Computational Models – Exercise 2

Due Wednesday, 5 April 2023

Each student must solve the problems on their own. If you encounter difficulties, you may ask a classmate for a hint or the general idea. However, detailed discussion, note-taking, or sharing of written solutions is not allowed. Do not write down your answers while communicating with other people or show the answers for feedback.

Our grading app has severe limitations, such as no zoom tool. To make sure we can grade your work, please follow these technical guidelines:

Submit a **single PDF file** through Moodle.

The file size is limited to **10 MB**. If necessary, google *reduce PDF file size*.

Fill in your answers **on this form*** in the allocated spaces. The space provided gives you an indication of the expected length and level of detail of the answer. You may add a little more space if you need.

Include everything from this form in your submission. In particular, **include the problem statements**. Do not delete any text or omit pages, just add your answers.

Ensure your answers are **legible** (easy to read) at zoom 100% on a standard computer screen. Your text should be **large**, **sharp**, and in **high contrast** with the background.

Do not squeeze scanned solutions to fit in the space, as the text will become small.

Verify that pages are properly **ordered** and **oriented**.

The page size must be **A4**. Before submitting your file, check its page size using Acrobat Reader: go to File > Properties > Description and confirm that Page Size is around 21 × 29 cm. Note that scanning A4 pages does not guarantee the resulting page size will be A4, due to scaling. If necessary, google *resize PDF to A4*.

Do not add your answers as PDF comments. If you can drag them in Acrobat Reader, they are comments. If necessary, google *flatten PDF*.

A **5-point bonus** will be given to solutions typed in a word processor. Hand-sketched illustrations or diagrams will not deny you this bonus.

If there are technical issues with your submission, you may receive a fine. In extreme cases, your submission may not be graded at all.

If you need help or have questions, please use the course forum at Piazza.

*The only exception is in case you use LaTeX or a similar typesetting system. In that case, copy-paste everything from this file, except for illustrations or other hard-to-reproduce graphical elements. No need to fix corrupted formulas.

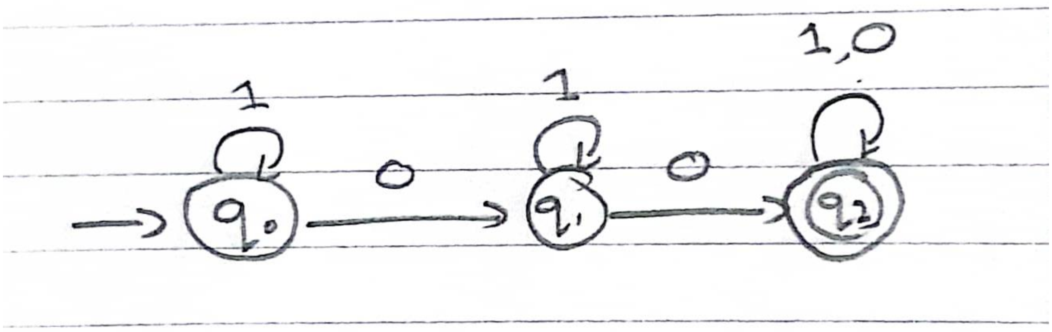
Worked with Jemma Diamond – 806839

Problem 1

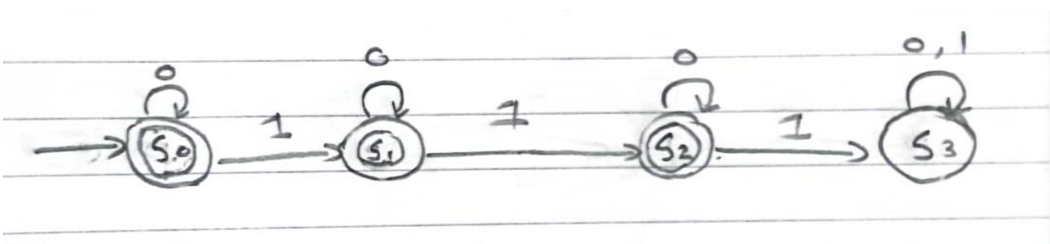
Let,

$$L_1 = \{w \in \{0,1\}^* | w\ contains\ at\ \mathbf{least}\ two\ '0's\}$$
$$L_2 = \{w \in \{0,1\}^* | w\ contains\ at\ \mathbf{most}\ two\ '1's\}$$

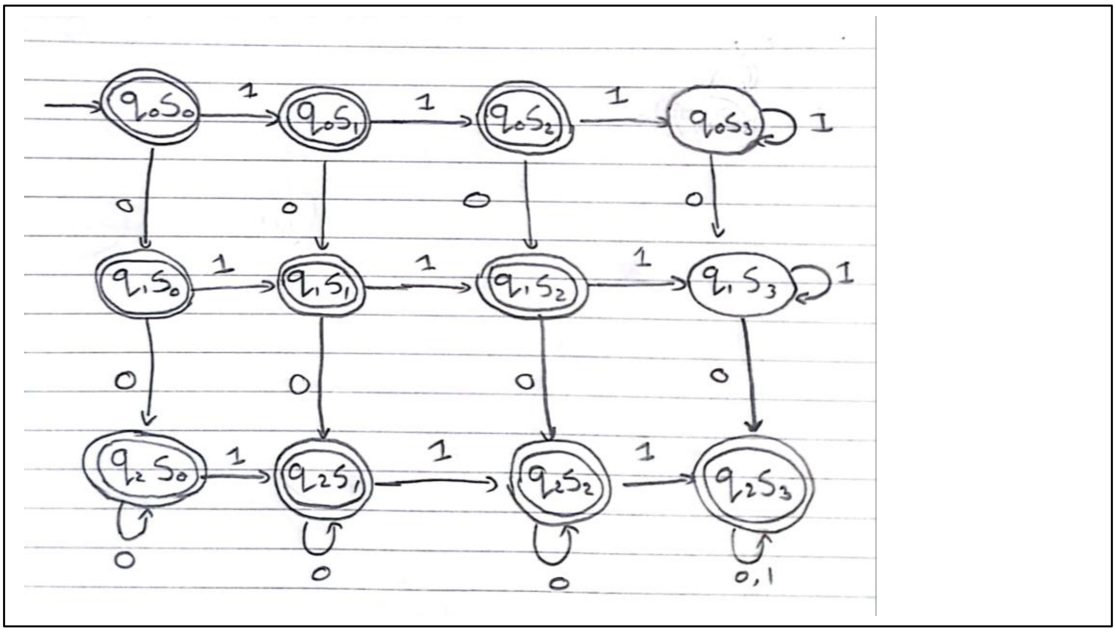
[6 pt] **1.** Draw a DFA (state diagram), M_1, whose language is L_1, that is $L(M_1) = L_1$.



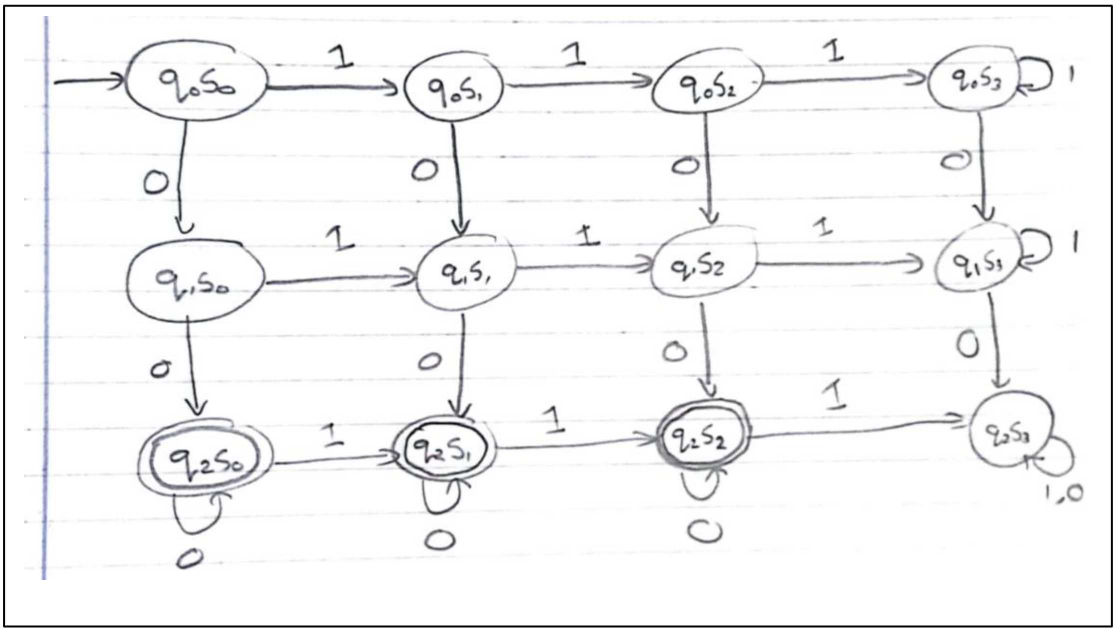
[6 pt] **2.** Draw a DFA (state diagram), M_2, whose language is L_2, that is $L(M_2) = L_2$.



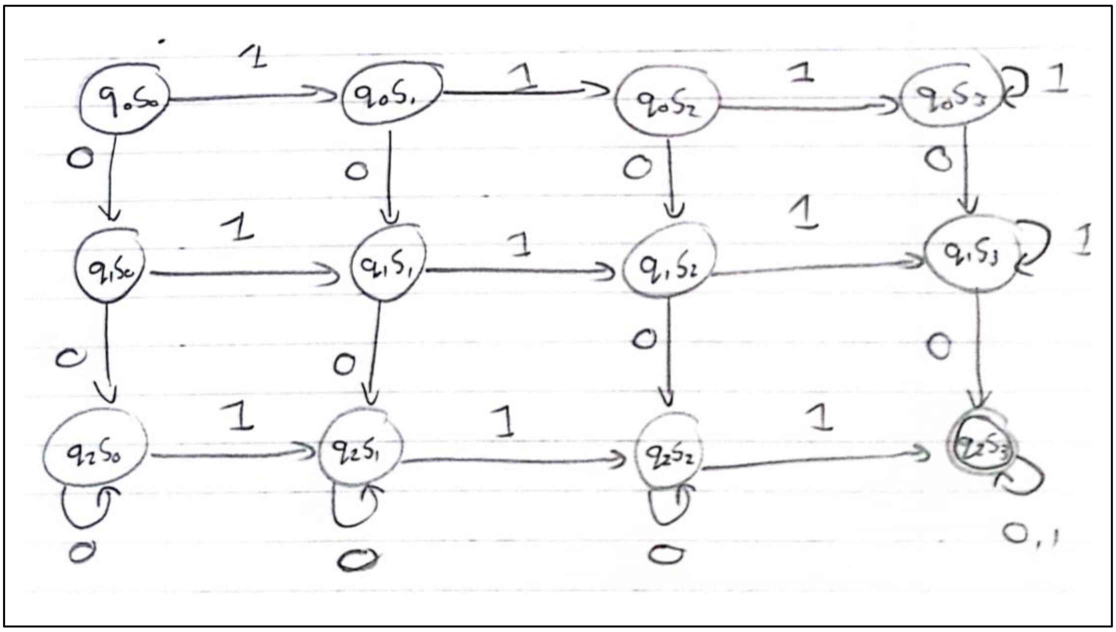
[10 pt] **3**. Draw a DFA (state diagram), $M_{1\cup 2}$, as taught in class (using a product automaton), whose language is $L_1 \cup L_2$, that is $L(M_{1\cup 2}) = L_1 \cup L_2$.



[10 pt] **4**. Draw a DFA (state diagram), $M_{1\cap 2}$, as taught in class (using a product automaton), whose language is $L_1 \cap L_2$, that is $L(M_{1\cap 2}) = L_1 \cap L_2$.

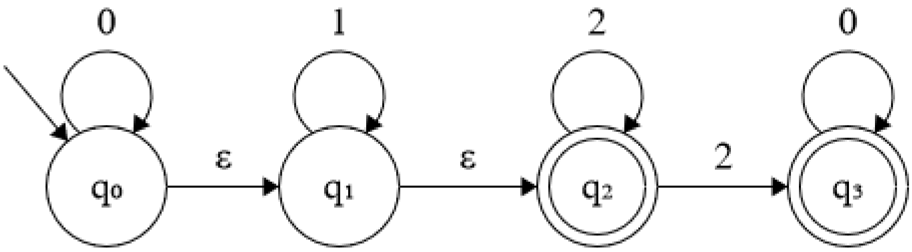


[10 pt] **5**. Draw a DFA (state diagram), $M_{1\setminus 2}$, as taught in class (using a product automaton), whose language is $L_1 \setminus L_2$, that is $L(M_{1\setminus 2}) = L_1 \setminus L_2$.



[6 pt] **Problem 2**

Let $N = (\{q_0, q_1, q_2, q_3\}, \{0,1,2\}, q_0, \delta, \{q_2, q_3\})$ be a non-deterministic finite automaton with epsilon transitions where δ is defined by the following state diagram:



What is $L(N)$? Briefly explain.

The language of words where all the 1's in the word comes before all the 2's in the word.

[20 pt] **Problem 3**

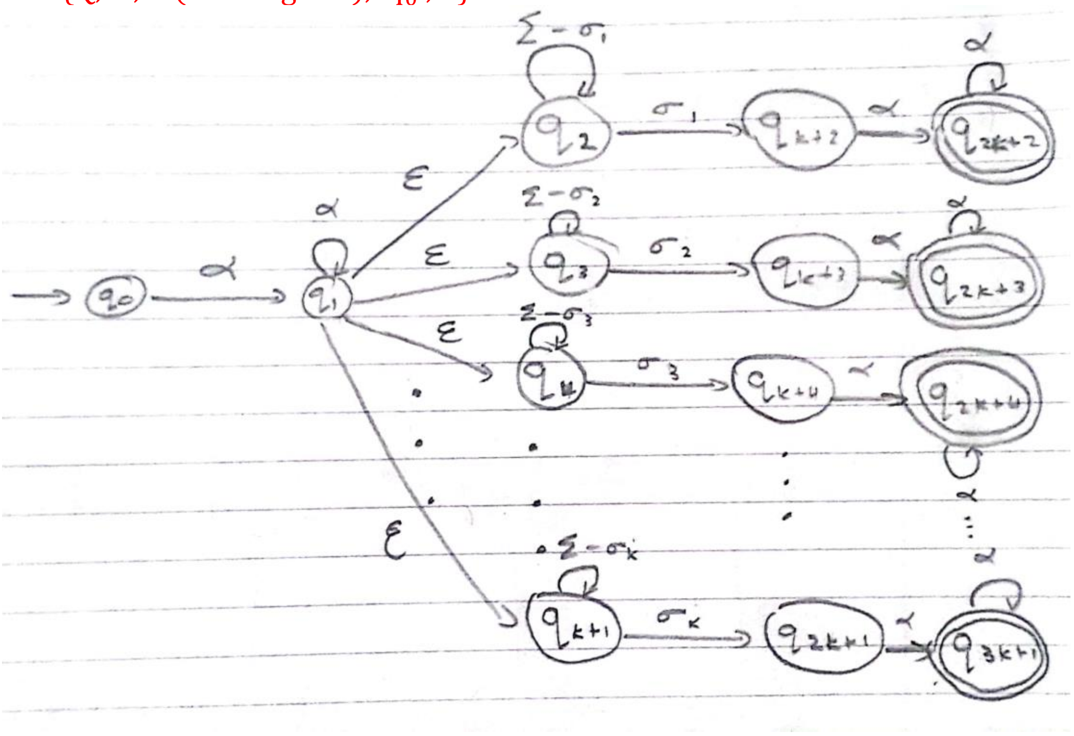
Let $\Sigma = \{\sigma_1, \dots, \sigma_k\}$ be an alphabet such that $k \geq 2$ and $\alpha \notin \Sigma$.

Construct a non-deterministic automaton (a description of the 5-tuple), N, for the following language L over the alphabet $\Sigma \cup \{\alpha\}$:

$$L = \{\alpha^{m_1} w_1 \ldots w_n \alpha^{m_2} | m_1, m_2, n \geq 1, w_1, \ldots w_n \in \Sigma \ and\ \forall 1 \leq i \leq n-1 : w_i \neq w_n\}.$$

δ can be described by a state diagram.

$Q = \{q_0, q_1, \ldots, q_{3k+1}\}$
$F = \{q_{2k+1}, \ldots, q_{3k+1}\}$
$N = \{Q, \Sigma, \delta \text{ (see diagram)}, q_0, F\}$

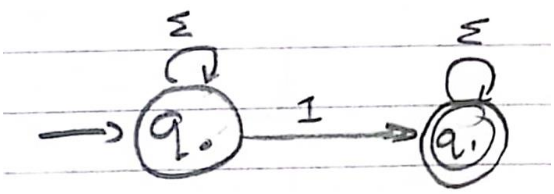


Problem 4

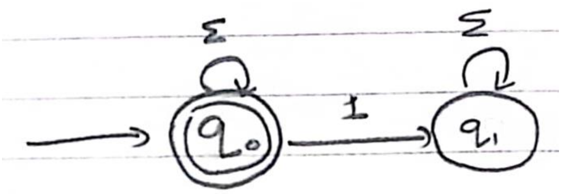
Prove / Disprove.

[6 pt] **1.** Let $N = (Q, \Sigma, q_s, \delta, F)$ a non-deterministic automaton without epsilon transitions and let $N' = (Q, \Sigma, q_s, \delta, Q \setminus F)$. It holds that $L(N') = \overline{L(N)}$.

False. Counterexample:
Let N = (Q = {q_0 , q_1} , Σ = {1,0} , q_0, δ (see diagram below), F = { q_1}) which is the language that accepts all words that contain a 1



Now the automaton for N', the language that should accept all words that do not contain a 1 would be:



However this automaton still accepts the string "11" through the path
q_0 (Initial state) → q_0 → q_0

[6 pt] **2.** Let L_1 and L_2 two languages over the same Σ. If L_1 is a regular language and $L_1 \cup L_2$ is a regular language, then L_2 is a regular language.

False. Counterexample:
Let L_1 = { w ∈ {a,b}* } which is a regular language
Let L_2 = { w ∈ {a,b}* | $\#_a(w) > \#_b(w)$ } which is an irregular language
Since $L_1 \subseteq L_2$, then $L_1 \cup L_2 = L_1$, which is a regular language

[20 pt] **Problem 5**

For some language L' we will define the following operation:

$$NotProperPrefix(L') = \{w \in L' | \nexists v \in L' \text{ such that } v \text{ is a proper prefix of } w\}.$$

Let L be a language over some Σ. Show that if L is a regular language then $NotProperPrefix(L)$ is a regular language.

Let the NFA that accepts L be M = $(Q, \Sigma, q_0, \delta, F)$. Create a new automaton M' s.t M' is two copies of M, M_0 and M_1, where all the accepting states of M_0 have epsilon transitions to their equivalent copies in M_1. Then make the accepting states of M' only the accepting states of the second copy M_1. Formally, M' is define as follows:
Q' = $\{q_{00}, q_{01}, \dots, q_{0k}, q_{10}, q_{11}, \dots, q_{1k}\}$ (where q_{0i} is states of M_0 and q_{1i} is states of M_1)
δ' is equivalent to δ_0 for M_0 and δ_1 for M_1, and adding the epsilon transitions
F' = F_1 from M_1
M' = (Q' , Σ , q_{00}, δ', F')

If an inputted word ends in an accepting state, we know that the word has a proper prefix, as it reached one of the original accepting states of M_0 (this would be the end of the prefix) and then aftewards reached an accepting state in the second copy, meaning that the overall inputted word is a valid word in L. Now take M' and create the equivalent DFA, and then reverse its accepting states to rejecting states and vice versa. This new DFA is now the compilement of M', i.e it accapets words of L that do not have a proper prefix. Since we have created a DFA that describes $NotProperPrefix(L')$, then $NotProperPrefix(L')$ is a regular language