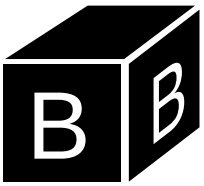


Chapter 2

Languages and Grammars

- Grammars
- Regular expressions
- Finite state machines

Techniques for describing a language's syntax

The four parts of a grammar

- *N*, a nonterminal alphabet
- *T*, a terminal alphabet
- *P*, a set of rules of production
- *S*, the start symbol, an element of *N*

N = {Identifier, Letter, Digit}
T = {a, b, c, 1, 2, 3}
P = the productions
1. Identifier → Letter
2. Identifier → Identifier Letter
3. Identifier → Identifier Digit
4. Letter → a
5. Letter → b
6. Letter → c
7. Digit → 1
8. Digit → 2
9. Digit → 3

S = Identifier

Figure 2.1
Grammar A for Component Pascal identifiers.

N = {I, F, M}
T = {+, -, d}
P = the productions
1. I → F M
2. F → +
3. F → -
4. F → ε
5. M → d M
6. M → d

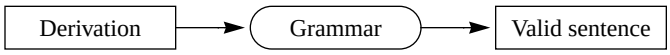
S = I

Figure 2.2
Grammar B for signed integers.

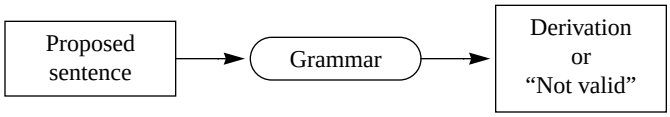
N = {A, B, C}
T = {a, b, c}
P = the productions
1. A → a A B C
2. A → a b C
3. C B → B C
4. b B → b b
5. b C → b c
6. c C → c c

S = A

Figure 2.3
Grammar C, a context sensitive grammar.



(a) Deriving a valid sentence.



(b) The parsing problem.

Figure 2.4
The difference between deriving an arbitrary sentence and parsing a proposed sentence.

N = {E, T, F}
T = {+, ×, (,), a}
P = the productions
1. E → E + T
2. E → T
3. T → T × F
4. T → F
5. F → (E)
6. F → a

S = E

Figure 2.5
Grammar D, a grammar for algebraic expressions.

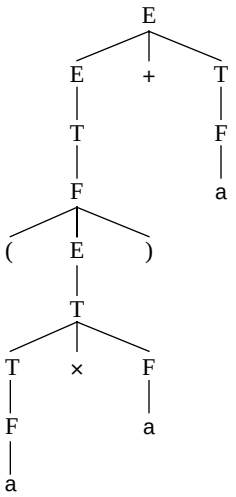


Figure 2.6
The syntax tree for the parse of (a × a) + a in Grammar D.

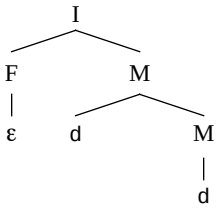


Figure 2.7
The syntax tree for the parse of dd in Grammar B.

- Alternation—select one of several alternatives
- Optional—include zero or one time
- Repetition—include zero or more times

The three operations of EBNF

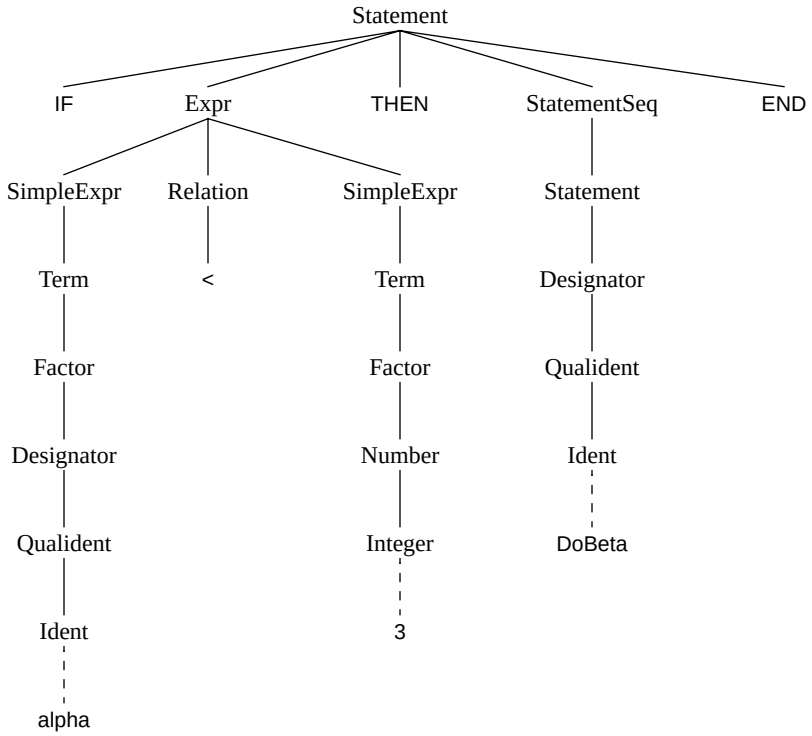


Figure 2.8
The syntax tree for the derivation in Example 2.13.