Parsing

**Parsing**: Deciding if $x \in \Sigma^*$ is in $L(G)$ for some CFG $G$.

**Review**

Consider the CFG $G$:

$$
S \rightarrow Aa \\
A \rightarrow AA \mid ABa \mid \lambda \\
B \rightarrow BBa \mid b \mid \lambda
$$

Is $ba$ in $L(G)$? Running time?

Remove $\lambda$-rules, then unit productions, and then useless productions from the grammar $G$ above. New grammar $G'$ is:

$$
S \rightarrow Aa \mid a \\
A \rightarrow AA \mid ABa \mid Aa \mid Ba \mid a \\
B \rightarrow BBa \mid Ba \mid a \mid b
$$

Is $ba$ in $L(G)$? Running time?

**Top-down Parser:**

- Start with $S$ and try to derive the string.

  $$
  S \rightarrow aS \mid b
  $$

- Examples: LL Parser, Recursive Descent
Bottom-up Parser:

- Start with string, work backwards, and try to derive S.

- Examples: Shift-reduce, Operator-Precedence, LR Parser

We will use the following functions FIRST and FOLLOW to aid in computing parse tables.

The function FIRST:

Some notation that we will use in defining FIRST and FOLLOW.

\[ G = (V, T, S, P) \]
\[ w, v \in (V \cup T)^* \]
\[ a \in T \]
\[ X, A, B \in V \]
\[ X_I \in (V \cup T)^+ \]

**Definition:** \( \text{FIRST}(w) = \) the set of terminals that begin strings derived from \( w \).

- If \( w \Rightarrow^* aw \) then 
  - \( a \) is in \( \text{FIRST}(w) \)
- If \( w \Rightarrow^* \lambda \) then 
  - \( \lambda \) is in \( \text{FIRST}(w) \)

To compute FIRST:

1. \( \text{FIRST}(a) = \{a\} \)
2. \( \text{FIRST}(X) \)
   - (a) If \( X \rightarrow aw \) then 
     - \( a \) is in \( \text{FIRST}(X) \)
   - (b) If \( X \rightarrow \lambda \) then 
     - \( \lambda \) is in \( \text{FIRST}(X) \)
   - (c) If \( X \rightarrow Aw \) and \( \lambda \in \text{FIRST}(A) \) then 
     - Everything in \( \text{FIRST}(w) \) is in \( \text{FIRST}(X) \)
3. In general, \( \text{FIRST}(X_1X_2X_3..X_K) = \)
   - \( \text{FIRST}(X_1) \)
   - \( \cup \text{FIRST}(X_2) \) if \( \lambda \) is in \( \text{FIRST}(X_1) \)
   - \( \cup \text{FIRST}(X_3) \) if \( \lambda \) is in \( \text{FIRST}(X_1) \)
   - and \( \lambda \) is in \( \text{FIRST}(X_2) \)
   - ...
   - \( \cup \text{FIRST}(X_K) \) if \( \lambda \) is in \( \text{FIRST}(X_1) \)
   - and \( \lambda \) is in \( \text{FIRST}(X_2) \)
   - ...
   - and \( \lambda \) is in \( \text{FIRST}(X_{K-1}) \)
   - \( \{\lambda\} \) if \( \lambda \notin \text{FIRST}(X_J) \) for all \( J \)
Example: \(L = \{a^n b^m c^n : n \geq 0, 0 \leq m \leq 1\}\)

\[
\begin{align*}
S & \rightarrow aSc \mid B \\
B & \rightarrow b \mid \lambda
\end{align*}
\]

FIRST(B) = 
FIRST(S) = 
FIRST(Sc) = 

Example

\[
\begin{align*}
S & \rightarrow BCD \mid aD \\
A & \rightarrow CEB \mid aA \\
B & \rightarrow b \mid \lambda \\
C & \rightarrow dB \mid \lambda \\
D & \rightarrow cA \mid \lambda \\
E & \rightarrow e \mid fE
\end{align*}
\]

FIRST(S) = 
FIRST(A) = 
FIRST(B) = 
FIRST(C) = 
FIRST(D) = 
FIRST(E) = 

Definition: FOLLOW(X) = set of terminals that can appear to the right of X in some derivation.

If \(S \Rightarrow wAav\) then
a is in FOLLOW(A)

(where \(w\) and \(v\) are strings of terminals and variables, \(a\) is a terminal, and \(A\) is a variable)
To compute FOLLOW:

1. $ is in FOLLOW(S)
2. If $A \rightarrow wBv$ and $v \neq \lambda$ then
   \[ \text{FIRST}(v) - \{ \lambda \} \text{ is in FOLLOW}(B) \]
3. IF $A \rightarrow wb$ OR
   \[ A \rightarrow wBv \text{ and } \lambda \text{ is in FIRST}(v) \text{ then} \]
   \[ \text{FOLLOW}(A) \text{ is in FOLLOW}(B) \]
4. $\lambda$ is never in FOLLOW

Example:

\[
\begin{align*}
S & \rightarrow aSc \mid B \\
B & \rightarrow b \mid \lambda
\end{align*}
\]

FOLLOW(S) =

FOLLOW(B) =

Example:

\[
\begin{align*}
S & \rightarrow BCD \mid aD \\
A & \rightarrow CEB \mid aA \\
B & \rightarrow b \mid \lambda \\
C & \rightarrow dB \mid \lambda \\
D & \rightarrow cA \mid \lambda \\
E & \rightarrow e \mid fE
\end{align*}
\]

FOLLOW(S) =

FOLLOW(A) =

FOLLOW(B) =

FOLLOW(C) =

FOLLOW(D) =

FOLLOW(E) =