CS143 Compilers - Written Assignment 2

Due Monday, April 30, 2018 at 11:59 PM

This assignment covers context free grammars and parsing. You may discuss this assignment with other students and work on the problems together. However, your write-up should be your own individual work, and you should indicate in your submission who you worked with, if applicable. Assignments can be submitted electronically through Gradescope as a PDF by 11:59 PM PDT. A LaTeX template for writing your solutions is available on the course website.

1. Give the context-free grammar for each of the following languages:

   (a) The set of all strings over the alphabet $\{2, 7, +, (,)\}$ representing valid arithmetic expressions in base-10 that evaluate to an odd number. Your CFG should allow for multi-digit numbers.
   Examples of strings in the language:

   $$7 \qquad (72 + 72) + 27 \qquad (((777)))$$

   Examples of strings **not** in the language:

   $$22 + 22 \qquad (72 + 27)) \qquad 22 + 77722 + 22 + +7$$

   (b) The set of all strings over the alphabet $\{a, b\}$ where the number of b's is at most one greater than the number of a's.
   Examples of strings in the language:

   $$aaaba \qquad b \qquad bbaabaaaaaaabaa$$

   Examples of strings **not** in the language:

   $$babb \qquad abababababbb \qquad bbbbb$$

   (c) The set of all strings over the alphabet $\{0, 1\}$ in the language $L : \{0^i 1^j 0^k \mid j = i + k\}$.
   Examples of strings in the language:

   $$001110 \qquad 000111 \qquad 111000$$

   Examples of strings **not** in the language:

   $$01110 \qquad 101 \qquad 11111$$

(d) The set of all strings over the alphabet $\{[,],\{,\},,\}$ which are sets (for clarification, "," is in the alphabet). We define a set to be a collection of zero or more comma-separated arrays enclosed in an open brace and a close brace. Similarly, we define an array to be a collection of zero or more comma-separated sets enclosed in an open bracket and a close bracket.
Examples of sets:

$$\{\} \qquad \{[],[]\} \qquad \{[],[\{\}]\}$$

Examples of arrays:

$$[] \qquad [\{\},\{[]\},\{\}] \qquad [\{[\{\}]\}]$$

Examples of strings in the language:

$$\{\} \qquad \{[],[]\} \qquad \{[\{\}],[],[\{\},\{\}]\}$$

Examples of strings **not** in the language:

$$[] \qquad \{[],\{\}\} \qquad \{\},[]$$

2. (a) Left factor the following grammar:

$$S \to S^* \mid S \cup S \mid S? \mid [T]$$
$$T \to Ta \mid Tb \mid Tc \mid \epsilon$$

(b) Eliminate left recursion from the following grammar:

$$S \to Sab \mid S! \mid (T) \mid bTb$$
$$T \to Ta \mid Tb \mid Tc \mid \epsilon$$

3. Consider the following CFG, where the set of terminals is $\Sigma = \{a, b, \#, \%, !\}$:

$$S \to \%aT \mid U!$$
$$T \to aS \mid baT \mid \epsilon$$
$$U \to \#aTU \mid \epsilon$$

(a) Construct the FIRST sets for each of the nonterminals.

(b) Construct the FOLLOW sets for each of the nonterminals.

(c) Construct the LL(1) parsing table for the grammar.

(d) Show the sequence of stack, input and action configurations that occur during an LL(1) parse of the string "$\#abaa\%aba!$". At the beginning of the parse, the stack should contain a single S.

4. What advantage does left recursion have over right recursion in shift-reduce parsing?
Hint: Consider left and right recursive grammars for the language a^*. What happens if your input has a million a's?

5. Consider the following grammar G over the alphabet $\Sigma = \{a, b, c\}$:

$$
\begin{aligned}
S' &\to S \\
S &\to Aa \\
S &\to Bb \\
A &\to Ac \\
A &\to \epsilon \\
B &\to Bc \\
B &\to \epsilon
\end{aligned}
$$

You want to implement G using an SLR(1) parser (note that we have already added the $S' \to S$ production for you).

(a) Construct the first state of the LR(0) machine, compute the FOLLOW sets of A and B, and point out the conflicts that prevent the grammar from being SLR(1).

(b) Show modifications to production 4 ($A \to Ac$) and production 6 ($B \to Bc$) that make the grammar SLR(1) while having the same language as the original grammar G. Explain the intuition behind this result.

6. Consider the following CFG, where the set of terminals is $\Sigma = \{[,], ,, \textbf{int}\}$:

$$
\begin{aligned}
S' &\to S \\
S &\to [B \\
A &\to \textbf{int} \mid [B \\
B &\to] \mid C \\
C &\to A] \mid A, C
\end{aligned}
$$

The grammar describes how arrays may be declared. Arrays consist of zero or more elements where each element is either an integer (represented by **int**) or an array (thus we may have nested arrays). Note that comma is a terminal, and we have already added a dummy production $S' \to S$ for you.

(a) Construct a DFA for viable prefixes of the grammar using LR(0) items.

(b) Identify any shift-reduce and reduce-reduce conflicts in the grammar under the SLR(1) rules.

(c) Assuming that an SLR(1) parser resolves shift-reduce conflicts by choosing to shift, show the operation of such a parser on the input string "[**int**,[]]". Your table should include a "Configuration" column, a "DFA Halt State" column, and an "Action" column.