Section: Parsing

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

Consider the CFG $G$:

$$
\begin{align*}
S & \rightarrow Aa \\
A & \rightarrow AA \mid ABa \mid \lambda \\
B & \rightarrow BBa \mid b \mid \lambda
\end{align*}
$$

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

New grammar $G'$ is:

$$
\begin{align*}
S & \rightarrow Aa \mid a \\
A & \rightarrow AA \mid ABa \mid Aa \mid Ba \mid a \\
B & \rightarrow BBa \mid Ba \mid a \mid b
\end{align*}
$$

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
The function FIRST:

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

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

If \( w \Rightarrow^* av \) then

\( a \) is in \( FIRST(w) \)

If \( w \Rightarrow^* \lambda \) then

\( \lambda \) is in \( FIRST(w) \)
To compute FIRST:

1. \text{FIRST}(a) = \{a\}

2. \text{FIRST}(X)
   
   (a) If $X \rightarrow aw$ then
       
       \begin{itemize}
       \item a is in FIRST(X)
       \end{itemize}
   
   (b) IF $X \rightarrow \lambda$ then
       
       \begin{itemize}
       \item $\lambda$ is in FIRST(X)
       \end{itemize}
   
   (c) If $X \rightarrow Aw$ and $\lambda \in \text{FIRST}(A)$ then
       
       Everything in FIRST(w) is in FIRST(X)
3. In general, FIRST($X_1X_2X_3\ldots X_K$) =

- FIRST($X_1$)
- $\cup$ FIRST($X_2$) if $\lambda$ is in FIRST($X_1$)
- $\cup$ FIRST($X_3$) if $\lambda$ is in FIRST($X_1$) and $\lambda$ is in FIRST($X_2$)
  ...
- $\cup$ FIRST($X_K$) if $\lambda$ is in FIRST($X_1$) and $\lambda$ is in FIRST($X_2$) ...
  and $\lambda$ is in FIRST($X_{K-1}$)
- $\{-\lambda\}$ if $\lambda \not\in$ FIRST($X_J$) for all $J$
Example:

\[ S \rightarrow aSc \mid B \]
\[ B \rightarrow b \mid \lambda \]

\[
\text{FIRST}(B) = \\
\text{FIRST}(S) = \\
\text{FIRST}(Sc) =
\]
Example

\[ S \rightarrow BCD \mid aD \]
\[ A \rightarrow CEB \mid aA \]
\[ B \rightarrow b \mid \lambda \]
\[ C \rightarrow dB \mid \lambda \]
\[ D \rightarrow cA \mid \lambda \]
\[ E \rightarrow e \mid fE \]

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.

\[
\text{If } S \xrightarrow{*} wAav \text{ then } a \text{ is in FOLLOW}(A)
\]

To compute FOLLOW:

1. $ is in FOLLOW(S)
2. If A \xrightarrow{} wBv and v \neq \lambda \text{ then} 
   FIRST(v) - \{\lambda\} \text{ is in FOLLOW(B)}
3. IF A \xrightarrow{} wB OR 
   A \xrightarrow{} wBv and \lambda \text{ is in FIRST(v)}
   then 
   FOLLOW(A) \text{ is in FOLLOW(B)}
4. \lambda \text{ is never in FOLLOW}
Example:

\[ S \rightarrow aSc \mid B \]
\[ B \rightarrow b \mid \lambda \]

FOLLOW(S) =

FOLLOW(B) =
Example:

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

\[
\begin{align*}
\text{FOLLOW}(S) & = \\
\text{FOLLOW}(A) & = \\
\text{FOLLOW}(B) & = \\
\text{FOLLOW}(C) & = \\
\text{FOLLOW}(D) & = \\
\text{FOLLOW}(E) & =
\end{align*}
\]