

Lecture 13: Context-free Grammars.

David Dill

Department of Computer Science

1

Context-Free Languages

Regular languages have many wonderful properties, but not all languages are regular.

Next, we'll study a more powerful class of languages, the *context-free languages*.

Context-free languages were identified in the 1950's by linguist Noam Chomsky, as a natural place in a hierarchy of languages, which included the regular languages.

3

Outline

- Context-free grammars
- Derivations and parse trees
- ambiguity
- CF Pumping Lemma

2

Formal Definition of Context-Free Grammars

A context-free grammar (CFG) is a 4-tuple (V, T, P, S)

- V is a finite set of *non-terminal symbols*,
- T is a finite set of *variables* (AKA *terminal symbols*),
- $P \subseteq V \times (V \cup T)^*$ is a finite set of *productions*, and
- $S \in V$ is the *Sentence symbol*

V and T must be disjoint sets.

Productions are written $A \rightarrow aBc$. A is the left-hand side (LHS), also called the *head*, and aBc is the right-hand side (RHS), also called the *body*.

4

Notation

$a, b, c, 0, 1, \dots$ are *terminals*

A, B, S, \dots are *non-terminals*

w, x, y, z are *terminal strings* (T^*)

α, β, γ are *strings of terminals and/or non-terminals* ($(V \cup T)^*$).

Several productions with common heads can be combined: $A \rightarrow a \mid Aa \mid bAb$

5

Example: Grammar for Regular Expressions

$\Sigma = \{a, b\}$.

$R \rightarrow \emptyset$

$R \rightarrow e$

$R \rightarrow a$

$R \rightarrow b$

$R \rightarrow R + R$

$R \rightarrow R \cdot R$

$R \rightarrow R^*$

$R \rightarrow (R)$

$(a + b \cdot a)^*$ is a regular expression because

$R \Rightarrow R^* \Rightarrow (R)^* \Rightarrow (R + R)^* \Rightarrow (R + R \cdot R)^* \Rightarrow$

$(a + R \cdot R)^* \Rightarrow (a + R \cdot a)^* \Rightarrow (a + b \cdot a)^*$

7

Derivations

The language of a given CFG, $G = (V, T, P, S)$, can be characterized using the concept of a *derivation*.

Def Derivation step: $\alpha A \beta \Rightarrow \alpha \gamma \beta$ whenever $A \rightarrow \gamma$ is in P .

$\alpha \xRightarrow{*} \beta$ if we can get from α to β in zero or more steps.

The language of G ($L(G)$) is $\{w \in T^* \mid S \xRightarrow{*} w\}$

A language is context-free if it is $L(G)$ for some CFG.

6

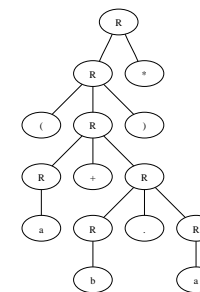
Parse Trees

A *parse tree* is a tree that shows how to derive a string from a non-terminal.

The children of a node in the tree correspond to the body of a production that has the node as head.

For $A \rightarrow \epsilon$, there is a single child, ϵ .

Parse tree for $(a + b \cdot a)^*$:



8

Yield of a Parse Tree

The concatenation of the symbols (or ϵ) at the leaves of a parse tree is called the *yield* of the parse tree.

The yield can always be derived from the symbol at the root of the tree.

If the root is S and the yield is $x \in T^*$, then x is in $L(G)$.

9

Ambiguity

A CFG is *ambiguous* if there is more than one leftmost derivation for the same string.

Equivalently: more than one parse tree for the same string.

Ambiguity often causes problems:

- With interpretation.
- With parsing.

Is our grammar for regular languages ambiguous?

11

Leftmost Derivations

There are many ways to extract a derivation from a parse tree.

If we put a restriction on how the derivation is done, we can get the derivation uniquely.

Leftmost derivation step is one where each step replaces the *leftmost non-terminal in the sentential form*

$$R \Longrightarrow R^* \Longrightarrow (R)^* \Longrightarrow (R + R)^* \Longrightarrow (a + R)^* \Longrightarrow (a + R \cdot R)^* \Longrightarrow (a + b \cdot R)^* \Longrightarrow (a + b \cdot a)^*$$

10

Inherently Ambiguous CFLs

Some languages have context-free grammars, but no unambiguous CFGs.

$$\{a^i b^j c^k \mid i = j \vee j = k\}$$

Intuition (NOT proof). This is the union of two unambiguous grammars, but they overlap when $i = j = k$.

$$\begin{aligned} S &\rightarrow AC \\ A &\rightarrow aAb \mid \epsilon \\ C &\rightarrow Cc \mid \epsilon \\ S &\rightarrow DE \\ D &\rightarrow aD \mid \epsilon \\ E &\rightarrow bEc \mid \epsilon \end{aligned}$$

12

CF Pumping Lemma

Theorem: For every CFL L , there exists a positive integer n such that for every $z \in L$ such that $|z| \geq n$ there exists $uvwxy = z$ such that $|vwx| \leq n$ and $vx \neq \epsilon$ such that $uv^kwx^ky \in L$ for all $k \geq 0$.

Game:

CFL L is the "game board"

Player 1 picks n

Player 2 picks $z \in L$ such that $|z| \geq n$

Player 1 picks $uvwxy = z$ such that $|vwx| \leq n$ and $vx \neq \epsilon$.

Player 2 picks k

Player 1 wins if $uv^kwx^ky \in L$.

The CFPL says L is CF, Player 1 always has a winning strategy.

13

CFPL proof sketch

Unlike the PL for regular language, which is based on DFAs, the CFPL is based on context-free grammars (not automata).

Proof sketch:

Claim: A parse tree of height h cannot produce a string of length greater than m^h , where m is the length of the longest RHS of any production in G .

Let $n = m^{|V|+1}$, and z be any string in T^* that is of length n or great. Let τ be parse tree with the fewest nodes for z . τ must have height $|V| + 1$ or greater.

There is a path from the root to a leaf in this tree with a repeated non-terminal symbol. Let A be the repeated nonterminal that is closest to the leaf of this path.

Consider the lowest and second lowest occurrences of A on that path. Let w be the yield of the lowest A , vwx be the yield of the second lowest A , and let u and y be the substrings of z before and after vwx .

15

Application of the CFPL

Theorem The language L over $\Sigma = \{0, 1, 2\}$ where every string has equal numbers of 0s, 1s, and 2s is not a CFL.

proof: We prove by contradiction. Suppose L were context-free, so that the CF pumping lemma holds.

Let n be the pumping constant. Let $z = 0^n1^n2^n$.

By the CFPL, $z = uvwxy$ s.t. $|vwx| \leq n$, $vx \neq \epsilon$, and $uv^kwx^ky \in L$ for all $k \geq 0$. vwx must be of the form 0^*1^* or 1^*2^* because it's not long enough to cross from the 0's to the 2's.

Suppose vwx is of the form 0^*1^* . $vx \neq \epsilon$ so it has at least one 0 or 1 (or both 0s and 1s, of course).

If vwx is 1^*2^* , the proof is similar.

Hence, uv^kwx^ky when $k \neq 1$ will change the number 0s and 1s but not the number of 2s, so it will not be in L , contradicting the CFPL. Hence, L is not context-free. \square

(Intuition: $\{0^n\}$ is regular, $\{0^n1^n\}$ is CF but not regular, $\{0^n1^n2^n\}$ is not CF.)

14

CFPL proof, cont.

The string can be pumped by repeating A zero or more times, as shown in $\langle\langle$ figure in book $\rangle\rangle$, so $uv^kwx^ky \in L$ for all $k \geq 0$.

$|vwx| \leq n$ because it's sub-parse tree is of height no more than $|V| + 1$ (there is only one repeated nonterminal).

Suppose $vx = \epsilon$. Then then the second lowest A yields $vwx = w$, so we could just replace it with the lowest A , reducing the number of nodes in the tree, contradicting our choice of the tree with fewest nodes, so $vx \neq \epsilon$

16