Universität Zürich
Institut für Informatik

Student Name:
Student Number:

Autumn 2020
Sven Seuken
Dennis Komm

Foundations of Computing II
Assignment 2

Deterministic and Non-deterministic Automata and Regular Languages

Distributed: 05.10.2020 – Due Date: 25.10.2020

Upload your solutions to the OLAT system.

2.1 Deterministic Finite Automata

For a word w, we denote the number of occurrences of the letter a in w by $|w|_a$; for instance, we have $|1010110|_0 = 3$ and $|xyzxzzy|_x = 2$. Draw DFAs for the following languages.

a) $L_1 = \{w \in \{a,b\}^* \mid |w|_a + |w|_b \text{ is even}\}$,

b) $L_2 = \{w \in \{a,b\}^* \mid (|w|_a + 2|w|_b) \bmod 4 = 3\}$,

c) $L_3 = \{w \in \{a,b,c\}^* \mid (|w|_a + 2|w|_c) \bmod 5 = 0\}$.

2.2 Size of Deterministic Finite Automata

Consider the language

$$L = \{w \in \{a,b\}^* \mid w = xay \text{ with } x \in \{a,b\}^* \text{ and } y \in \{a,b\}^2\},$$

that is, the language that contains all words over the alphabet $\{a,b\}$ which have at least three letters and an a at the third-to-last position. Prove that any DFA for L has to have at least four states in the following way.

Take a set of four words (that are not necessarily in L) and show that no two of them are allowed to end in the same state of any DFA for L. To this end, for any two words w_1 and w_2, supply a *shortest suffix* that implies that one of the two words has to end in an accepting state while the other one must not end in an accepting state.

2.3 Non-Deterministic Finite Automata

a) Construct an NFA for the following languages.

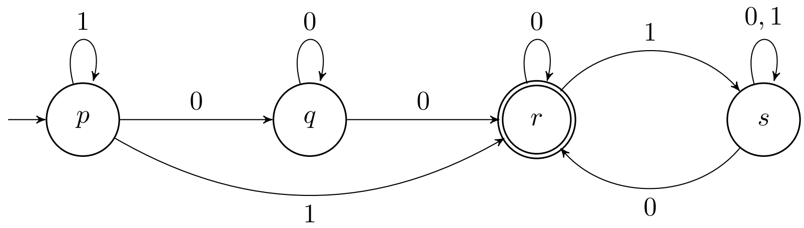
(i) $L_1 = \{w \in \{0,1,2\}^* \mid w \text{ contains the string } 111\}$,

(ii) $L_2 = \{w \in \{0,1,2\}^* \mid w \text{ ends with the string } 111\}$,

(iii) L_3, which is matched by the regular expression $(1+0)^*1(0+1)11(0+1)(0+1)00$.

Hint: Here, you do not have to use the construction from the lecture.

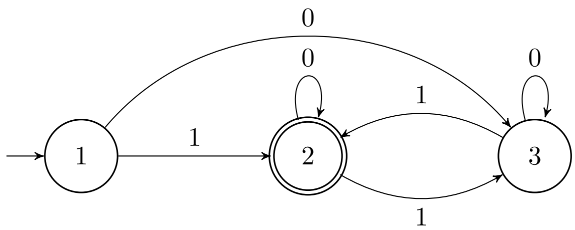
b) Consider the following NFA.



Use the powerset construction to transform this NFA into a DFA.

2.4 Finite Automata and Regular Expressions

a) Use the method that was introduced in the lecture to transform the following DFA into a regular expression.



Write down all steps and comment on what you are doing.

b) Convert the regular expression $1 + (0 + 1)^* + 0$ into an NFA with ε-transitions.

c) Convert the regular expression $01^* + 1$ into an NFA with ε-transitions.

2.5 The Product Automaton

Consider the two languages

$$L_{10} = \{w \in \{0,1\}^* \mid w \text{ contains } 10\} \text{ and}$$
$$L_{011} = \{w \in \{0,1\}^* \mid w \text{ starts with } 011\}.$$

Construct the product automaton for $L_{10} \cap L_{011}$ with the technique described in the lecture.

2.6 Non-Regularity

a) In the last assignment (more specifically, exercise 1.2(b)iv), we already sketched that there is a problem when designing a DFA for the language $L_{01} = \{0^k 1^k \mid k \in \mathbb{N}\}$. Now use the pumping lemma to prove that this language is indeed not regular.

b) Again, using the pumping lemma, prove that the language $L_{\text{sq}} = \{0^k \mid k \text{ is a square}\}$ is not regular.