Grammars

By Dr. Fathy

Grammars

- Describes underlying rules (syntax) of programming languages
  - Compilers (parsers) are based on such descriptions
- More expressive than regular expressions/finite automata
- Context-free grammar (CFG) or just *grammar*

Definition

A grammar is a 4-tuple G = (V,T,P,S)

- V: set of variables or nonterminals
- T: set of terminal symbols (terminals)
- P: set of productions
  - Each production: **head** $\rightarrow$ **body**, where **head** is a variable, and **body** is a string of zero or more terminals and variables
- S: a start symbol from V

Example 1: Assignment statements

- V = { S, E }, T = { i, =, +, *, n }
- Productions:

  S → i = E

  E → n

  E → i

  E → E + E

  E → E * E

This grammar represents strings such as:

i = n + n

i = n * i + i + n * n

Example 2: Palindromes

- V = { S }, T = { a, b }
- Productions:

  $S \rightarrow \varepsilon$
  $S \rightarrow a$
  $S \rightarrow b$
  $S \rightarrow aSa$
  $S \rightarrow bSb$

This grammar represents strings such as:
a
baab
babab
abaaabaaaba
ε

Example 3: 0^n1^n

- $V = \{ S \}$, $T = \{ 0, 1 \}$
- Productions:
  $S \rightarrow \varepsilon$
  $S \rightarrow 0S1$

This grammar represents strings such as:
0011
000111
01
ε

Derivation of strings

- Applying productions: replace variable (head) with corresponding string (body)
- Example: to derive i = n + n in for the assignment statement grammar:
  $S \Rightarrow i = E \Rightarrow i = E + E$
  $\Rightarrow i = E + n \Rightarrow i = n + n$
- Above derivation was carried out through four applications of the productions:
  $S \rightarrow i = E,\ E \rightarrow E + E,\ E \rightarrow n,\ E \rightarrow n$

L(G): Language of a grammar

- Definition: Given a grammar G, and a string w over the alphabet T, $S \Rightarrow^{*}_{G} w$ if there is a sequence of productions that derive w
- L(G) = { w in T* | $S \Rightarrow^{*}_{G} w$ }, the language of the grammar G

Leftmost vs rightmost derivations

- Leftmost derivation: the leftmost variable is always the one replaced when applying a production
  - Example: $S \Rightarrow i = E \Rightarrow i = E + E$
    $\Rightarrow i = n + E \Rightarrow i = n + n$
- Rightmost derivation: rightmost variable is replaced
  - Example: $S \Rightarrow i = E \Rightarrow i = E + E$
    $\Rightarrow i = E + n \Rightarrow i = n + n$

Context-free languages

- A language generated by a grammar is called a **context-free language**
- The set of regular languages is a subset of the set of context-free languages
  - Proof?
- Some languages are context-free, but not regular
  - e.g., palindromes (proven not regular through the pumping lemma)

Definition: Context-Free Grammars

Grammar $G = (V, T, S, P)$

Variables

Terminal symbols

Start variable

Productions of the form:

$$A \rightarrow x$$

Non-Terminal

String of variables and terminals

Derivation Tree of A Context-free Grammar

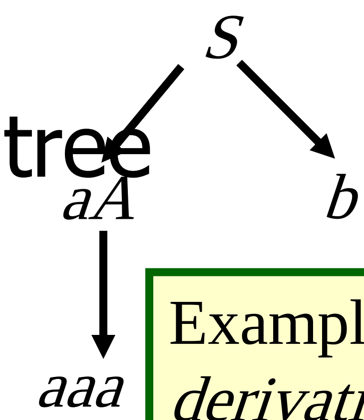
- Represents the language using an ordered rooted tree.

- Root represents the starting symbol.
- Internal vertices represent the nonterminal symbol that arise in the production.
- Leaves represent the terminal symbols.

- If the production $A \rightarrow w$ arise in the derivation, where w is a word, the vertex that represents A has as children vertices that represent each symbol in w, in order from left to right.

Language Generated by a Grammar

- Example: Let $G = (\{S,A,a,b\},\{a,b\}, S, \{S \rightarrow aA, S \rightarrow b, A \rightarrow aa\})$. What is $L(G)$?
- Easy: We can just draw a tree of all possible derivations.
  - We have: $S \Rightarrow aA \Rightarrow aaa$.
  - and $S \Rightarrow b$.
- Answer: $L = \{aaa, b\}$.



Example of a *derivation tree* or *parse tree* or *sentence diagram.*

Example: Derivation Tree

- Let G be a context-free grammar with the productions $P = \{S \rightarrow aAB, A \rightarrow Bba, B \rightarrow bB, B \rightarrow c\}$. The word w = acbabc can be derived from S as follows:

$$S \Rightarrow aAB \rightarrow a(Bba)B \Rightarrow acbaB \Rightarrow acba(bB) \Rightarrow acbabc$$

Thus, the derivation tree is given as follows: