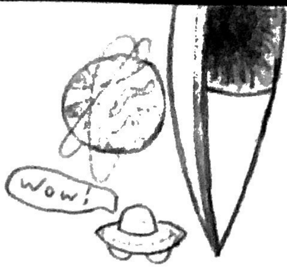
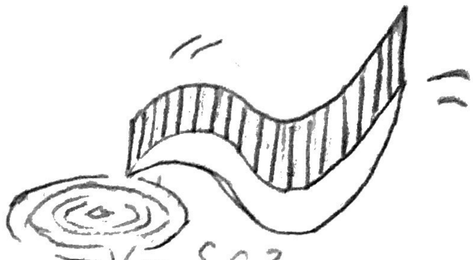
CS 334 ~ PS #5



"I pledge my honor that I have abided by the Stevens Honor System" - [illegible]

1.



A CFG in Chomsky Normal Form has each rule in the following form:

1) $A \to BC$, where B, C are variables in $V - \{S\}$

2) $A \to a$, where a is a terminal $\in \Sigma$

3) $S \to \varepsilon$, where S is the start variable in V

Show $2n-1$ applications are necessary to form a string of length of n, $n \geq 1$.

__Base__: $n=1$, then we just need to find a variable which points to the required terminal variable (prop. 2). That's one application.

__Assumption__: $2k-1$ for $n=k$. Show $2(k+1)-1$ for $n=k+1$ is true as a result.

Suppose we have a string of length k formed from our CFG.
→ By our inductive hypothesis, it took $2k-1$ applications to form that string.

→ Since it takes one application to transition every variable to a terminal, our string had k variables initially. This also means our string has $(2k-1)-(k) = k-1$ applications so far.

1 cont'd.

→ We want to add another 1 variable to our string to get $k+1$ characters. To do this, we take 1 variable say X & use property (II) to convert $X \to AB$, where AB is some variables in our CFG. This takes 1 action. We now have $(k-1)+1 = k$ applications.

→ Then we convert each of the $(k+1)$ variables to terminals, which requires $(k+1)$ actions. We now have $(k+1)+k = 2k+1$ applications in total & a string of length $k+1$.

But $2k+1 = 2(k+1)-1$ thus a string of length $k+1$ requires $2(k+1)-1$ applications to be created in a Chomsky Normal Form. ■

1b.* Convert to Chomsky Normal Form.

$$\left.\begin{array}{l} S \to aAa \mid bBb \mid \varepsilon \\ A \to C \mid a \\ B \to C \mid b \\ C \to CDA \mid \varepsilon \\ D \to A \mid B \mid ab \end{array}\right\} \to \left\{\begin{array}{l} S \to aAa \mid bBb \mid \varepsilon \\ A \to C \mid a \mid \varepsilon \\ B \to C \mid b \mid \varepsilon \\ C \to CDA \mid DA \\ D \to A \mid B \mid ab \end{array}\right.$$

We have start state

Eliminate $C \to \varepsilon$

1b cont'd.

$S \to aAa \mid bBb \mid aa \mid bb \mid \varepsilon$
$A \to C \mid a$
$B \to C \mid b$
$C \to CDA \mid CD \mid DA \mid D$
$D \to A \mid B \mid ab \mid \varepsilon$

eliminate $A \to \varepsilon$ & $B \to \varepsilon$

$S \to aAa \mid bBb \mid aa \mid bb \mid \varepsilon$
$A \to C \mid a$
$B \to C \mid b$
$C \to CDA \mid CA \mid CD \mid DA \mid A \mid D$
$D \to A \mid B \mid ab$

get rid of alone $D \to \varepsilon$
C to itself is redundant

$S \to aAa \mid bBb \mid aa \mid bb \mid \varepsilon$
$A \to CDA \mid CA \mid CD \mid DA \mid D \mid a$
$B \to CDA \mid CA \mid CD \mid DA \mid A \mid D \mid b$
$C \to CDA \mid CA \mid CD \mid DA \mid A \mid D$
$D \to A \mid B \mid ab$

$S \to aAa \mid bBb \mid aa \mid bb \mid \varepsilon$
$A \to CDA \mid CA \mid CD \mid DA \mid B \mid ab \mid a$
$B \to CDA \mid CA \mid CD \mid DA \mid A \mid ab \mid b$
$C \to CDA \mid CA \mid CD \mid DA \mid B \mid ab \mid a$
$D \to A \mid B \mid ab$

$S \to aAa \mid bBb \mid aa \mid bb \mid \varepsilon$
$A \to CDA \mid CA \mid CD \mid DA \mid ab \mid a \mid b$
$B \to CDA \mid CA \mid CD \mid DA \mid ab \mid a \mid b$
$C \to CDA \mid CA \mid CD \mid DA \mid ab \mid a \mid b$
$D \to CDA \mid CA \mid CD \mid DA \mid ab \mid a \mid b$

if you sub A into B / vice versa to eliminate the 1 variable transition; you get same result

Remove all 1 var transitions

$S \to aAa \mid bAb \mid aa \mid bb \mid \varepsilon$

$A \to AAA \mid AA \mid ab \mid a \mid b$

Condense all same variables

$\to$

$S \to BE \mid CF \mid EE \mid FF \mid \varepsilon$

$A \to DA \mid AA \mid EF \mid a \mid b$

$B \to aA$

$C \to bA$

$D \to AA$

$E \to a$

$F \to b$

Add variables to remove inconsistencies

$S \to BE \mid CF \mid EE \mid FF \mid \varepsilon$

$A \to DA \mid AA \mid EF \mid a \mid b$

$B \to EA$

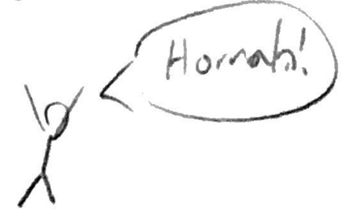
$C \to FA$

$D \to AA$

$E \to a$

$F \to b$

Our Chomsky Normal Form



#2) Since our start state references variables that only reference themselves, & are called independently we resolve the language described by each independently.

$$S \to TT \mid U$$

1) TT where $T \to 0T \mid T0 \mid \#$

Notice by the definition of T, the language described by T alone is the string containing the # terminal with any number of 0's on either side of it. Thus, $0^i \# 0^j$ $i,j \geq 0$.

$\to$ TT will describe the language $0^a \# 0^b 0^c \# 0^d$ $a,b,c,d \geq 0$

$\to$ $= 0^i \# 0^j \# 0^k$ $i,j,k \geq 0$

---

2) U where $U \to 0U00 \mid \#$

Notice the language described by U is the terminal #, with a number of zeroes to the left of # & double that number to the right of #, ie. $0^i \# 0^{2i}$ $i \geq 0$

Thus $L(G) = \{0^i \# 0^j \# 0^k : i,j,k \geq 0\} \cup \{0^i \# 0^{2i} : i \geq 0\}$

We now show $L(G)$ is not a regular language. Since regular languages are closed under union we can show just $\{0^i \# 0^{2i} : i \geq 0\}$ is non-regular.

To do so, we show there exists some string $s \in \{0^i\#0^{2i} : i \geq 0\}$ can be found s.t for any $p \geq 1$, then no matter how s is divided into $xyz = s$, then our 3 conditions

i) $\forall i \geq 0, xy^iz \in A$
ii) $|y| > 0$
iii) $|xy| \leq p$

cannot all be true.

Consider $s = 0^p\#0^{2p}$. Since $|xy| \leq p$ & $|y| > 0$, y must belong to the first string of 0's. However, if that is true

$$\rightarrow \forall i \geq 0 \quad xy^iz = 0^{p-|y|+|y|i}\#0^{2p} \notin \{0^i\#0^{2i} : i \geq 0\}$$

Thus $\{0^i\#0^{2i} : i \geq 0\}$ is not regular, & since union is closed under regularity (I know I mentioned this before, but it doesn't seem complete w/o it) L(G) is not regular. ∎

#3) $L_{add} = \{a^i b^{i+j} c^j : i,j \geq 0\}$ can be represented as a CFG s.t,

$S \to AB$

$A \to aAb \mid \varepsilon$

$B \to bBc \mid \varepsilon$

#3b) $L_{mult} = \{a^i b^{ij} c^j : i,j \geq 0\}$, we show that Lmult is not a context-free grammar.

That is there exists a string, for any pumping length $p \geq 1$ s.t the 3 properties:

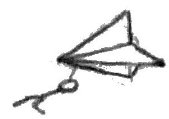
1. $\forall i \geq 0$, $uv^i x y^i z \in L_{mult}$
2. $|vy| > 0$
3. $|vxy| \leq p$

Suppose we take $w = a^p b^{p^2} c^p$ s.t p = pumping length.
→ vy can contain one of 5 different combinations of variables, since $|vxy| \leq p$.

1) vy only contains a's
2) vy only contains c's

→ whenever we pump only a or only c, w will become $w' = a^{p+k} b^{p^2} c^p$ & $w' = a^p b^{p^2} c^{p+k}$, for any $k \geq -|vy|$
→ our pumped string $w' \notin L_{mult}$.

3) vy only contains b's → when vy is pumped for $\forall i \geq 0$ then $w' = a^p b^{p^2+k} c^p$ s.t $k \geq 1 \cdot |vy|$, then since the # of b's can vary but a's & c's can't, then $w' \notin$ Lmult.



4) vy contains a's & b's

→ Then $\forall i \geq 0$, $uv^i x y^i z \in$ Lmult only if

$v = a^k$
$y = b^{jk}$ for all $k \geq 1$, since Lmult is in the form $\{a^i b^{i \cdot j} c^j : i,j \geq 0\}$

In other words we'd have to have y increase b by multiples of j, the # of a's added in each pump.

But if $k = 1$, → $v = a$, $y = b^j = b^p$ ← we are using $j, i = p$ for our selected string.

→ $|vxy| \geq p$ which violates property 3. that $|vxy| \leq p$.

Thus case 4 cannot satisfy all 3 properties of the extended pumping lemma simultaneously with our selected string.

5) vy contains b's & c's

→ The argument for this "case" is identical to case 4 except $v = b^{ik}$, $y = c^k$ for all $k \geq 1$.

Thus it is impossible for all 3 properties of ← to be satisfied simultaneously for $a^p b^{p^2} c^p$. Thus Lmult is not a Context free grammar.