Chapter 12: Context-Free Grammars *

Peter Cappello
Department of Computer Science
University of California, Santa Barbara
Santa Barbara, CA 93106
cappello@cs.ucsb.edu

- The corresponding textbook chapter should be read before attending this lecture.
- These notes are not intended to be complete. They are supplemented with figures, and other material that arises during the lecture period in response to questions.

*Based on **Theory of Computing**, 2nd Ed., D. Cohen, John Wiley & Sons, Inc.

Symbolism for Generative Grammars

- The book chapter gives a good explanation of the background and reason for studying this material.
- A **generative grammar** is a grammar with which one can *generate* all the words (sentences) in a language.

Definition A context-free grammar (CFG) is a collection of 3 things:

- An alphabet Σ of letters called *terminals*.
- A set of symbols called *nonterminals*, 1 of which is the symbol S, the "start" symbol.
- A finite set of productions of the form

  $$Nonterminal \ \rightarrow \ (terminals + nonterminals)^*$$

  At least 1 production has S as its left side.

By convention, we use:

- lower case letters and special symbols for terminals
- upper case letters for nonterminals.

Definition The **language generated** by a CFG is the set of all strings of terminals produced from S using productions as substitutions.

A language generated by a CFG is a **context-free language** (CFL).

Example

Production 1 $S \to aS$

Production 2 $S \to \Lambda$

Applying production 1 two times, followed by production 2, yields:

$$\begin{aligned} S &\Rightarrow aS \\ &\Rightarrow aaS \\ &\Rightarrow aa\Lambda \\ &= aa \end{aligned}$$

- The language generated by this CFG is a^*.
- $\to$ means "can be replaced by".
- $\Rightarrow$ (used in a derivation) means "can develop into".
- A derivation's right hand side (RHS) is a working string when it contains nonterminals.

Example

Define a CFG that accepts $(a+b)^*$.

$S \rightarrow aS$

$S \rightarrow bS$

$S \rightarrow \Lambda$

Example

Define a CFG that accepts $(a+b)^*aa(a+b)^*$.

$S \rightarrow XaaX$

$X \rightarrow aX$

$X \rightarrow bX$

$X \rightarrow \Lambda$

Give a derivation of $ababaaaba$.

Example

Define a CFG that accepts $\{a^n b^n\}$.

$S \to aSb$

$S \to \Lambda$

Equivalently:

$S \to aSb \mid \Lambda$

- Give a derivation of $aaabbb$.
- Define a CFG that accepts palindromes over $\{a, b\}$. (It should include strings such as aba.)

Trees

Given the CFG

$S \rightarrow AA$

$A \rightarrow AAA|bA|Ab|a$

- Derive the word *bbaaaab*: $S \Rightarrow AA \Rightarrow bAA \Rightarrow bbAA \Rightarrow bbaA \Rightarrow bbaAAA \Rightarrow bbaaAA \Rightarrow bbaaaA \Rightarrow bbaaaAb \Rightarrow bbaaaab$
- Draw the tree corresponding to this derivation.
- Such a tree is called a syntax tree or parse tree.

Lukasiewicz Notation

- Consider the CFG:

  $S \rightarrow +|*|n$

  $+ \rightarrow + \ + | + \ * | + \ n | * \ + | * \ * | * \ n | n \ + | n \ * | n \ n$

  $* \rightarrow + \ + | + \ * | + \ n | * \ + | * \ * | * \ n | n \ + | n \ * | n \ n$

  $n \rightarrow 1|2|3|4|5|6|7|8|9$

- One possible derivation is $S \Rightarrow + \Rightarrow 3 * \Rightarrow 3\ 4\ 5$.
- Write the parse tree for this is on the board.
- From the parse tree construct the prefix notation by walking around the tree, writing down the symbols in the order in which they are first visited (excluding S): $+\ 3\ *\ 4\ 5$.

- Think of these items as having been pushed on a stack in the order of visitation.
- To evaluate the expression, when the top 2 items are numbers:

  1. pop the top 3 items
  2. evaluate that expression (e.g., + 35 evaluates to 8)
  3. push the resulting value on the stack

  Continue until the stack contains only 1 number.
- Do this for our string of + 3 $*$ 4 5.
- Do this for the following tree $(((1 + 2) * (3 + 4)) + 5) * 6$. (Its value should be 156.)

Ambiguity

Definition: A CFG G is **ambiguous** if there exists a $w \in L(G)$ with 2 derivations that correspond to different parse trees.

If a CFG is not ambiguous it is **unambiguous**.

Example Let CFG G be $S \rightarrow aS|Sa|a$, the regular language aa^*.

The word aa has 2 derivations:

- $S \Rightarrow Sa \Rightarrow aa$
- $S \Rightarrow aS \Rightarrow aa$

distinct with a distinct parse tree.

However a^+ can be defined by $S \rightarrow aS|a$, which is not ambiguous.

The Total Language Tree

Definition: For a given CFG, define a tree:

- Its root is S
- For each nonterminal node in the tree,
  For each nonterminal, N at that node, construct a child node in the tree for each production with N as the LHS.

Example Let CFG G be:

$S \rightarrow aa|bX|aXX$

$X \rightarrow ab|b$

- What is the total language tree for this CFG?
- What is the total language tree for $S \rightarrow \Lambda|aSb$?