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

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

with $apply(F, \sigma)$ denoted by $F\sigma$, 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:

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

where we have denoted $(apply(F))(\sigma)$ as $F\sigma$.

E: Convince yourself that this does the trick.

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

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

where $\mathsf{arity}(f) = n$, $\mathsf{arity}(P) = m$, $U_{\mathcal{A}} \neq \emptyset$ 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:

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

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:

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

Example

The "Standard" Interpretation for Peano Arithmetic:

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

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

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

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*):

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

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

Substitution Lemma

The following propositions, to be proved by structural induction, hold for all Σ-algebras $\mathcal{A}$, assignments β, and substitutions σ.

Lemma 3.3 *For any Σ-term t*

$$\mathcal{A}(\beta)(t\sigma) = \mathcal{A}(\beta \circ \sigma)(t),$$

where $\beta \circ \sigma : X \to \mathcal{A}$ is the assignment $\beta \circ \sigma(x) = \mathcal{A}(\beta)(x\sigma)$.

Proposition 3.4 *For any Σ-formula F, $\mathcal{A}(\beta)(F\sigma) = \mathcal{A}(\beta \circ \sigma)(F)$.*

Corollary 3.5 $\mathcal{A}, \beta \models F\sigma \quad \Leftrightarrow \quad \mathcal{A}, \beta \circ \sigma \models F$

These theorems basically express that the syntactic concept of substitution corresponds to the semantic concept of an assignment.

Entailment and Equivalence

F entails (implies) G (or *G is a consequence of F*), written $F \models G$, if for all $\mathcal{A} \in \Sigma\text{-}\mathsf{Alg}$ and $\beta \in X \to U_{\mathcal{A}}$, whenever $\mathcal{A}, \beta \models F$, then $\mathcal{A}, \beta \models G$.

F and G are called *equivalent*, written $F \models\!\models G$, if for all $\mathcal{A} \in \Sigma\text{-}\mathsf{Alg}$ und $\beta \in X \to U_{\mathcal{A}}$ we have $\mathcal{A}, \beta \models F \quad \Leftrightarrow \quad \mathcal{A}, \beta \models G$.

Proposition 3.6 *F entails G iff $(F \to G)$ is valid*

Proposition 3.7 *F and G are equivalent iff $(F \leftrightarrow G)$ is valid.*

Extension to sets of formulas N in the "natural way", e. g., $N \models F$

$:\Leftrightarrow$ for all $\mathcal{A} \in \Sigma\text{-}\mathsf{Alg}$ and $\beta \in X \to U_{\mathcal{A}}$: if $\mathcal{A}, \beta \models G$, for all $G \in N$, then $\mathcal{A}, \beta \models F$.

Validity vs. Unsatisfiability

Validity and unsatisfiability are just two sides of the same medal as explained by the following proposition.

Proposition 3.8 *Let F and G be formulas, let N be a set of formulas. Then*

(i) F is valid if and only if $\neg F$ is unsatisfiable.

(ii) $F \models G$ if and only if $F \wedge \neg G$ is unsatisfiable.

(iii) $N \models G$ if and only if $N \cup \{\neg G\}$ is unsatisfiable.

Hence in order to design a theorem prover (validity checker) it is sufficient to design a checker for unsatisfiability.

Theory of a Structure

Let $\mathcal{A} \in \Sigma\text{-}\mathsf{Alg}$. The *(first-order) theory* of $\mathcal{A}$ is defined as

$$Th(\mathcal{A}) = \{G \in \mathrm{F}_\Sigma(X) \mid \mathcal{A} \models G\}$$

Problem of axiomatizability:

For which structures $\mathcal{A}$ can one *axiomatize* $Th(\mathcal{A})$, that is, can one write down a formula F (or a recursively enumerable set F of formulas) such that

$$Th(\mathcal{A}) = \{G \mid F \models G\}?$$

Analogously for sets of structures.

Two Interesting Theories

Let $\Sigma_{Pres} = (\{0/0, s/1, +/2\},\ \emptyset)$ and $\mathbb{Z}_+ = (\mathbb{Z}, 0, s, +)$ its standard interpretation on the integers. $Th(\mathbb{Z}_+)$ is called *Presburger arithmetic* (M. Presburger, 1929). (There is no essential difference when one, instead of $\mathbb{Z}$, considers the natural numbers $\mathbb{N}$ as standard interpretation.)

Presburger arithmetic is decidable in 3EXPTIME (D. Oppen, JCSS, 16(3):323–332, 1978), and in 2EXPSPACE, using automata-theoretic methods (and there is a constant $c \geq 0$ such that $Th(\mathbb{Z}_+) \notin \text{NTIME}(2^{2^{cn}})$).

However, $\mathbb{N}_* = (\mathbb{N}, 0, s, +, *)$, the standard interpretation of $\Sigma_{PA} = (\{0/0, s/1, +/2, */2\}, \emptyset)$, has as theory the so-called *Peano arithmetic* which is undecidable, not even recursively enumerable.

Note: The choice of signature can make a big difference with regard to the computational complexity of theories.

3.4 Algorithmic Problems

Validity(F): $\models F$?

Satisfiability(F): F satisfiable?

Entailment(F,G): does F entail G?

Model(A,F): $A \models F$?

Solve(A,F): find an assignment β such that $A, \beta \models F$.

Solve(F): find a substitution σ such that $\models F\sigma$.

Abduce(F): find G with "certain properties" such that $G \models F$.

Gödel's Famous Theorems

1. For most signatures Σ, validity is undecidable for Σ-formulas. (One can easily encode Turing machines in most signatures.)
2. For each signature Σ, the set of valid Σ-formulas is recursively enumerable. (We will prove this by giving complete deduction systems.)
3. For $\Sigma = \Sigma_{PA}$ and $\mathbb{N}_* = (\mathbb{N}, 0, s, +, *)$, the theory $Th(\mathbb{N}_*)$ is not recursively enumerable.

These complexity results motivate the study of subclasses of formulas (*fragments*) of first-order logic

Q: Can you think of any fragments of first-order logic for which validity is decidable?

Some Decidable Fragments

Some decidable fragments:

- *Monadic class*: no function symbols, all predicates unary; validity is NEXPTIME-complete.
- Variable-free formulas without equality: satisfiability is NP-complete. (why?)
- Variable-free Horn clauses (clauses with at most one positive atom): entailment is decidable in linear time.
- Finite model checking is decidable in time polynomial in the size of the structure and the formula.

3.5 Normal Forms and Skolemization (Traditional)

Study of normal forms motivated by

- reduction of logical concepts,
- efficient data structures for theorem proving.

The main problem in first-order logic is the treatment of quantifiers. The subsequent normal form transformations are intended to eliminate many of them.

Prenex Normal Form

Prenex formulas have the form

$$Q_1 x_1 \dots Q_n x_n \; F,$$

where F is quantifier-free and $Q_i \in \{\forall, \exists\}$; we call $Q_1 x_1 \dots Q_n x_n$ the *quantifier prefix* and F the *matrix* of the formula.

Computing prenex normal form by the rewrite relation $\Rightarrow_P$:

$$\begin{array}{rcll} (F \leftrightarrow G) & \Rightarrow_P & (F \to G) \wedge (G \to F) & \\ \neg Q x F & \Rightarrow_P & \overline{Q} x \neg F & (\neg Q) \\ (Q x F \;\rho\; G) & \Rightarrow_P & Q y (F[y/x] \;\rho\; G),\ y \text{ fresh},\ \rho \in \{\wedge, \vee\} & \\ (Q x F \to G) & \Rightarrow_P & \overline{Q} y (F[y/x] \to G),\ y \text{ fresh} & \\ (F \;\rho\; Q x G) & \Rightarrow_P & Q y (F \;\rho\; G[y/x]),\ y \text{ fresh},\ \rho \in \{\wedge, \vee, \to\} & \end{array}$$

Here $\overline{Q}$ denotes the quantifier *dual* to Q, i. e., $\overline{\forall} = \exists$ and $\overline{\exists} = \forall$.

Skolemization

Intuition: replacement of $\exists y$ by a concrete choice function computing y from all the arguments y depends on.

Transformation $\Rightarrow_S$ (to be applied outermost, *not* in subformulas):

$$\forall x_1, \dots, x_n \exists y F \;\Rightarrow_S\; \forall x_1, \dots, x_n F[f(x_1, \dots, x_n)/y]$$

where f, where $\mathsf{arity}(f) = n$, is a new function symbol (*Skolem function*).

Together: $F \stackrel{*}{\Rightarrow}_P \underbrace{G}_{\text{prenex}} \stackrel{*}{\Rightarrow}_S \underbrace{H}_{\text{prenex, no } \exists}$

Theorem 3.9 *Let F, G, and H as defined above and closed. Then*