CS 131 Compilers: Discussion 4: Syntax-Derivative Tree Scheme

杨易为 季杨彪 尤存翰
{yangyw,jiyb,youch}@shanghaitech.edu.cn

2021 年 3 月 23 日

1 LL Parsing Ambiguities

An LL(k) grammar is a CFG used by a parser that scans input left-to-right (“L”), leftmost derivation (“L”), and uses k tokens of lookahead to predict the correct production. We’ ve previously seen that a grammar is ambiguous if it has a parse tree that is not unique. A more formal definition of LL conflicts uses FIRST and FOLLOW sets.

1. **FIRST(A)** the set of all terminals that could occur first in an expansion of the terminal or nonterminal A (include ϵ if A can expand to ϵ)
2. **FOLLOW(A)** the set of all terminals that could follow an occurrence of the terminal or nonterminal A in a (partial) derivation.

There are two main types of LL(1) conflicts:

1. **FIRST/FIRST** The FIRST sets of two different productions for same nonterminal intersect.
2. **FIRST/FOLLOW**: The FIRST set of a grammar rule contains an epsilon and the intersection with its FOLLOW set is not empty.

Is the following grammar LL(1)? Justify your answer using FIRST and FOLLOW sets.

$S \rightarrow Xd$
$X \rightarrow C|Ba$
$C \rightarrow \epsilon$
$B \rightarrow d$

Answer:

2 Resolving Conflicts

Consider the following grammar for numerical expressions with division, addition, and unary minus:

$$E \rightarrow Num|E/E|E+E|-E$$

1. Rewrite the grammar so that it is LL(1), so that ‘/’ has higher precedence than ‘+’, and so that ‘-’ has highest precedence. ‘+’ and ‘/’ should be parsed in a right-associative way.
2. Compute the FIRST and FOLLOW sets for your re-written LL(1) grammar.
3. Draw the LL(1) parsing table for the grammar. You may need the following rules:
   (a) For each production $X \rightarrow A_1 \ldots A_n$:
   (b) For each $1 \leq i \leq n$, and for each b in First (A_i) : Set $T[X,b] = X \rightarrow A_1 \ldots A_n$. Stop when ϵ is not in First (A_i) .
   (c) If $A_1 \ldots A_n \rightarrow^* \epsilon$, then for each b in Follow (X) : Set $T[X,b] = \epsilon$

Answer:

3 Earley's Algorithm

Consider the following CFG with terminals $\{(,), +, *, a, b\}$ (+ represents union) that is used to represent regular expressions over alphabet $\{a, b\}$:

$$R \rightarrow R + R|RR|(R)\,|R^{*}|\; a \mid b$$

1. Using the above CFG, provide a derivation for the following input string $(a + (ba)^{*}b)^{*}$.
2. For the derivation in above solution, provide the corresponding parse tree.

Answer: