Department of Information and Computing Sciences
Utrecht University

INFOB3TC – Final exam 2021

Jurriaan Hage

Thursday 4th of February 2021, 15.15-18.15

Preliminaries

- The exam consists of 10 pages (including this page). Please verify that you got all the pages.
- Fill out the answers **on the exam itself**.
- Write your **name** and **student number** here:

- For open questions, the maximum score is stated at the top of each question. The total amount of points you can get is 90. You obtain your grade by dividing by that number and multiplying by 10, rounded to one digit after the comma.
- Try to give simple and concise answers. Write readable text. You may use Dutch or English.
- When writing grammar and language constructs, you may use any set, sequence, or language operations covered in the lecture notes.

Good luck!

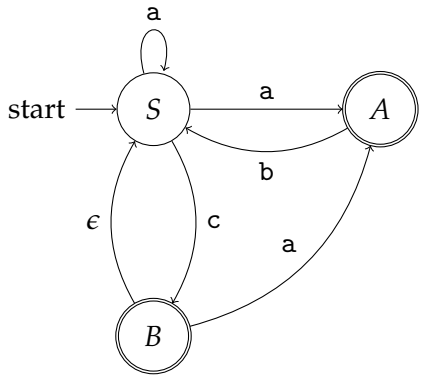
1. Given is the following left-recursive grammar.

$S \rightarrow aSb$
$S \rightarrow Sba$
$S \rightarrow A$
$A \rightarrow Aa$
$A \rightarrow bS$
$A \rightarrow c$

Construct the grammar in which left-recursion is removed. •

. . . / **10**

2. Consider the following NFA, with start state S, terminals $\{\texttt{a}, \texttt{b}, \texttt{c}\}$, and final state A and B.



Construct a DFA (Deterministic Finite Automaton) that accepts the same language. You should use the subset construction, but should exclude states that are unreachable from the start state or that can never lead to a final state. It is *essential* that in your answer it is clear what each of your DFA states corresponds to in the original NFA; do not forget to indicate the start and end states of the DFA. •

. . . /**10**

An AVL tree is a classical data structure, designed in 1962 by Georgy Adelson-Velsky and Evgenii Landis. In an AVL tree, the heights of the two child subtrees of any node differ by at most one; if at any time they differ by more than one, rebalancing is done to restore this property. The datatype *AVL* can defined as follows (note the invariants given for the constructors in comments).

```
data AVL e = E                          — Empty Tree
           | N (AVL e) e (AVL e)        — left height = right height + 1
           | Z (AVL e) e (AVL e)        — right height = left height
           | P (AVL e) e (AVL e)        — right height = left height + 1
```

3. Give the algebra type for the datatype *AVL e*. •

. . . /**4**

4. Now, give the type and the definition of the fold function associated with the datatype *AVL e*. •

. . . /**4**

5. The height of an *AVL* tree is an essential concept in *AVL* trees; we note the empty AVL tree has height zero. Implement the function *heightAVL* **efficiently** as a *foldAVL* **making use of the invariants given for the constructors**? •

. . . /**4**

As a reminder, the following formulation of the pumping lemma proof strategy is provided:
For *every* natural number n,
find a word xyz in L with $|y| \geqslant n$ (*you choose* the word),
such that for *every* splitting $y = uvw$ with $|v| > 0$,
there exists a number i (*you figure out* the number),
such that $xuv^iwz \notin L$ (you have to *prove* it).

6. Given is the following language $R = \{a^n b^{m+n} \mid n, m \geqslant 1\}$. Prove with the Pumping Lemma for Regular Languages that R is not *regular*. •

. . . /**12**

7. Given is the following regular expression:

$$ba(((ba)^* + aa)^* + a)^*b$$

Give a finite state automaton that generates exactly the language of sentences described by this regular expression.

. . . /**8**

8. Suppose we have an nfa with a finite set of states Q and alphabet X. And suppose that while drawing its diagram we observe the following following properties:

- it has one start state,
- it has one end state, and
- for all states $q \in Q$ and labels $v \in X \cup \{\epsilon\}$, q has at most one outgoing edge labelled with v.

Is this nfa necessarily a dfa? Explain your answer.

. . . /**6**

9. Construct the $LR(0)$ automaton for the following grammar:

$S' \to S\,\$$
$S \to aA$
$S \to Sba$
$A \to bS$
$A \to c$

•

. . . /**10**

10. List all the conflicts in this $LR(0)$ automaton, and indicate which kind of conflict each is. •

. . . /**4**

11. Is the use of 1-lookahead in the SLR(1) parser enough to resolve the conflicts in the $LR(0)$ automaton? Argue using the follow sets of the non-terminals why this is the case. •

. . . /**4**

12. Construct a pushdown automaton for the language over the alphabet $\{a, b\}$

$$L = \{w \mid \text{nr of } a\text{'s in } w \text{ is one more or one less than the nr of } b\text{'s in } w\}$$

It may, but need not be deterministic.

. . . /8

13. Suppose we have a context-free grammar $G_1 = (T_1, N_1, R_1, S_1)$. We now define

$$G = (T_1, N_1 \cup \{S\}, R_1 \cup \{S \to S_1 S_1\}, S)$$

where $S \notin N_1$. Describe the language of G in terms of the language generated by G_1?

. . . / **6**