Chapter 1
Syntax and Semantics

Wer die Form zerstört, beschädigt auch den Inhalt. (Who destroys the form, also damages the content.) — Herbert von Karajan

In this chapter, we discuss the (abstract) syntax and semantics of formal languages, how to operate on the resulting syntactic phrases, and how to reason about the properties of these languages. We will in the later chapters apply these ideas to the language of first-order logic as the basis of mathematics, as well as to formal specification languages, and ultimately to programming languages.

1.1 Abstract Syntax

th To define formal languages like programming languages, we will use a variant of the well-known BNF (Backus-Naur Form) grammars. As an example, take the grammar

$$
\begin{array}{l}
E \in \mathit{Expression} \\
N \in \mathit{Numeral} \\
E \ ::= \ \mathsf{n}(N) \ \mid \ \mathsf{s}(E_1, E_2) \ \mid \ \mathsf{p}(E_1, E_2) \\
N \ ::= \ \mathsf{z}() \ \mid \ \mathsf{o}() \ \mid \ \mathsf{nz}(N) \ \mid \ \mathsf{no}(N)
\end{array}
$$

which introduces two syntactic domains *Expression* and *Numeral* with typed variables E and N denoting elements of these domains. The rule for domain *Expression* has three alternatives that start with symbols n, s, and p; the rule for domain *Numeral* has four alternatives starting with symbols z, o, nz, and no. From this grammar, the syntactic phrases o(), nz(o()), and no(o()) are elements of domain *Numeral* and the phrase s(n(o()), p(n(nz(o()))), n(no(o())))) is an element of domain *Expression*. The justification of these claims, based on the formal definition of grammars and their meaning, is given below.

Definition 1.1 (Abstract Syntax: Grammar). The grammar of an *abstract syntax* contains a sequence of $n \geq 1$ declarations of form

$$Var_i \in Domain_i$$

Each declaration i introduces a unique name $Domain_i$ for a new syntactic domain and a unique name Var_i of a variable that subsequently denotes elements of this domain; we call these variables also *nonterminals*.

Furthermore, for each declaration i, the grammar contains a rule of form

$$Var_i ::= Alternative_{i,1} \mid \ldots \mid Alternative_{i,n_i}$$

with $n_i \geq 1$ alternatives. Each $Alternative_{i,j}$ is a syntactic term

$$Symbol_{i,j}(V_{i,j,1}, \ldots, V_{i,j,m_{i,j}})$$

The term starts with a symbol $Symbol_{i,j}$ that is different from the symbol associated with any other alternative of the same rule; we call these symbols also *terminals*. Furthermore, the alternative contains $m_{i,j} \geq 0$ occurrences $V_{i,j,k}$ each of which denotes one of the nonterminals $Var_{i'}$. Multiple occurrences of the same nonterminal in an alternative may receive different subscripts for easier reference.

The elements of the domains *Expression* and *Numeral* are syntactic phrases which we will call "expressions" and "numerals". These phrases are constructed according to the following definition.

Definition 1.2 (Abstract Syntax: Language). Every syntactic domain $Domain_i$ introduced by the grammar of an abstract syntax denotes a set of syntactic phrases. The first domain introduced by the grammar, is considered as the *language* of the grammar. Each $Domain_i$ is defined as that set of phrases such that p is in $Domain_i$ if and only if

- p can be derived from Var_i by a sequence of substitutions of nonterminals according to the rules of the grammar, but
- p itself does not contain a nonterminal any more.

We write the first requirement as $Var_i \rightarrow^* p$, and formalize it as follows:

- Let $p \rightarrow_i p'$ denote that phrase p' is identical to phrase p except that some occurrence of nonterminal Var_i in p has been substituted in p' by one of the alternatives $Alternative_{i,j}$ from the grammar rule for Var_i.
- Let $p \rightarrow p'$ denote that $p \rightarrow_i p'$ holds for some $0 \leq i < n$, i.e., p' equals p except that some nonterminal in p has been substituted.
- Let $p \rightarrow^* p'$ denote that there exists a sequence of $m \geq 0$ phrases $p_0, \ldots, p_m$ such that $p = p_0$ and $p_m = p'$ and $p_0 \rightarrow p_1$, $p_1 \rightarrow p_2$, $\ldots$, $p_{m-1} \rightarrow p_m$. In other words, p' is derived from p by a sequence of substitutions of nonterminals.

Thus a phrase p without nonterminals is in $Domain_i$ if and only if $Var_i \rightarrow^* p$ holds.

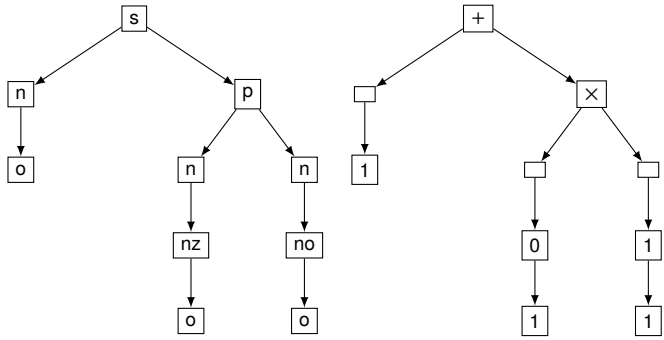

Fig. 1.1 Abstract Syntax Trees

For the grammar given on page 3, we thus have the relations

$$E \longrightarrow \mathsf{s}(E, E) \longrightarrow \mathsf{s}(E, \mathsf{p}(E, E)) \longrightarrow^* \mathsf{s}(\mathsf{n}(N), \mathsf{p}(\mathsf{n}(N), \mathsf{n}(N)))$$
$$N \longrightarrow \mathsf{o}()$$
$$N \longrightarrow \mathsf{nz}(N) \longrightarrow \mathsf{nz}(\mathsf{o}())$$
$$N \longrightarrow \mathsf{no}(N) \longrightarrow \mathsf{no}(\mathsf{o}())$$

and therefore

$$E \longrightarrow^* \mathsf{s}(\mathsf{n}(N), \mathsf{p}(\mathsf{n}(N), \mathsf{n}(N))) \longrightarrow^* \mathsf{s}(\mathsf{n}(\mathsf{o}()), \mathsf{p}(\mathsf{n}(\mathsf{nz}(\mathsf{o}())), \mathsf{n}(\mathsf{no}(\mathsf{o}()))))$$

Consequently the syntactic phrase $\mathsf{s}(\mathsf{n}(\mathsf{o}()), \mathsf{p}(\mathsf{n}(\mathsf{nz}(\mathsf{o}())), \mathsf{n}(\mathsf{no}(\mathsf{o}()))))$ is an element of domain *Expression*. The left part of Figure 1.1 depicts this phrase as an *abstract syntax tree*; every node represents the root of a phrase with the symbol associated to the phrase labeling the node.

In practice, our abstract syntax definitions will not rigorously stick to the standard notation where in each alternative a terminal always precedes the subphrases; it may also occur among or after the subphrases and multiple nonterminals may be used to separate the subphrases. Furthermore, terminals need not be unique or may be dropped at all, if the number/types of the subphrases uniquely determine the corresponding alternative. The grammar on page 3 is therefore typically written in the more readable form of

$$E \in \mathit{Expression}$$
$$N \in \mathit{Numeral}$$
$$E \ ::= \ N \ \mid \ E_1 + E_2 \ \mid \ E_1 \times E_2$$
$$N \ ::= \ 0 \ \mid \ 1 \ \mid \ N0 \ \mid \ N1$$

The expression $1 + (10 \times 11)$ of the new language (we use parentheses to clarify its structure) matches the expression $\mathsf{s}(\mathsf{n}(\mathsf{o}()), \mathsf{p}(\mathsf{n}(\mathsf{nz}(\mathsf{o}())), \mathsf{n}(\mathsf{no}(\mathsf{o}()))))$ of the original one. As depicted in Figure 1.2, despite of the different linear representations of both expressions, their syntax trees correspond to each other node by node.

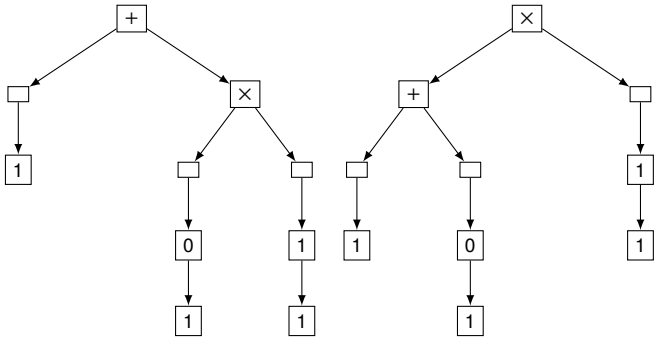

Fig. 1.2 The Abstract Syntax Trees for 1 + (10 × 11) and (1 + 10) × 11

The syntax described by these grammars is "abstract" rather than "concrete" because it describes trees rather than strings of symbols. We are not concerned that a string like $1 + 10 \times 11$ can be parsed in two different ways (giving rise to the two different trees depicted in Figure 1.2) because we will in the following only deal with abstract syntax trees. We will use linear notation only to describe such trees in a convenient form; if necessary, we will use parentheses as in $1 + (10 \times 11)$ or $(1 + 10) \times 11$ to make the intended tree clear.

1.2 Structural Induction

We may prove properties of syntactic domains by a particular proof principle.

Definition 1.3 (Structural Induction). Let $F[p]$ be a statement about p and assume that we want to prove a statement of form

For every phrase p in domain *Domain*, $F[p]$.

Then it suffices to perform, for every alternative *Alternative* in the grammar rule for *Domain*, a separate subproof where

- under the assumption that $F[Var]$ holds for every occurrence of a nonterminal *Var* from *Domain* in the alternative,
- it is proved that $F[Alternative]$ holds.

Example 1.1. We want to prove for the language on page 5 the (trivial) statement

For every numeral n in *Numeral*, n does not contain the symbol 2.

According to the four alternatives in the grammar rule for domain *Numeral*, we have to prove