

Exercise 1 (Classification):

(3 Points)

In the lecture you have seen different categorisations of CFGs.

- a) Provide a language that is recognised by a $LL(1)$ grammar for which there does not exist any $LL(0)$ grammar.
- b) Show there are regular languages for which no $LR(0)$ grammars exist.
- c) Show there is an $LR(0)$ grammar which is not an $LL(1)$ grammar. *Note, here we do not ask anything about their languages.*

Lösung: _____

- a) The languages recognised by $LL(0)$ grammars are singleton sets. Thus, the language $\{a, b\}$ is recognised by the $LL(1)$ grammar

$$S \rightarrow a \mid b$$

but not by any $LL(0)$ grammar.

- b) We know that language recognised by $LR(0)$ grammars have prefix property (refer exercise series 5). Hence, any regular language, which violates the prefix property cannot be recognised by a $LR(0)$. E.g. $\{a, ab\}$.

- c) Consider the following grammar G :

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow a A b \\ B &\rightarrow a a B c \end{aligned}$$

The figure 1. shows the automaton accepting viable prefixes of G . It has no conflict. The grammar G is not $LL(1)$, since $la(S \rightarrow A) \cap la(S \rightarrow B)$ is not empty.

_____.

Exercise 2 (LR(1) and LALR(1)):

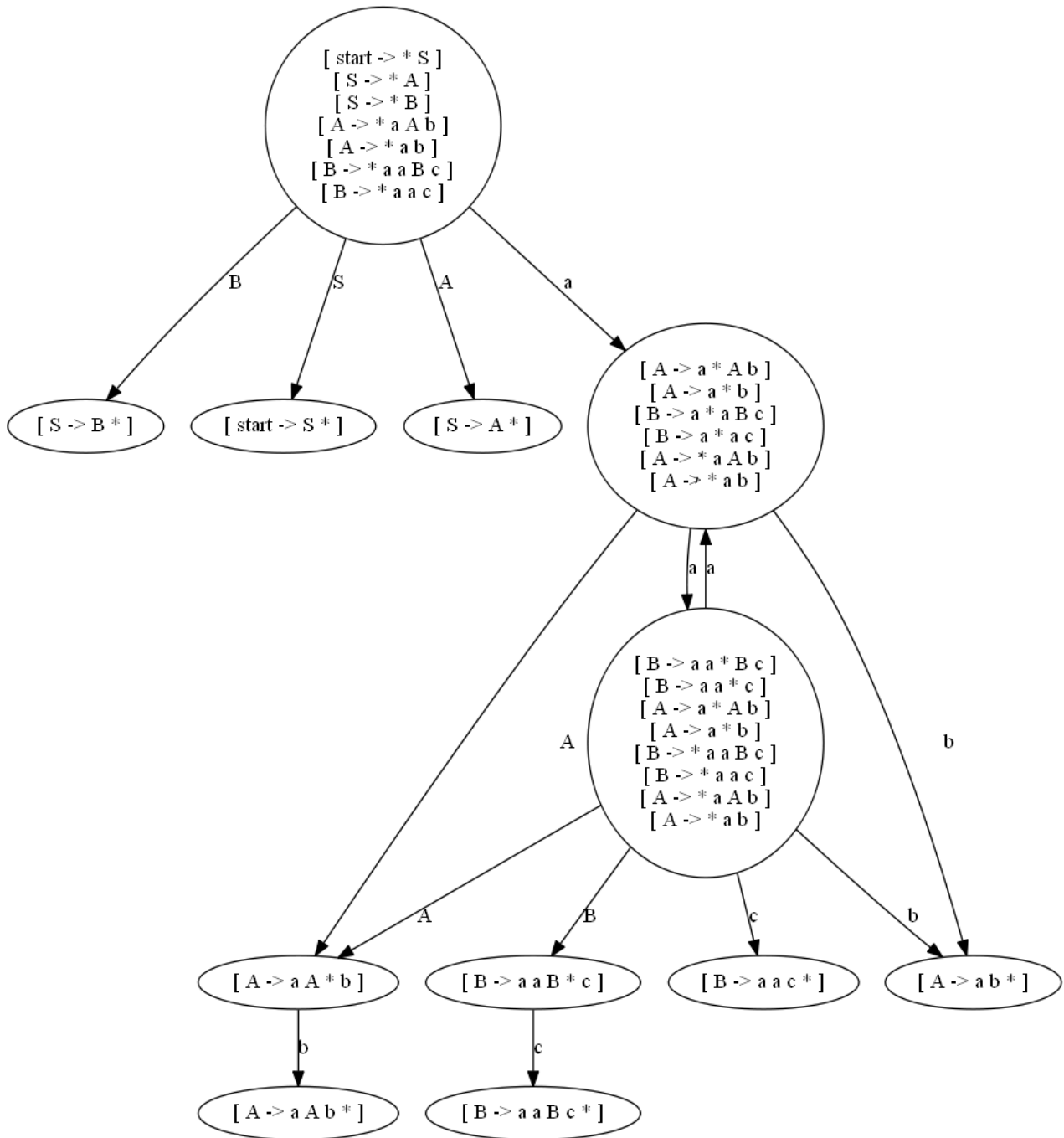
(4 Points)

Let grammar G be given by:

$$\begin{aligned} S' &\rightarrow S \\ S &\rightarrow Aa \mid bAc \mid Bc \mid bBa \\ A &\rightarrow d \\ B &\rightarrow d \end{aligned}$$

- a) Check whether $G \in LR(1)$ by computing the $LR(1)$ -sets of G .
- b) Is $G \in LALR(1)$? Justify your answer.

Lösung: _____

Abbildung 1: Automaton accepting viable prefixes of G .

$$\begin{aligned}
LR(1)(\epsilon) &= \{[S' \rightarrow \cdot S, \epsilon], \\
&\quad [S \rightarrow \cdot Aa, \epsilon], [S \rightarrow \cdot bAc, \epsilon], [S \rightarrow \cdot Bc, \epsilon], [S \rightarrow \cdot bBa, \epsilon], \\
&\quad [A \rightarrow \cdot d, a], [B \rightarrow \cdot d, c]\} \\
LR(1)(S) &= \{[S' \rightarrow S \cdot, \epsilon]\} \\
LR(1)(A) &= \{[S \rightarrow A \cdot a, \epsilon]\} \\
LR(1)(B) &= \{[S \rightarrow B \cdot c, \epsilon]\} \\
LR(1)(b) &= \{[S \rightarrow b \cdot Ac, \epsilon], [S \rightarrow b \cdot Ba, \epsilon], \\
&\quad [A \rightarrow \cdot d, c], [B \rightarrow \cdot d, a]\} \\
\text{a) } LR(1)(d) &= \{[A \rightarrow d \cdot, a], [B \rightarrow d \cdot, c]\} \\
LR(1)(bd) &= \{[A \rightarrow d \cdot, c], [B \rightarrow d \cdot, a]\} \\
LR(1)(Aa) &= \{[S \rightarrow Aa \cdot, \epsilon]\} \\
LR(1)(Bc) &= \{[S \rightarrow Bc \cdot, \epsilon]\} \\
LR(1)(bA) &= \{[S \rightarrow bA \cdot c, \epsilon]\} \\
LR(1)(bB) &= \{[S \rightarrow bB \cdot a, \epsilon]\} \\
LR(1)(bAc) &= \{[S \rightarrow bAc \cdot, \epsilon]\} \\
LR(1)(bBa) &= \{[S \rightarrow bBa \cdot, \epsilon]\}
\end{aligned}$$

$G \in LR(1)$? Yes!

- No shift/reduce conflict.
- No reduce/reduce conflict as in reduce sets, rules are separated by lookahead symbol.

b) $G \in LALR(1)$? No!

$$LR(1)(d) \stackrel{LR(0)}{\sim} LR(1)(bd)$$

$\Rightarrow LR(1)(d) \cup LR(1)(bd) = \{[A \rightarrow d \cdot, a/c], [B \rightarrow d \cdot, a/c]\}$ is $LALR(1)$ information

$\Rightarrow$ reduce/reduce conflict

Exercise 3 (LR(0) parser):

(3 Points)

Previously, you have generated the $LR(0)$ sets which constitute the states of the so called goto-automaton. Using this, we can implement our first parser which will be an $LR(0)$ parser. Please use the framework that can be found along with this sheet on our web page.

- Implement the method `Parser.parse(List<Symbol>)`

In the previous exercise, we have seen that the grammar of our WHILE language is not an $LR(0)$ grammar. Therefore we have provided simpler grammars for testing your implementation.

Lösung: _____