Atoms

Atoms (also called atomic formulas) over Σ are formed according to this syntax:

$$\begin{array}{rcll} A, B & ::= & P(s_1, \ldots, s_m) \ , \ P/m \in \Pi & \text{(non-equational atom)} \\ & \Big[\mid & (s \approx t) & \text{(equation)} \ \Big] \end{array}$$

Whenever we admit equations as atomic formulas we are in the realm of *first-order logic with equality*. Admitting equality does not really increase the expressiveness of first-order logic, (cf. exercises). But deductive systems where equality is treated specifically are much more efficient.

Literals

$$\begin{array}{rcll} L & ::= & A & \text{(positive literal)} \\ & \mid & \neg A & \text{(negative literal)} \end{array}$$

Clauses

$$\begin{array}{rcll} C, D & ::= & \bot & \text{(empty clause)} \\ & \mid & L_1 \vee \ldots \vee L_k, \ k \geq 1 & \text{(non-empty clause)} \end{array}$$

General First-Order Formulas

$\mathrm{F}_\Sigma(X)$ is the set of first-order formulas over Σ defined as follows:

$$\begin{array}{rcll} \phi, \psi, \chi & ::= & \bot & \text{(falsum)} \\ & \mid & \top & \text{(verum)} \\ & \mid & A & \text{(atomic formula)} \\ & \mid & \neg\phi & \text{(negation)} \\ & \mid & (\phi \wedge \psi) & \text{(conjunction)} \\ & \mid & (\phi \vee \psi) & \text{(disjunction)} \\ & \mid & (\phi \rightarrow \psi) & \text{(implication)} \\ & \mid & (\phi \leftrightarrow \psi) & \text{(equivalence)} \\ & \mid & \forall x \phi & \text{(universal quantification)} \\ & \mid & \exists x \phi & \text{(existential quantification)} \end{array}$$

Notational Conventions

We omit brackets according to the conventions for propositional logic.

Furthermore, $\forall x_1, \ldots, x_n\, \phi$ $(\exists x_1, \ldots, x_n\, \phi)$ abbreviates $\forall x_1 \ldots \forall x_n\, \phi$ $(\exists x_1 \ldots \exists x_n\, \phi)$.

We use infix-, prefix-, postfix-, or mixfix-notation with the usual operator precedences.

Examples:

$s + t * u$	for	$+(s, *(t, u))$
$s * u \leq t + v$	for	$\leq (*(s, u), +(t, v))$
$-s$	for	$-(s)$
0	for	$0()$

Example: Peano Arithmetic

$\Sigma_{PA} = (\Omega_{PA},\ \Pi_{PA})$
$\Omega_{PA} = \{0/0,\ +/2,\ */2,\ s/1\}$
$\Pi_{PA} = \{\leq/2,\ </2\}$
$+, *, <, \leq$ infix; $*\ >_p\ +\ >_p\ <\ >_p\ \leq$

Examples of formulas over this signature are:

$\forall x, y(x \leq y \leftrightarrow \exists z(x + z \approx y))$
$\exists x \forall y(x + y \approx y)$
$\forall x, y(x * s(y) \approx x * y + x)$
$\forall x, y(s(x) \approx s(y) \rightarrow x \approx y)$
$\forall x \exists y(x < y \wedge \neg \exists z(x < z \wedge z < y))$

Remarks About the Example

We observe that the symbols $\leq,\ <,\ 0,\ s$ are redundant as they can be defined in first-order logic with equality just with the help of $+$. The first formula defines $\leq$, while the second defines zero. The last formula, respectively, defines s.

Eliminating the existential quantifiers by Skolemization (cf. below) reintroduces the "redundant" symbols.

Consequently there is a *trade-off* between the complexity of the quantification structure and the complexity of the signature.

Positions in Terms and Formulas

The set of positions is extended from propositional logic to first-order logic:

The *Positions* of a term s (formula ϕ):

$\mathrm{pos}(x) = \{\varepsilon\}$,
$\mathrm{pos}(f(s_1, \ldots, s_n)) = \{\varepsilon\} \cup \bigcup_{i=1}^{n} \{\, ip \mid p \in \mathrm{pos}(s_i) \,\}$.

$\mathrm{pos}(P(t_1, \ldots, t_n)) = \{\varepsilon\} \cup \bigcup_{i=1}^{n} \{\, ip \mid p \in \mathrm{pos}(t_i) \,\}$,
$\mathrm{pos}(\forall x \phi) = \{\varepsilon\} \cup \{\, 1p \mid p \in \mathrm{pos}(\phi) \,\}$,
$\mathrm{pos}(\exists x \phi) = \{\varepsilon\} \cup \{\, 1p \mid p \in \mathrm{pos}(\phi) \,\}$.

The prefix order $\leq$, the subformula (subterm) operator, the formula (term) replacement operator and the size operator are extended accordingly. See the definitions in the propositional logic section.

Bound and Free Variables

In $Qx\phi$, $Q \in \{\exists, \forall\}$, we call ϕ the *scope* of the quantifier Qx. An *occurrence* of a variable x is called *bound*, if it is inside the scope of a quantifier Qx. Any other occurrence of a variable is called *free*.

Formulas without free variables are also called *closed formulas* or *sentential forms*.

Formulas without variables are called *ground*.

Example:

$$\overbrace{\forall y \quad (\forall \overbrace{x \quad P(x)}^{\text{scope}} \quad \rightarrow \quad Q(x, y))}^{\text{scope}}$$

The occurrence of y is bound, as is the first occurrence of x. The second occurrence of x is a free occurrence.

Substitutions

Substitution is a fundamental operation on terms and formulas that occurs in all inference systems for first-order logic.

In general, *substitutions* are mappings

$$\sigma : X \rightarrow \mathrm{T}_\Sigma(X)$$

such that the *domain* of σ, that is, the set

$$dom(\sigma) = \{\, x \in X \mid \sigma(x) \neq x \,\},$$

is finite. The set of variables *introduced* by σ, that is, the set of variables occurring in one of the terms $\sigma(x)$, with $x \in dom(\sigma)$, is denoted by $codom(\sigma)$.

Substitutions are often written as $\{x_1 \mapsto s_1, \ldots, x_n \mapsto s_n\}$, with x_i pairwise distinct, and then denote the mapping

$$\{x_1 \mapsto s_1, \ldots, x_n \mapsto s_n\}(y) = \begin{cases} s_i, & \text{if } y = x_i \\ y, & \text{otherwise} \end{cases}$$

We also write $x\sigma$ for $\sigma(x)$.

The *modification* of a substitution σ at x is defined as follows:

$$\sigma[x \mapsto t](y) = \begin{cases} t, & \text{if } y = x \\ \sigma(y), & \text{otherwise} \end{cases}$$

Why Substitution is Complicated

We define the application of a substitution σ to a term t or formula ϕ by structural induction over the syntactic structure of t or ϕ by the equations depicted on the next page.

In the presence of quantification it is surprisingly complex: We need to make sure that the (free) variables in the codomain of σ are not *captured* upon placing them into the scope of a quantifier Qy, hence the bound variable must be renamed into a "fresh", that is, previously unused, variable z.

Why this definition of substitution is well-defined will be discussed below.

Application of a Substitution

"Homomorphic" extension of σ to terms and formulas:

$$
\begin{aligned}
f(s_1, \ldots, s_n)\sigma &= f(s_1\sigma, \ldots, s_n\sigma) \\
\bot\sigma &= \bot \\
\top\sigma &= \top \\
P(s_1, \ldots, s_n)\sigma &= P(s_1\sigma, \ldots, s_n\sigma) \\
(u \approx v)\sigma &= (u\sigma \approx v\sigma) \\
\neg\phi\sigma &= \neg(\phi\sigma) \\
(\phi\rho\psi)\sigma &= (\phi\sigma\, \rho\, \psi\sigma)\ ; \quad \text{for each binary connective } \rho \\
(Qx\,\phi)\sigma &= Qz\,(\phi\,\sigma[x \mapsto z])\ ; \quad \text{with } z \text{ a fresh variable}
\end{aligned}
$$

Structural Induction

Proposition 3.1 *Let $G = (N, T, P, S)$ be a context-free grammar (possibly infinite) and let q be a property of T^* (the words over the alphabet T of terminal symbols of G).*

q holds for all *words $w \in L(G)$, whenever one can prove the following two properties:*

1. *(*base cases*)*
   $q(w')$ holds for each $w' \in T^*$ such that $X ::= w'$ is a rule in P.
2. *(*step cases*)*
   If $X ::= w_0 X_0 w_1 \ldots w_n X_n w_{n+1}$ is in P with $X_i \in N$, $w_i \in T^*$, $n \geq 0$, then for all $w'_i \in L(G, X_i)$, whenever $q(w'_i)$ holds for $0 \leq i \leq n$, then also $q(w_0 w'_0 w_1 \ldots w_n w'_n w_{n+1})$ holds.

Here $L(G, X_i) \subseteq T^*$ denotes the language generated by the grammar G from the non-terminal X_i.

Structural Recursion

Proposition 3.2 *Let $G = (N, T, P, S)$ be a* unambiguous *(why?) context-free grammar. A function f is well-defined on $L(G)$ (that is, unambiguously defined) whenever these 2 properties are satisfied:*

1. *(base cases)*
   f is well-defined on the words $w' \in T^*$ for each rule $X ::= w'$ in P.

2. *(step cases)*
   If $X ::= w_0 X_0 w_1 \ldots w_n X_n w_{n+1}$ *is a rule in* P *then* $f(w_0 w_0' w_1 \ldots w_n w_n' w_{n+1})$ *is well-defined, assuming that each of the* $f(w_i')$ *is well-defined.*

Substitution Revisited

Q: Does Proposition 3.2 justify that our homomorphic extension

$$\mathit{apply} : \mathrm{F}_\Sigma(X) \times (X \to \mathrm{T}_\Sigma(X)) \quad \to \quad \mathrm{F}_\Sigma(X),$$

with $\mathit{apply}(\phi, \sigma)$ denoted by $\phi\sigma$, of a substitution is well-defined?

A: We have two problems here. One is that *"fresh"* is (deliberately) left unspecified. That can be easily fixed by adding an extra variable counter argument to the apply function.

The second problem is that Proposition 3.2 applies to unary functions only. The standard solution to this problem is to curryfy, that is, to consider the binary function as a unary function producing a unary (residual) function as a result:

$$\mathit{apply} : \mathrm{F}_\Sigma(X) \quad \to \quad ((X \to \mathrm{T}_\Sigma(X)) \to \mathrm{F}_\Sigma(X))$$

where we have denoted $(\mathit{apply}(\phi))(\sigma)$ as $\phi\sigma$.

3.2 Semantics

To give semantics to a logical system means to define a notion of truth for the formulas. The concept of truth that we will now define for first-order logic goes back to Tarski.

As in the propositional case, we use a two-valued logic with truth values "true" and "false" denoted by 1 and 0, respectively.

Structures

A Σ-*algebra* (also called Σ-interpretation or Σ-structure) is a triple

$$\mathcal{A} = (U_\mathcal{A},\ (f_\mathcal{A} : U_\mathcal{A}^n \to U_\mathcal{A})_{f/n \in \Omega},\ (P_\mathcal{A} \subseteq U_\mathcal{A}^m)_{P/m \in \Pi})$$

where $U_\mathcal{A} \neq \emptyset$ is a set, called the *universe* of $\mathcal{A}$.

By Σ-*Alg* we denote the class of all Σ-algebras.

Assignments

A variable has no intrinsic meaning. The meaning of a variable has to be defined externally (explicitly or implicitly in a given context) by an assignment.

A *(variable) assignment*, also called a *valuation* (over a given Σ-algebra $\mathcal{A}$), is a map $\beta : X \to U_{\mathcal{A}}$.

Variable assignments are the semantic counterparts of substitutions.

Value of a Term in $\mathcal{A}$ with Respect to β

By structural induction we define

$$\mathcal{A}(\beta) : \mathrm{T}_\Sigma(X) \to U_{\mathcal{A}}$$

as follows:

$$\begin{aligned} \mathcal{A}(\beta)(x) &= \beta(x), && x \in X \\ \mathcal{A}(\beta)(f(s_1, \ldots, s_n)) &= f_{\mathcal{A}}(\mathcal{A}(\beta)(s_1), \ldots, \mathcal{A}(\beta)(s_n)), && f/n \in \Omega \end{aligned}$$

In the scope of a quantifier we need to evaluate terms with respect to modified assignments. To that end, let $\beta[x \mapsto a] : X \to U_{\mathcal{A}}$, for $x \in X$ and $a \in \mathcal{A}$, denote the assignment

$$\beta[x \mapsto a](y) = \begin{cases} a & \text{if } x = y \\ \beta(y) & \text{otherwise} \end{cases}$$

Truth Value of a Formula in $\mathcal{A}$ with Respect to β

$\mathcal{A}(\beta) : \mathrm{F}_\Sigma(X) \to \{0, 1\}$ is defined inductively as follows:

$$\begin{aligned} \mathcal{A}(\beta)(\bot) &= 0 \\ \mathcal{A}(\beta)(\top) &= 1 \\ \mathcal{A}(\beta)(P(s_1, \ldots, s_n)) = 1 &\quad\Leftrightarrow\quad (\mathcal{A}(\beta)(s_1), \ldots, \mathcal{A}(\beta)(s_n)) \in P_{\mathcal{A}} \\ \mathcal{A}(\beta)(s \approx t) = 1 &\quad\Leftrightarrow\quad \mathcal{A}(\beta)(s) = \mathcal{A}(\beta)(t) \\ \mathcal{A}(\beta)(\neg\phi) = 1 &\quad\Leftrightarrow\quad \mathcal{A}(\beta)(\phi) = 0 \\ \mathcal{A}(\beta)(\phi\rho\psi) &= \mathsf{B}_\rho(\mathcal{A}(\beta)(\phi), \mathcal{A}(\beta)(\psi)) \\ &\quad \text{with } \mathsf{B}_\rho \text{ the Boolean function associated with } \rho \\ \mathcal{A}(\beta)(\forall x\phi) &= \min_{a \in U}\{\mathcal{A}(\beta[x \mapsto a])(\phi)\} \\ \mathcal{A}(\beta)(\exists x\phi) &= \max_{a \in U}\{\mathcal{A}(\beta[x \mapsto a])(\phi)\} \end{aligned}$$

Example

The "Standard" Interpretation for Peano Arithmetic:

$$\begin{array}{rcl} U_{\mathbb{N}} & = & \{0, 1, 2, \ldots\} \\ 0_{\mathbb{N}} & = & 0 \\ s_{\mathbb{N}} & : & n \mapsto n + 1 \\ +_{\mathbb{N}} & : & (n, m) \mapsto n + m \\ *_{\mathbb{N}} & : & (n, m) \mapsto n * m \\ \leq_{\mathbb{N}} & = & \{\, (n, m) \mid n \text{ less than or equal to } m \,\} \\ <_{\mathbb{N}} & = & \{\, (n, m) \mid n \text{ less than } m \,\} \end{array}$$

Note that $\mathbb{N}$ is just one out of many possible Σ_{PA}-interpretations.

Values over $\mathbb{N}$ for Sample Terms and Formulas:

Under the assignment $\beta : x \mapsto 1, y \mapsto 3$ we obtain

$$\begin{array}{lcl} \mathbb{N}(\beta)(s(x) + s(0)) & = & 3 \\ \mathbb{N}(\beta)(x + y \approx s(y)) & = & 1 \\ \mathbb{N}(\beta)(\forall x, y(x + y \approx y + x)) & = & 1 \\ \mathbb{N}(\beta)(\forall z\ z \leq y) & = & 0 \\ \mathbb{N}(\beta)(\forall x \exists y\ x < y) & = & 1 \end{array}$$

3.3 Models, Validity, and Satisfiability

F is *valid* in $\mathcal{A}$ under assignment β:

$$\mathcal{A}, \beta \models F \quad :\Leftrightarrow \quad \mathcal{A}(\beta)(F) = 1$$

F is *valid* in $\mathcal{A}$ ($\mathcal{A}$ is a *model* of F):

$$\mathcal{A} \models F \quad :\Leftrightarrow \quad \mathcal{A}, \beta \models F, \text{ for all } \beta \in X \to U_{\mathcal{A}}$$

F is *valid* (or is a *tautology*):

$$\models F \quad :\Leftrightarrow \quad \mathcal{A} \models F, \text{ for all } \mathcal{A} \in \Sigma\text{-Alg}$$

F is called *satisfiable* iff there exist $\mathcal{A}$ and β such that $\mathcal{A}, \beta \models F$. Otherwise F is called *unsatisfiable*.