CS 241 – Week 8 Tutorial

Top-Down Parsing: LL(1) Parsing

Spring 2015

Definitions

- LL(1) stands for *Left-to-right* scan of input, *Left-canonical* derivation, *1* symbol of lookahead.
- An LL(1) parser is a *top-down* parser; it begins from the start symbol and finds a derivation for the input string.
- An LL(1) grammar is a grammar that can be parsed by an LL(1) parser
- The LL(1) prediction function is defined as:

$$\begin{aligned} Predict(A,a) &= \{A \to \gamma | a \in First(\gamma) \vee (\gamma \Rightarrow^* \varepsilon \wedge a \in Follow(A))\} \\ First(\gamma) &= \{b | \gamma \Rightarrow^* b\beta \text{ for some } \beta\} \\ Follow(A) &= \{c | S' \Rightarrow^* \alpha A c \beta \text{ for some } \alpha, \beta\} \end{aligned}$$

  In other words, $First(\gamma)$ is dependent on the $First$ sets for leftmost non-Terminals contained within.

- A grammar is LL(1) if $\forall A, a \ |Predict(A,a)| \leq 1$.

Top-down Parsing (LL(1))

Consider the following context-free grammar:

$$\begin{aligned} S' &\to \vdash S \dashv \\ S &\to aXYb \\ S &\to XY \\ X &\to pX \\ X &\to \epsilon \\ Y &\to q \\ Y &\to \epsilon \end{aligned}$$

1. Using the definition for *Predict* create the LL(1) predictor table for the grammar.
2. Using the LL(1) predictor table for this grammar, perform a top-down parse of the string ⊢appqb⊣ and draw the parse tree.
3. Consider the following context-free grammar:

$$\begin{aligned} S' &\to \vdash S \dashv \\ S &\to Sab \\ S &\to XY \end{aligned}$$

$$\begin{aligned} X &\rightarrow pX \\ X &\rightarrow \epsilon \\ Y &\rightarrow q \\ Y &\rightarrow \epsilon \end{aligned}$$

Why is it impossible to parse this grammar with a top-down LL(1) parser? (That is, what aspect of the grammar makes it not LL(1)?)