Department of Information and Computing Sciences
Utrecht University

INFOB3TC – Solutions for Exam 2

Johan Jeuring

Monday, 25 January 2016, 8:30–10:30

Please keep in mind that there are often many possible solutions and that these example solutions may contain mistakes.

Questions

Regular expressions, languages and pumping lemmas

1 (10 points)**.** Consider the DFA (X, Q, d, S, F) where $X = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$, $Q = \{q_1, q_2\}$, d is defined by:

$$\begin{aligned}
&d\ q_1\ \mathtt{a} = q_1\\
&d\ q_1\ \mathtt{b} = q_2\\
&d\ q_2\ \mathtt{a} = q_1\\
&d\ q_2\ \mathtt{c} = q_2
\end{aligned}$$

$S = q_1$, and $F = \{q_2\}$. Give the regular expression denoting the language accepted by this automaton. •

Solution 1.

$\mathtt{a}^*\mathtt{b}(\mathtt{a}^+\mathtt{b} + \mathtt{c})^*$

Marking
Any regular expression that generates the same language gives full points; an expression that generates either fewer or more sentences gives no points.

∘

For the following three tasks: consider the following three languages:

$$\begin{aligned}
L_1 &= \{a^n b^m c d^m e^n \mid n, m \geqslant 0\}\\
L_2 &= \{(ab)^n c d^m \mid n, m \geqslant 0\}\\
L_3 &= \{a^n b (cd)^n e^n \mid n \geqslant 0\}
\end{aligned}$$

2 (5 points)**.** One of the languages is regular, one context-free and not regular and one not context-free. Which are the regular and the non-regular context-free languages? •

Solution 2.
L_2 is regular, L_1 is context-free, L_3 is neither.

Marking
-2 per error. ○

3 (5 points)**.** Give a regular grammar for the regular language, and a context-free for the context-free language. •

Solution 3.

Here is a regular grammar for L_2:

$$\begin{aligned} S &\to abS \\ S &\to c \\ S &\to cB \\ B &\to d \\ B &\to dB \end{aligned}$$

Here is a context-free grammar for L_1:

$$\begin{aligned} S &\to aSe \\ S &\to T \\ T &\to bTd \\ T &\to c \end{aligned}$$

Marking
Each grammar gives 2.5 points; a grammar is either correct (of the correct form, generating the same language) or wrong. ○

4 (10 points)**.** Prove that the grammar that is context-free but not regular is indeed not regular by using the pumping lemma for regular languages . •

Solution 4.

The language L_1 is not regular. To prove it, we assume it is regular and find a contradiction using the pumping lemma.

For any n,
let $x = \varepsilon$, $y = a^n$, $z = b^m c d^m e^n$.
Then, $xyz = a^n b^m c d^m e^n \in L_1$ and $|y| \geqslant n$.

From the pumping lemma, we know there must be a loop in y, i.e. $y = uvw$ with $q = |v| > 0$ such that $xuv^iwz \in L_1$ for all $i \in \mathbb{N}$.

Let $i = 2$. We expect $xuv^2wz \in L_1$. If $u = a^s$, $v = a^q$, $w = a^t$, then we get $a^{s+2\,q+t}b^m cd^m e^n = a^{n+q}b^m cd^m e^n \in L$. But this word is not in L, since $q > 0$. Therefore, L_1 is not regular.

Marking

Proof with errors: -2.5

Correct reasoning but no formal proof: -5

Some argument that deserves merit, but by far correct: -7.5

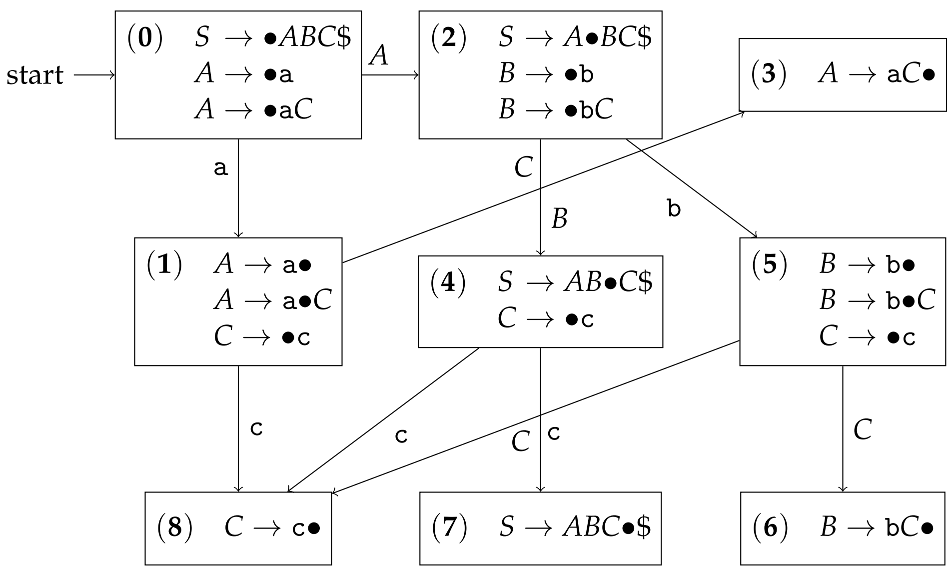
LR parsing

Consider the following grammar:

$$
\begin{aligned}
S &\to ABC\$ \\
A &\to \mathtt{a} \\
A &\to \mathtt{a}C \\
B &\to \mathtt{b} \\
B &\to \mathtt{b}C \\
C &\to \mathtt{c}
\end{aligned}
$$

5 (10 points)**.** This grammar is not LR(0). Construct the LR(0) automaton for this grammar, and show which conflicts appear where.

Solution 5.



States (**1**) and (**5**) have a shift/reduce conflict.

Marking

7.5 points for the correct automaton, 2.5 points for pointing out the correct conflicts.
Errors in closures: -2.5
No transitions with nonterminals: -5

○

6 (10 points)**.** Is this grammar SLR(1)? If so, construct the SLR-table. If not, explain where you cannot make a choice in a shift/reduce conflict or a reduce/reduce conflict. ●

Solution 6. This grammar is not SLR(1). The follow symbol of A is `b`, so the conflict in state $(\mathbf{1})$ can be resolved: shift if you see a `c` in the input, reduce if you see a `b`. The follow symbol of B is `c`, so the conflict in state $(\mathbf{5})$ cannot be resolved.

Marking

The argument doesn't mention 'Follow': -5

○

7 (10 points)**.** Play through the LR parsing process for the sentence "`acbcc$`". If there is a choice somewhere, make this explicit. Show in each step at which state in your LR(0) automaton you are. ●

Solution 7.

stack	input	remark
$(\mathbf{0})$	`acbcc$`	shift
$(\mathbf{0})\mathtt{a}(\mathbf{1})$	`cbcc$`	shift (choice)
$(\mathbf{0})\mathtt{a}(\mathbf{1})\ \mathtt{c}(\mathbf{8})$	`bcc$`	reduce by $C \to \mathtt{c}$
$(\mathbf{0})\mathtt{a}(\mathbf{1})\ C(\mathbf{3})$	`bcc$`	reduce by $A \to \mathtt{a}C$
$(\mathbf{0})A(\mathbf{2})$	`bcc$`	shift
$(\mathbf{0})A(\mathbf{2})\ \mathtt{b}(\mathbf{5})$	`cc$`	shift (choice)
$(\mathbf{0})A(\mathbf{2})\ \mathtt{b}(\mathbf{5})\ \mathtt{c}(\mathbf{8})$	`c$`	reduce by $C \to \mathtt{c}$
$(\mathbf{0})A(\mathbf{2})\ \mathtt{b}(\mathbf{5})\ C(\mathbf{6})$	`c$`	reduce by $B \to \mathtt{b}C$
$(\mathbf{0})A(\mathbf{2})\ B(\mathbf{4})$	`c$`	shift
$(\mathbf{0})A(\mathbf{2})\ B(\mathbf{4})\ \mathtt{c}(\mathbf{8})$	$	reduce by $C \to \mathtt{c}$
$(\mathbf{0})A(\mathbf{2})\ B(\mathbf{4})\ C(\mathbf{7})$	$	shift
$(\mathbf{0})A(\mathbf{2})\ B(\mathbf{4})\ C(\mathbf{7})\$$		reduce by $S \to ABC\$$
S		accept

Marking

No choices mentioned in an otherwise correct derivation: -1
States from the automaton are not given: -2

○

LL parsing

In these exercises we will look at the grammar

$$S \to AB$$
$$A \to \mathtt{a}A\mathtt{a} \mid \varepsilon$$
$$B \to \mathtt{b}B\mathtt{b} \mid \varepsilon$$

8 (15 points)**.** Complete the table below by computing the values in the columns for the appropriate rows. Use *True* and *False* for property values and set notation for everything else.

NT	Production	*empty*	*emptyRhs*	*first*	*firstRhs*	*follow*	*lookAhead*
S							
	$A \to AB$						
A							
	$A \to \mathtt{a}A\mathtt{a}$						
	$A \to \varepsilon$						
B							
	$B \to \mathtt{b}B\mathtt{b}$						
	$B \to \varepsilon$						

•

Solution 8.

NT	Production	*empty*	*emptyRhs*	*first*	*firstRhs*	*follow*	*lookAhead*
S		True		$\{\mathtt{a},\mathtt{b}\}$		$\{\}$	
	$S \to AB$		True		$\{\mathtt{a},\mathtt{b}\}$		$\{\mathtt{a},\mathtt{b}\}$
A		True		$\{\mathtt{a}\}$		$\{\mathtt{a},\mathtt{b}\}$	
	$A \to \mathtt{a}A\mathtt{a}$		False		$\{\mathtt{a}\}$		$\{\mathtt{a}\}$
	$A \to \varepsilon$		True		$\{\}$		$\{\mathtt{a},\mathtt{b}\}$
B		True		$\{\mathtt{b}\}$		$\{\mathtt{b}\}$	
	$B \to \mathtt{b}B\mathtt{b}$		False		$\{\mathtt{b}\}$		$\{\mathtt{b}\}$
	$B \to \varepsilon$		True		$\{\}$		$\{\mathtt{b}\}$

Marking

-1 per error (if many correct), or, alternatively, +1 per correct answer (if many missing or incorrect). ○

9 (10 points)**.** Is the above grammar LL(1)? Explain how you arrived at your answer. If the grammar is not LL(1), give a grammar that generates the same language and is LL(1). •

Solution 9.

The above grammar is not LL(1) because the *lookAhead* sets of the A and B productions have a non-empty intersection. The following grammar generates the same language and is LL(1).

$$
\begin{aligned}
S &\to AB \\
A &\to \mathtt{aa}A \mid \varepsilon \\
B &\to \mathtt{bb}B \mid \varepsilon
\end{aligned}
$$

Since $\mathit{Follow}\ (A) = \{\mathtt{b}\}$, and $\mathit{Follow}\ (B) = \{\ \}$, the intersections of lookahead sets of the productions for A and B, respectively, are empty.

Marking

Not LL(1): 2 points, correct argument: 4 (using the wrong terminology: -1), and a correct LL(1) grammar: 4 points. ○

10 (5 points)**.** Show the steps that a parser for the above LL(1) grammar goes through to recognize the following input sequence:

`aabb`

For each step (one per line), show the stack, the remaining input, and the action (followed by the relevant symbol or production) performed. If you reach a step in which you cannot proceed, note the action as "error." ●

Solution 10.

stack	**input**	**action**
S	`aabb`	initial state
AB	`aabb`	expand S
`aa`AB	`aabb`	expand A
AB	`bb`	match (2×)
B	`bb`	expand A
`bb`B	`bb`	expand B
B	ε	match (2×)
ε	ε	expand B

Marking

Wrong terminology: -1

○