

Section: Parsing Ch. 15

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

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?

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

The function FIRST:

$$\mathbf{G} = (\mathbf{V}, \mathbf{T}, \mathbf{S}, \mathbf{P})$$

$$\mathbf{w}, \mathbf{v} \in (\mathbf{V} \cup \mathbf{T})^*$$

$$\mathbf{a} \in \mathbf{T}$$

$$\mathbf{X}, \mathbf{A}, \mathbf{B} \in \mathbf{V}$$

$$\mathbf{X}_I \in (\mathbf{V} \cup \mathbf{T})^+$$

Definition: FIRST

Given a context-free grammar $\mathbf{G} = (\mathbf{V}, \mathbf{T}, \mathbf{S}, \mathbf{P})$, $\mathbf{a} \in \mathbf{T}$ and $\mathbf{w}, \mathbf{v} \in (\mathbf{V} \cup \mathbf{T})^*$, the $\mathbf{FIRST}(\mathbf{w})$ is the set of terminals that can be the first terminal \mathbf{a} in $\mathbf{w} \xRightarrow{*} \mathbf{a}\mathbf{v}$. λ is in $\mathbf{FIRST}(\mathbf{w})$ if $\mathbf{w} \xRightarrow{*} \lambda$.

We show how to calculate FIRST for variables and terminals in the grammar, for λ and for strings.

Algorithm for FIRST

Given a grammar $G=(V, T, S, P)$,
calculate $\text{FIRST}(w)$ for w in $(V \cup T)^*$,

1. For $a \in T$, $\text{FIRST}(a) = \{a\}$.
2. $\text{FIRST}(\lambda) = \{\lambda\}$.
3. For $A \in V$, set $\text{FIRST}(A) = \{\}$.
4. Repeat these steps until no more terminals or λ can be added to any FIRST set for variables.

For every production $A \rightarrow w$

$$\text{FIRST}(A) = \text{FIRST}(A) \cup \text{FIRST}(w)$$

5. For $w = x_1x_2x_3 \dots x_n$ where $x_i \in (V \cup T)$

a) $\text{FIRST}(w) = \text{FIRST}(x_1)$

b)

For i from 2 to n do:

if $x_j \xRightarrow{*} \lambda$ for all j from 1 to $i - 1$ then

$\text{FIRST}(w) = \text{FIRST}(w) \cup \text{FIRST}(x_i) - \{\lambda\}$

c)

If $x_i \xRightarrow{*} \lambda$ for all i from 1 to n then

$\text{FIRST}(w) = \text{FIRST}(w) \cup \{\lambda\}$

Example:

$$\begin{aligned} S &\rightarrow aSc \mid B \\ B &\rightarrow b \mid \lambda \end{aligned}$$

$$\text{FIRST}(B) =$$

$$\text{FIRST}(S) =$$

$$\text{FIRST}(Sc) =$$

Example

$$\begin{aligned} S &\rightarrow \mathbf{BCD} \mid \mathbf{aD} \\ A &\rightarrow \mathbf{CEB} \mid \mathbf{aA} \\ B &\rightarrow \mathbf{b} \mid \lambda \\ C &\rightarrow \mathbf{dB} \mid \lambda \\ D &\rightarrow \mathbf{cA} \mid \lambda \\ E &\rightarrow \mathbf{e} \mid \mathbf{fE} \end{aligned}$$

$$\mathbf{FIRST(S)} =$$

$$\mathbf{FIRST(A)} =$$

$$\mathbf{FIRST(B)} =$$

$$\mathbf{FIRST(C)} =$$

$$\mathbf{FIRST(D)} =$$

$$\mathbf{FIRST(E)} =$$

Definition: FOLLOW

Given a context-free grammar $G = (V, T, S, P)$, $A \in V$, $a \in T$ and $w, v \in (V \cup T)^*$, **FOLLOW**(A) is the set of terminals that can be the first terminal a immediately following A in some sentential form $vAaw$. $\$$ is always in **FOLLOW**(S).

Algorithm for FOLLOW

To calculate FOLLOW for the variables in $G=(V, T, S, P)$. Let $A, B \in V$ and $v, w \in (V \cup T)^*$.

1. $\$$ is in $FOLLOW(S)$.
2. For $A \rightarrow vB$, $FOLLOW(A)$ is in $FOLLOW(B)$.
3. For $A \rightarrow vBw$:
 - (a) $FIRST(w) - \{\lambda\}$ is in $FOLLOW(B)$.
 - (b) If $\lambda \in FIRST(w)$, then $FOLLOW(A)$ is in $FOLLOW(B)$.

Example:

$$\begin{aligned} S &\rightarrow aSc \mid B \\ B &\rightarrow b \mid \lambda \end{aligned}$$

$$\text{FOLLOW}(S) =$$

$$\text{FOLLOW}(B) =$$

Example:

$$\begin{aligned} S &\rightarrow \mathbf{BCD} \mid \mathbf{aD} \\ A &\rightarrow \mathbf{CEB} \mid \mathbf{aA} \\ B &\rightarrow \mathbf{b} \mid \lambda \\ C &\rightarrow \mathbf{dB} \mid \lambda \\ D &\rightarrow \mathbf{cA} \mid \lambda \\ E &\rightarrow \mathbf{e} \mid \mathbf{fE} \end{aligned}$$

FOLLOW(S) =

FOLLOW(A) =

FOLLOW(B) =

FOLLOW(C) =

FOLLOW(D) =

FOLLOW(E) =