COMS W3261 – Lecture 6, Part 1:

CFG review and Chomsky Normal Form

Teaser: Is the grammar

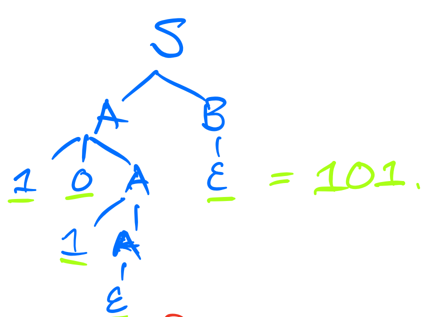
abbreviates two rules

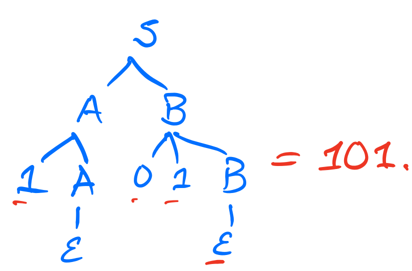
(V, Σ, R, S)

$$S \to AB \mid BA$$
$$A \to 10A \mid 1A \mid \varepsilon$$
$$B \to 01B \mid \varepsilon$$

ambiguous?

we say that a string is derived ambiguously if it has two or more parse trees, or equivalently, if it has two leftmost derivations.





Two parse trees for $s = 101 \implies s$ is ambiguously derived.
A grammar is ambiguous if at least one string is derived ambiguously.

YES

Leftmost derivation: a derivation (sequence of replacements) in which we always replace the leftmost variable.

$$S \Rightarrow AB \Rightarrow 10AB \Rightarrow 101AB \Rightarrow 101B \Rightarrow 101.$$

$$S \stackrel{*}{\Rightarrow} 101.$$

Announcements:
- HW #3 due Monday, 7/19/21 @ 11:59 PM EST
- HW #4 due Monday, 7/26/21 —— " ——
- HW #1 and #2 published, solutions up.
- Latex tutorials: Always OK in OH; also see Ed.
- Moving Thursday PM virtual OH → 5:30–7:00 PM EST.
- Reminder: write up your HW solutions yourself.

Today:
1. Review, focus on CFGs.
2. Chomsky Normal Form.
3. Pushdown Automata — automata with "stack memory"

CFG Review.

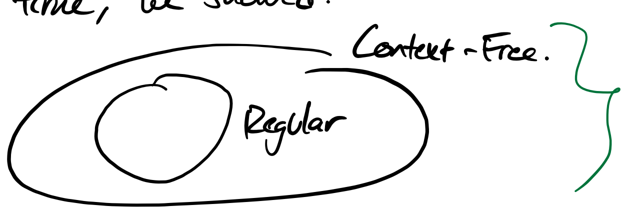
– A context-free grammar is a set of substitution rules that turn single variables into strings of variables and terminals.

– A string is derived ($\stackrel{*}{\Rightarrow}$) from another string if it can be obtained by repeated substitutions.

– The set of all strings derived from the start variable is the language of the grammar.

– Languages described by some CFG are called context-free.

– Last time, we showed:



Why? Context-Free $\not\subseteq$ Regular, because nonregular languages like $\{0^n 1^n \mid n \geq 0\}$ are context-free.

$=$ Regular $\subseteq$ Context-free Languages, because any DFA can be turned into a CFG.

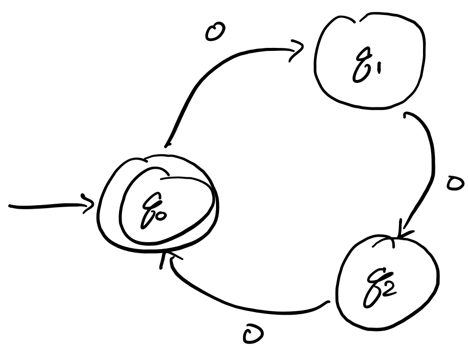
We showed how to convert DFAs to CFGs:

Idea: simulated the execution of the DFA on a string with substitution rules.

To convert DFA $\to$ CFG:

- $\star$ Create rules $R_i \to a R_j$ for each transition $\delta(q_i, a) = q_j$.
- $\star$ Create rules $R_i \to \varepsilon$ for each accept state q_i

Example: Convert $\{w \mid w$ is all strings in $\{0\}^*$ such that $|w|$ is divisible by $3.\}$



$$R_0 \to 0R_1 \mid \varepsilon$$
$$R_1 \to 0R_2$$
$$R_2 \to 0R_3$$

$$R_0 \to 0R_1 \to 00R_2 \to 000R_0 \to 000$$
$$R_0 \to \varepsilon$$
$$R_0 \overset{*}{\Rightarrow} 000000$$

Example. Consider $G = (V, \Sigma, R, S)$, where

$V = \{S, A, B\}$, $\Sigma = \{1, 2, +, =\}$, and R:

$$S \to 1A2$$
$$A \to 1A1 \mid B$$
$$B \to +1=$$

What language does this grammar describe?

- Tactic 1: derive some strings.

$$S \Rightarrow 1A2 \Rightarrow 11A12 \Rightarrow 11B12 \Rightarrow 11+1=12.$$

$$S \Rightarrow 1A2 \Rightarrow 1B2 \Rightarrow 1+1=2.$$

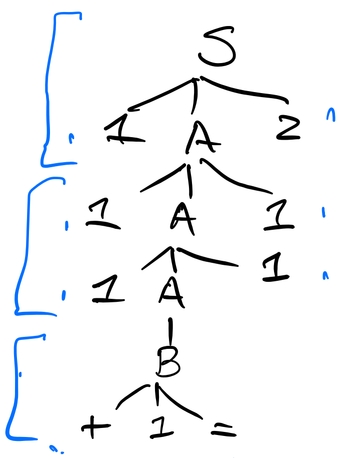
$$S \Rightarrow 1A2 \overset{*}{\Rightarrow} 1111A1112 \Rightarrow 1111B1112 \Rightarrow 1111+1=1112.$$

- Tactic 2: simplify the grammar.

Observe that B goes only to terminal symbols.

$$S \to 1A2$$
$$A \to 1A1 \mid +1=$$

- Tactic 3: draw some parse trees.



$11\,1+1=112$

not a power operation
1 repeated (n-1) times.

$$\{1^n+1=1^{n-1}2 \mid n \geq 0\}$$

Design trick. How to build a CFG for $L_1 \cup L_2$?

Suppose we have

G_1: a grammar for L_1:

$S_1 \to \dots \mid \dots$

G_2: a grammar for L_2.

$S_2 \to ABc \dots \mid \dots$

Claim: if we add the new start rule $S \to S_1 \mid S_2$, we get a grammar that recognizes $L_1 \cup L_2$.

(Could write this formally:)

$$G_1 = (V_1, \Sigma_1, R_1, S_1)$$

$$G_2 = (V_2, \Sigma_2, R_2, S_2)$$

Then $G_3 = (V_1 \cup V_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, R_1 \cup R_2 \cup \{S \to S_1 \mid S_2\}, S)$ recognizes $L_1 \cup L_2$.

(There are tricks for $*$, $\circ$, and others...)

2. Chomsky Normal Form

Example. Two CFGs that generate the same language.

$$S \to 0S \mid A$$
$$A \to A1 \mid \varepsilon$$

$$S \to S1 \mid A$$
$$A \to 0A \mid \varepsilon$$

$S \overset{*}{\Rightarrow} 0000S \Rightarrow 0000A \overset{*}{\Rightarrow} 0000A111 \Rightarrow 0000111$

$S \overset{*}{\Rightarrow} S111 \Rightarrow A111 \Rightarrow 0000A111 \Rightarrow 0000111$

How do we tell if two grammars recognize the same language?

Define a standard "normal form."

(Example. $\frac{24}{18} \overset{?}{=} \frac{8}{6}$. $\Longrightarrow$ both equal to $4/3$.)

Def. A context-free grammar is in Chomsky normal form if every rule has the form $A \to BC$ or

$A \to a$,

where A, B, and C are variables and a is a terminal.

$$S \to AB \mid \varepsilon \mid 0 \mid 1 \mid BB \mid AA$$
$$A \to AA \mid 0$$
$$B \to BB \mid 1.$$

(Can also have the special rule $S \to \varepsilon$.)

(Additionally – B, C can't be the start variable.)

Takeaway: CFGs have a nice normal form. (See end of 2.1 in the text for more details.)

Theorem: Every CFG has an equivalent in Chomsky Normal Form.

Next: Automata w/ stacks: Pushdown Automata.