Lecture 7 Outline

- Chomsky Normal Form
- Pumping lemma for CFLs
- Closure under $\cap$, etc.

**Def.** Let $G$ be a CFG. From last time, it's in Chomsky Normal Form if all rules have the form:

- I. $A \rightarrow BC$ \quad $A \notin V_f$, $B, C \in \{A, \ldots, F\}$
- II. $A \rightarrow \alpha$ \quad $A \notin V_f$, $\alpha \in \Sigma^*$
- III. $S \rightarrow \epsilon$ \quad $S$ the start variable, $\epsilon$ the empty string

**Thm.** If $L$ is context-free, then $\exists G$ in Chomsky normal form such that $L = L(G)$.

**Pf.** Given $G = (V, \Sigma, R, S, A)$

1. Add a new start variable $S'$ (and rename the old one).

   $$G' = (V \cup \{S\}, \Sigma, R \cup \{S' \rightarrow A \ \text{for each occurrence of } A\}, S)$$

2. Remove rules of the form $A \rightarrow \epsilon$.

   For every rule $B \rightarrow uAv$
   add rule $B \rightarrow uv$ for every occurrence of $A$.

   e.g. $B \rightarrow uAvAw$, $A \rightarrow \epsilon$
   add rules $B \rightarrow uvAw$, $B \rightarrow uAv$, $B \rightarrow uvw$.

   If $A$ occurs $n$ times in a rule, we add $2^n - 1$ additional rules. So step 2 can blow up the size of the grammar quite a bit.
(Proof of Theorem Continued)

But there's a problem with step 2: we could keep going forever! e.g.

\[ A \rightarrow B \quad A \rightarrow \epsilon \]
\[ B \rightarrow A \quad B \rightarrow \epsilon \]

Removing \( A \rightarrow \epsilon \) or \( B \rightarrow \epsilon \), then removing the other puts \( A \rightarrow \epsilon \) or \( B \rightarrow \epsilon \) back again!

So...

1. **Fixed Step 2**
   - **If** \( B \rightarrow uA_v \), add the rule \( B \rightarrow uv \) unless this causes you to add a rule which has previously been removed.

2. **Remove "unit rules"** — rules of the form \( A \rightarrow B \) where \( A, B \) are variables.
   - To remove \( A \rightarrow B \), for every rule that has \( B \rightarrow u \) (\( u \) a string of variables and terminals), add the rule \( A \rightarrow u \).
   - (Only add rule \( A \rightarrow u \) if you haven't previously removed it.)

3. **Make sure terminals appear on RHS alone.**
   - For every terminal \( a \in \Sigma \), add a variable \( V_a \) and a rule \( V_a \rightarrow a \). Then replace \( a \) with \( V_a \) whenever \( a \) is not alone on the RHS.

4. **Make it so that only 2 variables appear on the RHS of a rule.**
   - If \( A \rightarrow uvw \), add a variable \( A_1 \) and make rules
     \[ A \rightarrow A_1 uA_1 \quad A_1 \rightarrow vw \]. Repeat this until every string of variables is only of length 2.
Example of a conversion to Chomsky Normal Form.

The language \( \{0^n1^n \mid n \geq 0\} \) is recognized by the following CFG:

\[
S \rightarrow 0S1 \mid \varepsilon
\]

After rule 1:

\[
S \rightarrow A \\
A \rightarrow 0A1 \mid \varepsilon
\]

After rule 2:

\[
S \rightarrow A \mid \varepsilon \\
A \rightarrow 0A1 \mid 01
\]

After rule 3:

\[
S \rightarrow 0A1 \mid 01 \mid \varepsilon \\
A \rightarrow 0A1 \mid 01
\]

After rule 4:

\[
S \rightarrow V_0AV_1 \mid V_0V_1 \mid \varepsilon \\
A \rightarrow V_0AV_1 \mid V_0V_1 \\
V_0 \rightarrow 0 \\
V_1 \rightarrow 1
\]

After rule 5:

\[
S \rightarrow V_0S_1 \mid V_0V_1 \mid \varepsilon \\
S_1 \rightarrow AV_1 \\
A \rightarrow V_0A \mid V_0V_1 \\
A_1 \rightarrow AV_1 \\
V_0 \rightarrow 0 \\
V_1 \rightarrow 1
\]

Chomsky Normal Form CFG recognizing \( \{0^n1^n \mid n \geq 0\} \).
**FACT:** If $s \in L(G)$ and $G$ is in Chomsky Normal Form, and $|s| = n$, then every derivation of $s$ has length $2n-1$. (Except if $s$ is the empty string ($n=0$), in which case the derivation has length 1.)

We will come back to Chomsky Normal Form later, for now we do the Pumping Lemma for CFLs.

**PUMPING LEMMA FOR CFLs.**

If $L$ is a CFL, then $\exists p$ such that $\forall s \in L$, $|s| \geq p$, we can write $s = uvxyz$ such that

(i) $|vy| \geq 1$

(ii) $|vxy| \leq p$

(iii) $\forall i \geq 0$, $uv^ixy^iz \in L$.

**PICTURE:**

```
  S -> A
    |    |    |    |    |
  u  v  x  y  z
  |    |    |    |
  v  x  y
  |    |
  v  y

\[ s = uvxyz \in L \]

\[ uv^2xy^2z \in L. \]
```
Proof of Pumping Lemma for CFLs

Let $b$ be the max # of symbols on the RHS of a rule.

A parse tree of height $h$ has at most $b^h$ leaves.

So a string $s$ of length $b^h+1$ must have a position such that the length of the path to the root is $\geq h+1$.

Let $s$ be any string in $L$ of length $\geq b^{|s|+1}$.
If no such $s$ exists, the pumping lemma is vacuously true for $p = b^{|s|+1}$, so we're done.

Otherwise, let $T$ be the smallest parse tree for $s$ or the smallest if there is more than one.
Let $j$ be the position in $s$ such that its path in the parse tree $T$ is of length $\geq |s|+1$. So there is a repeat variable $A$ on the way to position $j$. Pick $A$ such that it occurs twice within the bottom-most $|s|+1$ steps of the path to position $j$.

Let $x$ be the string derived from the lower $A$, and let $vyxy$ be the string derived from the upper $A$.
Let $uvxyz$ be the entire string $s$.

(i) $|vy| \neq 0$ because if $vy = \varepsilon$ then "replacing" upper $A$ with lower $A$ results in a smaller parse tree, contradicting our choice of $T$. So $|vy| \geq 1.$
(ii) $|vxy| \leq p$ because upper $A$ is in the bottom $|V|+1$ levels of the tree, so the length of the string generated by upper $A$ is of length at most $b^{|V|+1}$.

(iii) $uv^ixy^iz$ is in $L$ for all $i \geq 0$ by "re-hanging" the tree, with $i$ copies of $A$ replacing the lower $A$.

Application of Pumping Lemma for CFLs:
The language

$$L = \{a^n b^n c^n : n \geq 0\}$$

is not a Context-free language.

Proof: Let $p$ be a pumping length and consider $a^pb^pc = uvxyz$.

Case 1. $v$ is a string of $0$ or more $a$'s, $y$ is a string of $0$ or more $b$'s, $|vy| > 1$, then

$$uvvxyyzz = a^{p+k}b^{p+m}c^p \in L.$$
Case 2: \( v = a^p, y = a^m \)
\[ uvuvyyz = p^{p+q+m} b^p c^p \in L \]
Similarly if \( v = b^p, y = b^m \) or \( v = c^p, y = c^m \).

Case 3: \( v = a^p b^m, \quad p \geq 1, m \geq 1. \)
This doesn't work.

etc.

Are context-free languages closed under \( \cap, U, \cup, \ast, \circ \)?

\[ \checkmark \text{Closed Under Union.} \]
\[ \text{Make start state } S \rightarrow S_1, S \rightarrow S_2 \]
where \( S_1 \) and \( S_2 \) are start states of two
CFGs for two languages.

\[ \checkmark \text{Closed Under Kleene *} \]
\[ \checkmark \text{Closed Under Concatenation} \]

But \( L_1 \cap L_2 \) is not context-free.

Both context-free

X NOT closed under complement.

EeXerise.