Formal Languages and Compilers
Exercises on Syntax Analysis I with Solutions

MSc in Computer Science, University of Camerino
prof. Luca Tesei

Exercise 1

Consider the following grammar:

$$\begin{array}{lcl} S & \rightarrow & aSb \mid Ad \mid Bc \\ A & \rightarrow & Aa \mid c \\ B & \rightarrow & ddA \mid dC \\ C & \rightarrow & ac \end{array}$$

1. Formalise the language generated by the grammar
2. Is the grammar $LL(k)$ for some k?
3. Construct the table of a top-down non-recursive predictive parser for the language.

Solution

1) The language can be formalised as follows:

$$\{a^n\, c\, a^*\, d\, b^n \mid n \geq 0\} \cup \{a^n\, d\, a\, c\, c\, b^n \mid n \geq 0\} \cup \{a^n\, d\, d\, c\, a^*\, c\, b^n \mid n \geq 0\}$$

2) The grammar is not $LL(k)$ for any k because it has an immediate left recursion in the production: $A \rightarrow Aa$.

3) Let us eliminate the left recursion and also factorise the productions of B. We obtain the following grammar:

$$
\begin{array}{lcl}
S & \rightarrow & aSb \mid Ad \mid Bc \\
A & \rightarrow & cA' \\
A' & \rightarrow & aA' \mid \epsilon \\
B & \rightarrow & dB' \\
B' & \rightarrow & dA \mid C \\
C & \rightarrow & ac
\end{array}
$$

We have FOLLOW(S) $= \{\$, b\}$, FOLLOW($A$) $= \{d, c\} =$ FOLLOW(A'), FOLLOW(B) $= \{c\} =$ FOLLOW(B') $=$ FOLLOW(C).

This modified grammar is $LL(1)$ and the parsing table is the following:

	a	b	c	d	$
S	$S \rightarrow aSb$		$S \rightarrow Ad$	$S \rightarrow Bc$	
A			$A \rightarrow cA'$		
A'	$A' \rightarrow aA'$		$A' \rightarrow \epsilon$	$A' \rightarrow \epsilon$	
B				$B \rightarrow dB'$	
B'	$B' \rightarrow C$			$B' \rightarrow dA$	
C	$C \rightarrow ac$				

Exercise 2

Consider the following grammar:

$$
\begin{array}{lcl}
S & \rightarrow & A \mid Bbb \\
A & \rightarrow & aB \\
B & \rightarrow & aAb \mid b
\end{array}
$$

1. Formalise the language generated by the grammar
2. Is the grammar LR(1)?
3. Is the string $aaAb$ a viable prefix? If the answer is yes, enumerate the valid $LR(0)$ items for this prefix.

Solution

1) The language is

$$\{a^{2n+1}\, b^{n+1} \mid n \geq 0\} \cup \{a^{2n}\, b^{n+3} \mid n \geq 0\}$$

2) Let us first construct the collection of $LR(0)$ items. If there are no conflicts then the grammar is $SLR(1)$ and so also $LR(1)$. Let us augment the grammar, as usual, with the production $S' \to S$.

$I_0 =$	$S' \to \cdot S$ $S \to \cdot A$ $S \to \cdot Bbb$ $A \to \cdot aB$ $B \to \cdot aAb$ $B \to \cdot b$	$I_1 = goto(I_0, S) =$	$S' \to S\cdot$
$I_2 = goto(I_0, A) =$	$S \to A\cdot$	$I_3 = goto(I_0, B) =$	$S \to B \cdot bb$
$I_4 = goto(I_0, a) =$	$A \to a \cdot B$ $A \to a \cdot Ab$ $B \to \cdot aAb$ $B \to \cdot b$ $A \to \cdot aB$	$I_5 = goto(I_0, b) =$	$B \to b\cdot$
$I_6 = goto(I_3, b) =$	$B \to Bb \cdot b$	$I_7 = goto(I_4, B) =$	$A \to aB\cdot$
$I_8 = goto(I_4, A) =$	$A \to aA \cdot b$	$I_9 = goto(I_6, b) =$	$B \to Bbb\cdot$
$I_{10} = goto(I_8, b) =$	$B \to aAb\cdot$	$goto(I_4, a) = I_4$ $goto(I_4, b) = I_5$	

We have that FOLLOW(S') = FOLLOW(S) = $\{\$\}$. And also FOLLOW$(A)$ = FOLLOW(B) = $\{b, \$\}$.
There are no conflicts in the states, thus the grammar is $SLR(1)$.

3) We can use the fact that the construction of the collection of the $LR(0)$ items corresponds to the definition of a DFA starting in state I_0. All the states are accepting and this automaton recognises all the viable prefixes. Thus, we can test if the string $aaAb$ is accepted. A labelled path for the string on the automaton is $0 \xrightarrow{a} 4 \xrightarrow{a} 4 \xrightarrow{A} 8 \xrightarrow{b} 10$. This means that the string is a viable prefix.

The theory also tells us that the $LR(0)$ items contained in the final state reached with the viable prefix are exactly all the items that are valid for it. Looking at the state I_{10}, the only valid item for the viable prefix is $B \rightarrow aAb\cdot$.