

For grammar $G = (N, T, P, S)$
 Define $T_\epsilon = T \cup \{\epsilon\}$
 ϵ end-of-input marker
 $T_\epsilon = T \cup \{\epsilon\}$

For any grammar symbol X
 Define $FIRST(X)$ as follows:
 If $X \in T$, then
 $FIRST(X) = \{X\}$
 If $X \in N$, then
 For every production
 $X \rightarrow Y_1 Y_2 \dots Y_k$ in G ,
 any terminal $a \in T$,
 $a \in FIRST(X)$ iff
 $a \in FIRST(Y_i)$ for some $1 \leq i \leq k$
 and ϵ is in $FIRST(Y_i)$ for all $i \leq k$.
 (Note: $\epsilon \in FIRST(Y_i)$ means Y_i yields ϵ)
 If $\epsilon \in FIRST(Y_i)$ for all $1 \leq i \leq k$,
 then put ϵ into $FIRST(X)$.

Notes: $FIRST(X) \subseteq T_\epsilon$
 For all grammar symbols X ,
 Extend defn to $FIRST(u)$
 where u is a string of
 grammar symbols
 $u = Y_1 Y_2 \dots Y_k$ $a \in T$
 $a \in FIRST(u)$ iff $\exists i$ ($1 \leq i \leq k$)
 $a \in FIRST(Y_i)$ and $\epsilon \in FIRST(Y_1 \dots Y_{i-1})$
 $\epsilon \in FIRST(u)$ iff $\epsilon \in FIRST(Y_i)$
 for all $i, 1 \leq i \leq k$
 (if $k=0$, then $FIRST(\epsilon) = \{\epsilon\}$).

Feb 21-11:00 AM

Computing FIRST sets
 Start with for single grammar
 symbols X (for any string
 of k grammar symbols compute
 $FIRST(u)$ on-the-fly (as needed)
 based on table of $FIRST(X)$
 sets).

Start with all sets empty
 repeat
 For any $a \in T$, add
 a to $FIRST(a)$
 For $A \in N$ and
 production $A \rightarrow X_1 \dots X_k$
 add all terminals of
 $FIRST(X_i)$ to $FIRST(A)$
 If $\epsilon \in FIRST(X_i)$, then
 add all terminals of $FIRST(X_i)$
 to $FIRST(A)$, etc.
 If ϵ is in all the $FIRST(X_i)$,
 then add ϵ to $FIRST(A)$
 until nothing more can be added
 to any FIRST set

Follow sets: For any nonterminal
 A , $FOLLOW(A) \subseteq T_\epsilon$
 $FOLLOW(A)$ is all
 symbols that could
 possibly immediately
 yield A in a parse tree
 yield A in a parse tree

Define FOLLOW sets by
 the algorithm to construct
 them.

Feb 21-11:16 AM

Start with $FOLLOW(A) = \emptyset$
 for all $A \in N$.

repeat
 Add ϵ to $FOLLOW(\epsilon)$,
 where ϵ is the start
 symbol.
 Let $A \rightarrow \alpha B \gamma$ where
 $B \in N$ and α, γ are strings
 of grammar symbols.
 Put all terminals in
 $FIRST(\gamma)$ into $FOLLOW(B)$.

If $\epsilon \in FIRST(\gamma)$, then
 add everything in $FOLLOW(A)$
 to $FOLLOW(B)$.
 until nothing more can be added.

Ex:
 $E \rightarrow TT'$
 $T' \rightarrow \epsilon \mid +TT' \mid -TT'$
 $T \rightarrow FF'$
 $F' \rightarrow \epsilon \mid *FF' \mid /FF'$
 $F \rightarrow c \mid (E)$

nonterminal A	$FIRST(A)$
E	$\{T, (, c\}$
T	$\{F, +, -, c\}$
T'	$\{\epsilon, +, -, c\}$
F	$\{c, (*, /)\}$
F'	$\{\epsilon, (*, /)\}$

done

Feb 21-11:27 AM

A	$FOLLOW(A)$
E <td>$\{\\$,)\}$ $F \rightarrow (B)$</td>	$\{\$,)\}$ $F \rightarrow (B)$
T' <td>$\{\\$,)\}$ $E \rightarrow TT'$</td>	$\{\$,)\}$ $E \rightarrow TT'$
T <td>$\{\\$,), +, -\}$</td>	$\{\$,), +, -\}$
F' <td>$\{\\$,), +, -, \epsilon\}$ $T \rightarrow FF'$</td>	$\{\$,), +, -, \epsilon\}$ $T \rightarrow FF'$
F <td>$\{\\$,), +, -, \epsilon, /\}$</td>	$\{\$,), +, -, \epsilon, /\}$

done
 Let G be a grammar.
 Thm: G is LL(1) iff,
 for any two productions
 $A \rightarrow \alpha \mid \beta$,
 $FIRST(\alpha) \cap FIRST(\beta) = \emptyset$
 and if $\epsilon \in FIRST(\alpha)$ then
 $FOLLOW(A) \cap FIRST(\beta) = \emptyset$
 similarly, if $\epsilon \in FIRST(\beta)$ then
 $FOLLOW(A) \cap FIRST(\alpha) = \emptyset$

Bottom-up parsing & syntax-
 directed translation

pushdown automaton (PDA)

4 actions:
 accept — halting without error
 error — no production
 shift s — s is a terminal
 reduce $A \rightarrow \alpha$ — A and α is a production

Feb 21-11:43 AM

Shift s : s is the lookahead.
 — advance the lookahead
 (consume s)
 — push s onto the stack

reduce $A \rightarrow \alpha$: α is a production
 α is on the stack
 (rightmost symbol of
 α is top of stack)

— pop α from the stack
 — push A onto stack
 (don't advance lookahead)

Pushdown automaton (PDA)

$E \rightarrow S \mid TT'$ $S \in \{ \text{input} \}$
 $T \rightarrow T * F \mid F$ $F \in \{ +, -, *, / \}$
 $F \rightarrow c \mid (E)$

Feb 21-12:05 PM