \psi) \]

"is equivalent to"

\[ \text{if } X \in \Theta \text{ and } \varphi \in \mathcal{F}(\Sigma, \Delta, \psi) \text{ then } (\forall X \varphi), (\exists X \varphi) \in \mathcal{F}(\Sigma, \Delta, \psi) \]

"for all" "exists"

For a formula \( \varphi \), \( \mathcal{V}(\varphi) \) is the set of variables occurring in \( \varphi \).

A variable \( X \) occurs free in a formula \( \varphi \) iff

- \( \varphi \) is an atomic formula and \( X \in \mathcal{V}(\varphi) \)
- \( \varphi = \neg \varphi_1 \) and \( X \) occurs free in \( \varphi_1 \)
- \( \varphi = (\varphi_1 \circ \varphi_2) \) with \( \circ \in \{ \land, \lor, \rightarrow, \leftrightarrow \} \) and \( X \) occurs free in \( \varphi_1 \) or \( \varphi_2 \)
- \( \varphi = (Q \varphi \varphi_1) \) with \( Q \in \{ \forall, \exists \} \), \( X \) occurs free in \( \varphi_1 \), and \( X \neq \gamma \).

A formula is closed if it does not contain free variables.

A formula is quantifier-free if it does not contain \( \forall \) or \( \exists \).

We usually omit ( \(...\) ) whenever possible.

**Ex 2.16** We use the signature of Ex. 2.12.

female (monika) \( \in \mathcal{A}(\Sigma, \Delta, \psi) \)

mother of \( (X, \text{ susanne}) \in \mathcal{A}(\Sigma, \Delta, \psi) \)

born (monika, date(15,10,1966)) \( \in \mathcal{A}(\Sigma, \Delta, \psi) \)
\[ \forall W \ (\text{married}(\text{gard}, W) \land \text{motherOf}(W, C)) \in \mathcal{F}(\Sigma, \Delta, \mathcal{V}) \]

\text{gard} is married with all women \(W\) and they all are the mother of \(C\)

only free variable: \(C\)

\[ \forall W \ (\text{married}(\text{gard}, W) \land \forall W \ \text{motherOf}(W, C)) \in \mathcal{F}(\Sigma, \Delta, \mathcal{V}) \]

free variables: \(W, C\)

We abbreviate \( \forall X_1 (\ldots (\forall X_n \ y) \ldots) \) by \( \forall X_1, \ldots, X_n \ y \)
\( \exists X_1 (\ldots (\exists X_n \ y) \ldots) \) by \( \exists X_1, \ldots, X_n \ y \)

To distinguish variables from fct. and pred. symbols:
- Variables start with upper-case letters
- fct. + pred. symbols start with lower-case letters

**Ex 2.17** Every logic program stands for a set of formulas. Here, the variables are universally quantified (i.e., with \( \forall \)). Variables in terms and formulas stand for arbitrary objects \( \Rightarrow \) they can be substituted by objects (i.e., by terms).

**Def 2.1.8 (Substitution)**
A mapping \( \sigma : \mathcal{V} \rightarrow \mathcal{V}(\Sigma, \Delta) \) is a substitution iff
\( \sigma(\mathcal{V}) \neq \mathcal{V} \) holds for finitely many \( \forall X \in \mathcal{V} \cdot \text{DOM}(\sigma) = \{X \in \mathcal{V} \mid \sigma(X) \neq X\} \) is the domain of \( \sigma \) and \( \text{RANGE}(\sigma) = \{\sigma(X) \mid X \in \text{DOM}(\sigma)\} \) is the range of \( \sigma \).
A substitution can be denoted as \( \{X / \sigma(X) \mid X \in \text{DOM}(\sigma)\} \).
A subst. \( \sigma \) is a ground substitution \( \Rightarrow \sigma = \{\} \) or \( \{X / X \mid X \in \mathcal{V}\} \).
A substitution $\sigma$ is a ground substitution iff $\sigma(X)$ contains no variables for all $X \in \text{Dom}(\sigma)$.

A substitution $\sigma$ is a variable renaming iff it is injective and $\sigma(X) \notin \mathcal{V}$ for all $X \in \mathcal{V}$.

$$\begin{align*}
\text{Ex: } \sigma &= \{X / Y, Y / Z, Z / X\} \\
\sigma(X) &= Y \\
\sigma(Y) &= Z \\
\sigma(Z) &= X \\
\sigma(U) &= U
\end{align*}$$

Substitutions can be extended to terms, i.e., $\sigma : \mathcal{T}(\Sigma, \mathcal{V}) \rightarrow \mathcal{T}(\Sigma, \mathcal{V})$:

$$\sigma(f(t_1, \ldots, t_n)) = f(\sigma(t_1), \ldots, \sigma(t_n))$$

**Ex:**

$$\sigma(\text{date}(X, \text{monika}, Y)) = \\
\text{date}(\sigma(X), \text{monika}, \sigma(Y)) = \\
\text{date}(Y, \text{monika}, Z)$$

Substitutions can also be extended to formulas:

- $\sigma(\varphi(t_1, \ldots, t_n)) = \varphi(\sigma(t_1), \ldots, \sigma(t_n))$
- $\sigma(\neg \varphi) = \neg \sigma(\varphi)$
- $\sigma(\varphi_1 \circ \varphi_2) = \sigma(\varphi_1) \circ \sigma(\varphi_2)$ for $\circ \in \land, \lor, \Rightarrow, \Leftrightarrow$
- $\sigma(QX \varphi) = QX \sigma(\varphi)$ for $Q \in \{\forall, \exists\}$

Reason: $\forall X \text{human}(X)$ and $\forall Y \text{human}(Y)$ should be treated in the same way.

$\Rightarrow$ if $\sigma$ modifies $X$ or if the application of $\sigma$
\[ \sigma(QX \Psi) = QX' \sigma'(\Psi) \] for \( Q \in \{ V, F \} \), \( X \in \text{dom}(\sigma) \cup \text{range}(\sigma) \).

Here, \( X' \) is a fresh variable with \( X' \notin \text{dom}(\sigma) \cup \text{range}(\sigma) \cup \Sigma(\Psi) \) and \( \delta = \{ X/X' \} \).

**Ex. 2.15** \( \sigma = \{ X/\text{date}(X, y, z), Y/\text{monika}, Z/\text{date}(Z, z, z) \} \)

\[ \sigma(\text{date}(X, y, z)) = \text{date}(\text{date}(X, y, z), \text{monika}, \text{date}(Z, z, z)) \]

\[ \sigma(\forall Y \text{ married}(X, Y)) = \]

\[ \sigma(\forall Y' \text{ married}(X, Y')) = \]

\[ \forall Y' \text{ married}(\text{date}(X, y, z), Y') \]

An instance \( \sigma(t) \) or \( \sigma(\Psi) \) of a term \( t \) (resp. a formula \( \Psi \)) is a ground instance if it doesn't contain variables.