COMS W3261 – Lecture 5.2:

Context-Free Grammars.

Idea: Introduce a new, more powerful way of describing languages

Example:

$$A \longrightarrow 0A1$$
$$A \longrightarrow B$$
$$B \longrightarrow \#$$

Variables: things that can be substituted (A, B). Often written as capital letters.

Terminals: symbols in the final string, cannot be substituted. (0, 1, #).

How to generate a string.

1. Writing down the start variable (top left)
2. Replacing any variable using a substitution rule
3. Repeat step 2 until only terminals remain.

$$A \longrightarrow 0A1 \longrightarrow 00A11 \longrightarrow 00B11 \longrightarrow 00\#11.$$

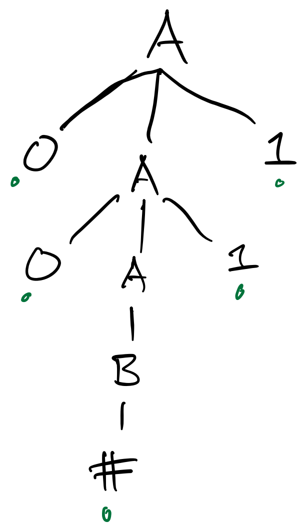
$$A \longrightarrow B \longrightarrow \#.$$

$$A \longrightarrow 0A1 \longrightarrow 0B1 \longrightarrow 0\#1.$$

Def. A sequence of substitutions used to create a string of terminals is called a derivation.

We can represent a derivation pictorally with a parse tree.

Ex. Parse tree for 00#11

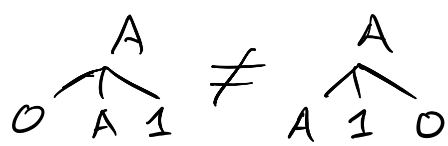


(Substituting each symbol according to the rule we use as we move downward.)

Read a parse tree by concatenating symbols left to right.

= 00#11.

$A \rightarrow 0A1$.



<u>Def.</u> The language $L(G)$ of a grammar G is the set of all strings that can be produced by derivation

$$G: \quad \begin{aligned} A &\rightarrow 0A1 \\ A &\rightarrow B \\ B &\rightarrow \# \end{aligned} \qquad\qquad L(G) = \{0^n \# 1^n \mid n \geq 0\}$$

<u>Def.</u> The set of all languages produced by a context-free (CFG) grammar is called the Context-Free Languages. (CFL)s.

<u>Example:</u> A fragment of English.

$S \longrightarrow \langle NP\rangle\langle VP\rangle$

// using $\langle\ \rangle$ to denote one single variable symbol.

$\langle NP\rangle \longrightarrow A\,N$

$\langle VP \rangle \to V \mid V\langle NP \rangle$ // the bar '|' abbreviates multiple rules as one

$N \to dog \mid cat \mid car \mid shoe \mid person$

$V \to smells \mid sees \mid is \mid eats$

$A \to a \mid the$

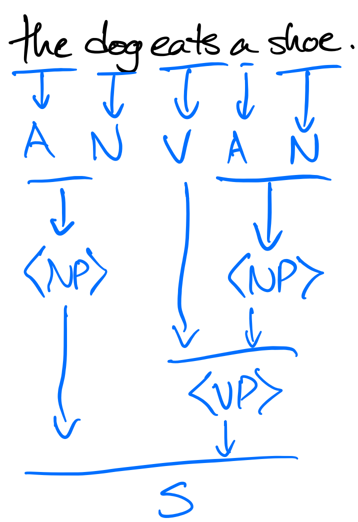
$N \to dog$
$N \to cat$
$N \to \cdots$

$S \to \langle NP \rangle\langle VP \rangle \to AN\langle VP \rangle \to aN\langle VP \rangle$

$aNV \to a\ shoe\ V \to a\ shoe\ is$

$S \to \langle NP \rangle\langle VP \rangle \to \langle NP \rangle V \langle NP \rangle \to ANV\langle NP \rangle \to$

$ANVAN \to \cdots \to$ the cat sees the person.



<u>Def.</u> (CFG, formally.) A context-free grammar is a 4-tuple, (V, Σ, R, S), where:

V is a finite set called the variables

Σ is a finite set called the terminals (disjoint from V)

R is a finite set of <u>rules</u>, where each rule maps 1 variable to a sequence of variables and terminals. (e.g., $A \to 01A$)

$S \in V$ is the start variable.

For any strings of variables and terminals u, v, and w, if $A \to w$ is a rule of the grammar, then we say that uAv yields uwv, where 'yields' is written $uAv \Rightarrow uvw$.

For any strings of variables + terminals u and v, we say u derives v, written $u \overset{*}{\Rightarrow} v$, if $u = v$ or if there exists a sequence $u_1, u_2, \cdots u_k$, $k \geq 0$, such that $u \Rightarrow u_1 \Rightarrow u_2 \cdots \Rightarrow u_k \Rightarrow v$.

(The language of a grammar G is $\{w \in \Sigma^* \mid S \overset{*}{\Rightarrow} w\}$.)

Example. $G_4 = (V, \Sigma, R, \langle Expr \rangle)$ where

$V = \{\langle Expr \rangle, \langle Term \rangle, \langle Factor \rangle\}$,

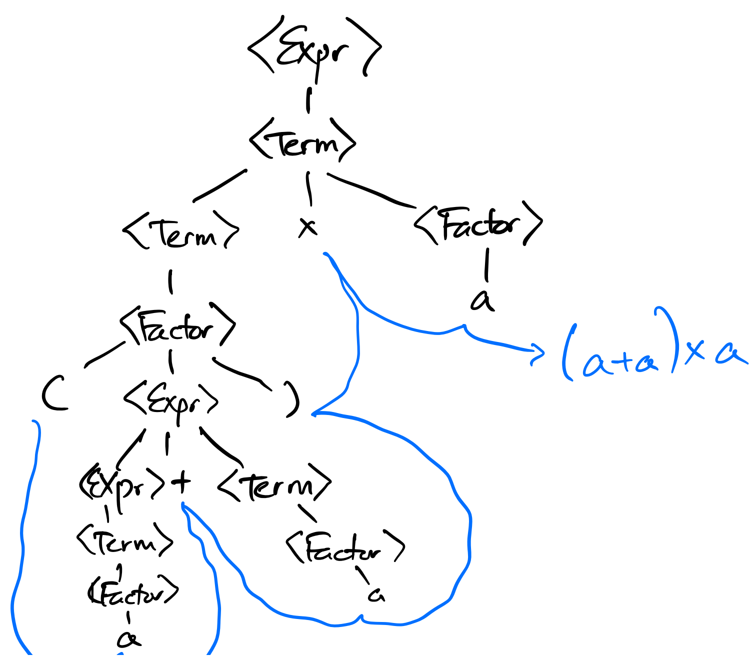
$\Sigma = \{a, +, \times, (,)\}$ "times symbol" $a \times b$.

$R = \quad \langle Expr \rangle \longrightarrow \langle Expr \rangle + \langle Term \rangle \mid \langle Term \rangle$

$\quad \langle Term \rangle \longrightarrow \langle Term \rangle \times \langle Factor \rangle \mid \langle Factor \rangle$

$\quad \langle Factor \rangle \longrightarrow (\langle Expr \rangle) \mid a$

$(a + a) \times a.$



Building CFGs — two techniques.

Suppose we want to build a CFG for $\{0^n 1^n \mid n \geq 0\}$

$L = \{0^n 1^n \mid n \geq 0\} \cup \{1^n 0^n \mid n \geq 0\}$

$S_1 \to 0S_1 1 \mid \varepsilon$ // recognizes $\{0^n 1^n \mid n \geq 0\}$

$S_2 \to 1S_1 0 \mid \varepsilon$ // recognizes $\{1^n 0^n \mid n \geq 0\}$

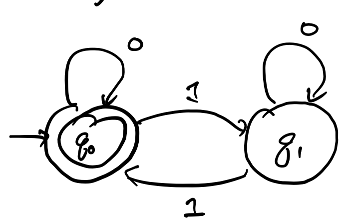
To take the 'union' of these CFGs, add a new start symbol.

$$G: \left.\begin{aligned} S &\to S_1 \mid S_2 \\ S_1 &\to 0S_1 1 \mid \varepsilon \\ S_2 &\to 1S_2 0 \mid \varepsilon \end{aligned}\right\} L(G) = L$$

(Aside: we made a CFG for a nonregular language (!!))

Technique 2. Converting a DFA to a CFG.

Goal: CFG for $L = \{w \mid w \in \{0,1\}^*,\ w$ has an even number of ones$\}$.



1. Make a variable R_i for each state q_i of the DFA.
2. For each transition $\delta(q_i, a) = q_j$, add the rule $R_i \to aR_j$
3. Add $R_i \to \varepsilon$ for each accept state.
4. R_0 is the start symbol.

(Σ is implicitly the same)

$\delta(q_0, 0) = 0$

$\Sigma = \Sigma$

$V = \{R_0, R_1\}$

$R = \begin{aligned} R_0 &\to 0R_0 \ * \\ R_0 &\to 1R_1 \ * \\ R_1 &\to 0R_1 \ - \\ R_1 &\to 1R_0 \ - \\ R_0 &\to \varepsilon \end{aligned}$

0101 : On my DFA, I start in q_0 and compute to, q_0, q_1, q_1, q_0, and then accept.

In DFA: $R_0 \to 0R_0 \to 01R_1 \to 010R_1 \to 0101R_0 \to 0101$.

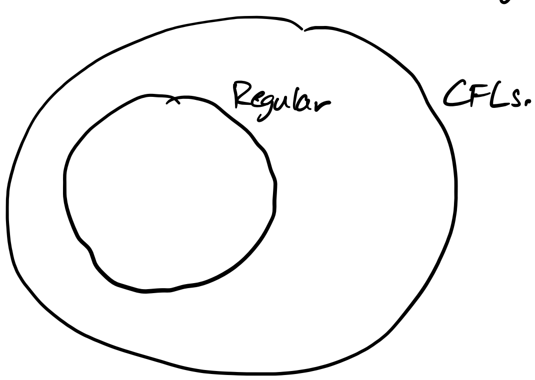
Claim: this (non-rigorous) construction shows that any regular language derives from some CFG (!)

(Reg. lang. $\to$ DFA $\to$ CFG.)

Regular Languages $\subseteq$ Context-Free Languages.

CFLs $\not\subseteq$ Reg. Languages.

Nonregular: $\{0^n 1^n \mid n \geq 0\}$



Def. If a grammar generates the same string in ways corresponding to different parse trees, it is ambiguous.

$\left(\begin{array}{l} S \to AA \\ A \to 1 \end{array}\right)$

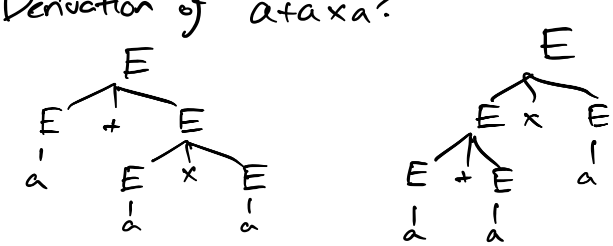
$S \to AA \to 1A \to 11$

$S \to AA \to A1 \to 11$

Example: $G_5 = (V, \Sigma, R, E)$, where $V = \{E\}$, $\Sigma = \{+, \times, (,), a\}$, R as follows:

$$E \to E \times E \mid E + E \mid (E) \mid a$$

Now: Derivation of $a + a \times a$?



Def. A leftmost derivation is one in which we always replace the leftmost variable. Formally, a grammar is ambiguous if it generates a string with at least two different leftmost derivations.

(leftmost #1) $E \to E+E \to a+E \to a+E\times E \to a+a\times E \to a+a\times a$

(not leftmost) $E \to E+\underline{E} \to E+E\times \underline{E} \to a+E\times E \to a+a\times E \to a+a\times a$

(leftmost #2) $E \to E\times E \to E+E\times E \to a+E\times E \to a+a\times \underline{E} \to a+a\times a.$

Next time: normal forms for grammars, pushdown automata!

Reminder: HW #3 due Monday @ 11:59 EST

Review: Section 2.1.