Homework I, Theory of Computation 2022

Submission: **The deadline for Homework 1 is 23:59 on 24 March.** Please submit your solutions on Moodle. Typing your solutions using a typesetting system such as LaTeX is strongly encouraged! If you must handwrite your solutions, write cleanly and with a pen. Messy and unreadable homeworks will not be graded. No late homeworks will be accepted.

Writing: Please be precise, concise and (reasonably) formal. Keep in mind that many of the problems ask you to provide a proof of a statement (as opposed to, say, just to provide an example). Therefore, make sure that your reasoning is correct and there are no holes in it. A solution that is hard/impossible to decipher/follow might not get full credit (even if it is in principle correct). You do not need to reprove anything that was shown in the class—just state clearly what was proved and where.

Collaboration: These problem sets are meant to be worked on in groups of 3–5 students. Please submit only one writeup per team—it should contain the names of all the students. You are strongly encouraged to solve these problems by yourself. If you must, you may use books or online resources to help solve homework problems, but you must credit all such sources in your writeup and you must never copy material verbatim. Even though only one writeup is submitted, it is expected that each one of the team members is able to fully explain the solutions if requested to do so.

Grading: Each of the two problems will be graded on a scale from 0 to 5.

Warning: Your attention is drawn to the EPFL policy on academic dishonesty. In particular, you should be aware that copying solutions, in whole or in part, from other students in the class or any other source without acknowledgement constitutes cheating. Any student found to be cheating risks automatically failing the class and being referred to the appropriate office.

Homework 1

1 Pick either one of the automata below (Figure 1 or 2) and find a description of the language it recognizes. Prove by induction (or otherwise) that your description is correct. (*Note: You will be awarded full points for analysing just one of the below automata—you choose which one!*)

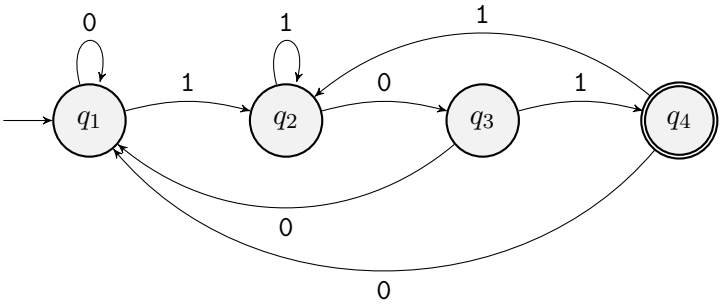

Figure 1. Original harder variant

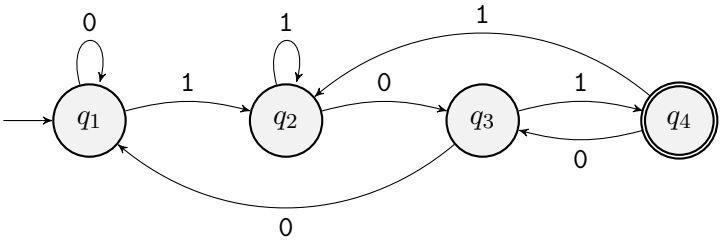

Figure 2. New easier variant

Solution: Figure 1 (Old harder variant)

Answer: the strings that the automaton recognizes are those that finish with odd number of pairs of symbols 10 and 1 after them ($(10)^{2k+1}1$, where k is any integer).

To prove it, we make a stronger statement that describes a structure of a string w that make automaton to end in a certain state:

- q_1: w doesn't end with 1 and ends with $(10)^{2k}$.
- q_2: w ends with $(10)^{2k}1$.
- q_3: w ends with $(10)^{2k+1}$.
- q_4: w ends with $(10)^{2k+1}1$.

Here k can be any non-negative integer, even 0.

To prove this statement, we use induction by length of the string. The base is obvious - the empty string finishes in q_1 and indeed has $(10)^{2k}$ in the end, where $k = 0$.

To prove the inductive step, we consider that for any string w the statement holds and prove that for any symbol a, for wa the statement will also hold. To make it we consider in case by case basis where could automaton finish after reading w.

- q_1 - then by induction w ends with $(10)^{2k}$. If $a = 0$ then wa is of type$(10)^{2k}$, because finishes with 00, and indeed brings automaton back to q_1. If $a = 1$ then wa is of type$(10)^{2k}1$, and indeed brings automaton to q_2.

- q_2 - then by induction w ends with $(10)^{2k}1$. If $a = 0$ then wa is of type$(10)^{2k+1}$, and this corresponds to description of q_3. If $a = 1$ then wa is of type$(10)^{2k}1$, where $k = 0$ and indeed brings automaton to q_2.

- q_3 - then by induction w ends with $(10)^{2k+1}$. If $a = 0$ then wa is of type$(10)^{2k}$, because finishes with 00, and indeed brings automaton back to q_1. If $a = 1$ then wa is of type$(10)^{2k+1}1$, where $k = 1$ and indeed brings automaton to q_4.

- q_4 - then by induction w ends with $(10)^{2k+1}1$. If $a = 0$ then wa is of type$(10)^{2k}$, and indeed brings automaton back to q_1. If $a = 1$ then wa is of type$(10)^{2k}1$, where $k = 0$ and indeed brings automaton back to q_2.

This concludes the proof of induction and the solution of the problem.

Solution: Figure 2 (New easier variant)

Answer: the strings that the automaton recognises are those and only those that finish with 101.

First, we show that strings that finish with 101 are recognized by the automaton. We consider all the states where automaton can be before having 101 substring in the end, and show that after reading 101 it anyway comes to q_4.

- q_1 - then substring 101 will make automaton to be in states $q_1 \to q_2 \to q_3 \to q_4$

- q_2 - then 101 will make automaton to be in states $q_1 \to q_2 \to q_3 \to q_4$

- q_3 - then 101 will make automaton to be in states $q_1 \to q_2 \to q_3 \to q_4$

- q_4 - then 101 will make automaton to be in states $q_1 \to q_2 \to q_3 \to q_4$

Now, we show if automaton ends up in q_4 after reading the string, then the last three symbols should have been 101. To reach q_4 we can only come from q_3 by reading 1. q_3 can be reached only from q_2 or q_4, but in both cases only by reading 0. Finally, to reach q_2 or q_4 we only can come by 1. Q.e.d.

2 Closure properties:

2a For a language $L \subseteq \Sigma^*$ define its *triple* by

$$L^3 \;\coloneqq\; \{www : w \in L\}.$$

Show that regular languages are *not* closed under tripling. That is, find some regular language L such that L^3 is not regular (and prove that it is not).

2b Suppose that $L \subseteq \Sigma^*$ is a regular language over a *unary alphabet*, $|\Sigma| = 1$. Show that in this case L^3 is regular.

(It suffices that you describe a finite automaton for L^3. You do ***not*** *need to prove correctness via induction.)*

Solution:

2a We use the pumping lemma to show that L^3 is not regular. Consider

$$L \coloneqq \{w : w \in \Sigma^*\}.$$

Assume L^3 is regular. We pick a string $s = a^p b^p a^p b^p a^p b^p \in L^3$, and show this can't be pumped to arrive at a contradiction. For every xyz decomposition of s such that, $|xy| \leq p$ and $|y| \geq 1$, we see that y would be of the form a^k where $k \geq 1$. Subsequently, $xy^2z = a^{k+p} b^p a^p b^p a^p b^p$ which clearly $\notin L^3$.

2b Since the signature Σ is unary (assume a), the language L just captures a subset of non-negative integers S (through the length of the unary string), and L^3 needs to capture $S^3 = \{3k, k \in S\}$. If the DFA for L is given by $(Q, \Sigma, \delta, q_1, F_1)$, the DFA for L^3 is given by $(Q', \Sigma, \delta', q_1, F_1)$. where we add auxiliary state sets Q_1, Q_2 as follows

$$Q' = Q \cup \{q_i' : q_i \in Q\}_{Q_1} \cup \{q_i'' : q_i \in Q\}_{Q_2}$$

$$\delta'(q_i, a) = q_i', \text{ where } q_i \in Q$$

$$\delta'(q_i', a) = q_i'', \text{ where } q_i' \in Q_1$$

$$\delta'(q_i'', a) = \delta(q_i, a), \text{ where } q_i'' \in Q_2$$

So by definition of this new δ', for every transition taken using the old δ from q_i to $\delta(q_i, a)$, we take exactly three transitions starting from q_i to $\delta(q_i, a)$. Thus for every string captured by L, we take the same path through Q in L^3, but taking thrice as many transitions making the string length triple.