

# CS310

---

## Context Free Languages and Grammars Sections:2.1 page 99

October 6, 2010

# Context Free Grammar

- Another way to represent a language
  - Can represent more languages than FA
- Produces a “Context Free Language”
- Pushdown Automata: machine that recognizes a context free language
- Trivia:
  - First used to describe human languages
  - Now used to parse computer languages (C, C++)

# Context Free Grammar

- Example

$$A \rightarrow 0A1$$

---

$$A \rightarrow B$$

$$B \rightarrow \#$$

Variables: A, B (may appear on LHS and RHS)

Terminals: 0, 1, # (only appear on the RHS)

Start variable: Variable on LHS of top rule

Language:

Example:

# Example

- $A \rightarrow$   $\rightarrow 00\#11$

- derivation

- write  $u \rightarrow^* v$  if there is a derivation of the string  $v$  from  $u$  using the grammar, where  $u$  and  $v$  are strings of terminals and variables
  - $0A1 \rightarrow^* 00\#11$
  - Parse Tree

# Exercise

$R \rightarrow XRX \mid S$

$S \rightarrow aTb \mid bTa$

$T \rightarrow XTX \mid X \mid \epsilon$

$X \rightarrow a \mid b$

Variables, terminals of G?

Start variable?

- True or false?  $T \xrightarrow{*} aba$

# Formal Definition

- A context free grammar (CFG)  $G$  is a 4-tuple  $(V, \Sigma, R, S)$ 
  - $V$  finite set of variables
  - $\Sigma$  finite set of terminals
  - $R$  set of rules of form:
    - variable (string of variables and terminals)
  - $S \in V$ , start variable
  - The language of the grammar is:
  - $L(G) = \{ w \in \Sigma^* \mid S \xrightarrow{*} w\}$ 
    - what?

# Example

- $L = \{ w \in \{a, b\}^* \mid aa \text{ is a substring} \}$

Find a grammar that generates this language

---

- Can we write this as a regular expression?

# Constructing a CFG from a Language, L

- Requires some thought and creativity, just like building a Finite Automata
- Hints:
  - If possible, break L into pieces  $L = L_1 \cup L_2$ 
    - Create grammar for  $L_1$  and  $L_2$ ,  $S \rightarrow S_{L_1} \mid S_{L_2}$
  - If L is regular, use regular expression as guide
  - If L is regular, construct DFA then construct CFG:
    - Make variable  $R_i$  for each state  $q_i$  in DFA
    - Add rule  $R_i \rightarrow \epsilon$  for all  $q_i \in F$ ,  $R_i \rightarrow aR_z$  if  $\delta(q_i, a) = q_z$
    - $R_0$  is start where  $q_0$  is start of DFA

# Example

- Grammar  $G_2$  on page 101
- Show derivation for “a boy sees a flower”
  - Notice how this statement is non-creepy?
- Show the parse tree

# Write the Grammar

$$\Sigma = \{0,1\}$$

- $\{w \mid w \text{ is a binary number greater than } 4\}$
- $\{w \mid w \text{ is } 1^n0^n, n \geq 0\}$       ?  $n \geq 1$
- $\{w \mid w \text{ is } 1^n0^n, n \geq 0, n \text{ is even}\}$
- $\{w \mid w \text{ contains at least three } 1\text{s}\}$
- $\{w \mid w \text{ contains more } 1\text{s than } 0\text{s}\}$
- $\{w \mid |w| \text{ is prime}\}$
- $\{a^i b^j c^k \mid i=j \text{ or } i=k\}$   $\Sigma = \{a, b, c\}$
- $\{w \mid w \text{ is a string of matched } ()\}$      $\Sigma = \{(,), (\,\,\,)\}$

# Ambiguous Grammar

- $E \rightarrow E + E \mid E \times E \mid E \mid a$
  - Find parse tree for:  $a + a \times a$
-

This window is titled "JFLAP : <untitled34>". It has tabs for Editor, Brute Parser, and Table Text Size. The Editor tab is selected. It contains a "Table Text Size" slider and a table with columns for LHS and RHS. The table entries are:

LHS	RHS
S	$\rightarrow A$
A	$\rightarrow 0A1$
A	$\rightarrow B$
B	$\rightarrow \#$

Below the table is a tree diagram with root node S, which branches into A and A, further branching into 0 and 1 respectively.

Derived 0A1 from A.

This window is titled "JFLAP : <untitled34>". It has tabs for File, Input, Test, Convert, and Help. The Brute Parser tab is selected. It contains a "Table Text Size" slider and buttons for Start, Pause, Step, and Noninverted Tree. The Step button is currently active. The input field shows "Input 0#1" and the message "String accepted! 6 nodes generated." Below this is another table with the same grammar rules as the Editor window. To the right is a tree diagram with root node S, which branches into A and A, further branching into 0 and 1 respectively.

Derived 0A1 from A.

# JFLAP

JFLAP appears to want the start symbol to be S

# More examples

$\Sigma = \{0,1\}$

- $\{w \mid |w| \text{ is odd, middle character is } 0\}$
- $\{w \mid w = xyx, x \in \Sigma, y \in \Sigma^*\}$
- $\{w \mid w = w^R\}$
- complement of  $\{w \mid w = 0^n 1^n, n \geq 1\}$
- $\{w\#x \mid w^R \text{ is a substring of } x; w, x \in \Sigma^*\}$
- $\{w \mid w = 0^{n+m} 1^n, n \geq 1, m \geq 1\}$
- $\{w \mid w \text{ contains at least as many } 0\text{s as } 1\text{s}\}$
- $\{w \mid w = 0^{2n} 1^n, n \geq 1\}$
- $\{w \mid w \text{ contains twice as many } 0\text{s as } 1\text{s}\}$