Chair for Foundations of Software Reliability and Theoretical Computer Science
Informatik
Technical University of Munich

Eexam
Place student sticker here

Note:

- During the attendance check a sticker containing a unique code will be put on this exam.
- This code contains a unique number that associates this exam with your registration number.
- This number is printed both next to the code and to the signature field in the attendance check list.

Automaten und formale Sprachen

Exam:	IN2041 / Endterm	**Date:**	Thursday 17th February, 2022
Examiner:	Prof. Javier Esparza	**Time:**	11:00 – 13:00

	P 1	P 2	P 3	P 4	P 5	P 6	P 7
I							

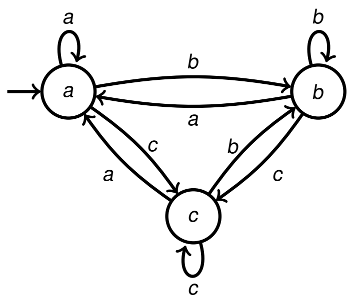
Working instructions

- This exam consists of **20 pages** with a total of **7 problems**.
- The total amount of achievable credits in this exam is 45 credits.
- Allowed resources:
  - any electronic resources accessible using only the external mouse
- All answers have to be written on your own paper.
- Only write on one side of each sheet of paper.
- Write with black or blue pen on white DIN A4 paper.
- Write your name and immatriculation number on every sheet.

Left room from ________ to ________ / Early submission at ________

Problem 1 **Acceptance conditions (4 credits)**

Consider the following ω-automaton $\mathcal{A}$



Notice that when we read any word w on $\mathcal{A}$, reading letter l leads to state l for every $l \in \{a, b, c\}$.
Consider the following ω-languages over $\Sigma = \{a, b, c\}$, where $\text{inf}(w)$ denotes the set of letters occurring infinitely often in the infinite word w:

- $L_1 = \{w \in \Sigma^\omega : \{a, b\} \subseteq \text{inf}(w)\}$,
- $L_2 = \{w \in \Sigma^\omega : a \notin \text{inf}(w) \text{ or } b \notin \text{inf}(w)\}$,

a) Interpreting $\mathcal{A}$ as a generalized Büchi automaton, can you define an acceptance condition such that $\mathcal{A}$ accepts language L_1? If yes, give the acceptance condition. If no, give a short justification.

b) Interpreting $\mathcal{A}$ as a Rabin automaton, can you define an acceptance condition such that $\mathcal{A}$ accepts language L_1? If yes, give the acceptance condition. If no, give a short justification.

c) Interpreting $\mathcal{A}$ as a Büchi automaton, can you define an acceptance condition such that $\mathcal{A}$ accepts language L_2? If yes, give the acceptance condition. If no, give a short justification.

d) Interpreting $\mathcal{A}$ as a Muller automaton, can you define an acceptance condition such that $\mathcal{A}$ accepts language L_2? If yes, give the acceptance condition. If no, give a short justification.

Problem 2 Pattern matching (5 credits)

Consider the pattern p = "*assas*" over the alphabet $\Sigma = \{a, s\}$.

a) Construct an NFA A_p recognizing $\Sigma^* p$ according to the construction specified in the lectures.

0
1

b) Construct the DFA B_p by applying the powerset construction on the NFA A_p.

0
1
2

0
1
2

c) Construct the lazy DFA C_p for the pattern p by using B_p.

Problem 3 Coordinators for DFA (8 credits)

A word w is said to be a *coordinator* for a DFA $A = (Q, \Sigma, \delta, q_0, F)$ if there is a state $p \in Q$ such that **for all states** $q \in Q$, $\delta(q, w) = p$. Intuitively, the word w acts as a coordinating mechanism among all the states, in the sense that reading this word from *any* state of the automaton leads to the same common state.

a) Give an example of a 2-state DFA A which has a coordinator and also give an example of a 2-state DFA B which **does not have** a coordinator.

0 1 2

b) Give an example of a 4-state DFA A such that every state of A is reachable from every other state and any shortest coordinator of A is of length 3.

0 1 2

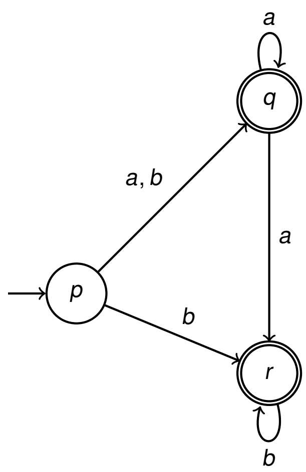
0 1 2 3 4

c) Describe an algorithm that takes as input a DFA A and decides whether A is coordinating or not. Your description has to be sufficiently precise but you do not need to give a pseudocode of your procedure. (**Hint:** You can take inspiration from the *NFAtoDFA* algorithm).

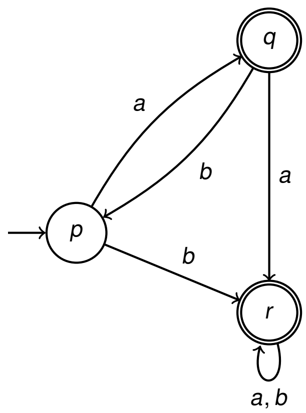
Problem 4 1-loop automata (9 credits)

An NFA A is said to be a *1-loop NFA* if A **does not** contain any simple cycle beyond self-loops, i.e. there are no two distinct states p, q such that p is reachable from q and q is reachable from p. A 1-loop DFA is a 1-loop NFA which is also a DFA.

The following automaton is a 1-loop NFA.



The following automaton **is not a** 1-loop NFA, because there is a cycle between the states p and q.



a) Prove or disprove: For every 1-loop NFA A, the output of *NFAtoDFA(A)* is a 1-loop DFA. Here, *NFAtoDFA* is the algorithm which converts an NFA to a DFA by means of the powerset construction.

0
1
2
3

b) Prove or disprove: For every pair of 1-loop NFAs A and B, the output of *IntersNFA*(A, B) is a 1-loop NFA. Here *IntersNFA* is the algorithm which takes as input two NFAs A and B and outputs an NFA which accepts the intersection of the languages of A and B.

0
1
2
3

c)

For **3 bonus points**, prove or disprove the following: For every 1-loop NFA A, the minimal DFA which recognizes the same language as A is a 1-loop DFA.

Problem 5 Graph of regular languages (6 credits)

Consider the following directed graph $G = (V, E)$:

- The set V of nodes is the set of all regular languages over the alphabet $\Sigma = \{a, b\}$. (So the graph has infinitely many nodes.)
- For any two regular languages $L_1, L_2 \subseteq \Sigma^*$, there is an edge $(L_1, L_2) \in E$, also denoted $L_1 \to L_2$, iff $L_2 = L_1^a$ or $L_2 = L_1^b$. (That is, iff L_2 is the residual of L_1 w.r.t. a or w.r.t. b.)

Given two nodes $L_1, L_2 \in V$, we say that L_2 is reachable from L_1 if $L_1 = L_2$ or if there exists a path (a sequence of edges) leading from L_1 to L_2. We write *Reach*(L) the set of languages reachable from a node L of V.

0 1

a) Give two regular languages L_1, L_2 such that $L_1 \neq L_2$ and $Reach(L_1) = Reach(L_2) = \{L_1, L_2\}$. Describe the languages as regular expressions.

0 1 2

b) A sink of G is a language L such that $Reach(L) = \{L\}$. Give regular expressions for all sinks of G, and prove that there is no other sink.

0 1

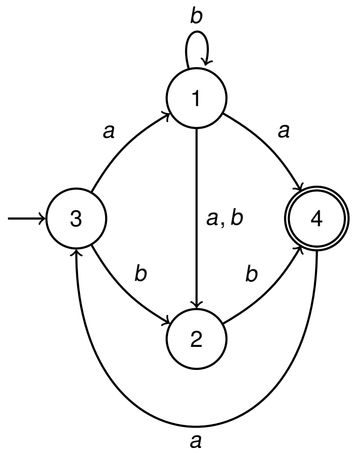
c) Let L be the language described by the regular expression $(aaa)^*a$. Draw the fragment of G containing all the languages of $Reach(L)$ and all edges between them. Represent all languages as regular expressions, and recall that $\Sigma = \{a, b\}$.

0 1 2

d) Prove or disprove: For every regular language L the set $Reach(L)$ is finite.

Problem 6 **Automata and regular expressions (7 credits)**

Let A be the following NFA.



For the purposes of this problem,

- Whenever you use the algorithm *NFAtoRE*, **you must remove states in ascending order**, i.e., you must first remove the state 1, then state 2 and so on.

- While writing the solution, if you come across a long regular expression, you can abbreviate it by a variable and use this abbreviation. **For example**, you can let σ stand for the regular expression $(b^*a + a^*b)^*$ and then instead of writing $(b^*a + a^*b)^*$ throughout the solution, you can instead use σ.

a)

0 1 2 3 4

Use the *NFAtoRE* algorithm, as described in the lectures, to convert A into a regular expression. **The solution must contain the automaton after the preprocessing step and also the automata obtained after removing each state.**

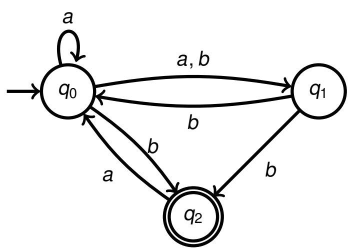
b)

0
1
2
3

Consider A as a non-deterministic Büchi automaton and compute an ω-regular expression for A. You may use the results of the first subproblem for this subproblem. If you are using *NFAtoRE*, you do not need to draw each intermediate automaton. It is sufficient to give the final result while describing the steps that you have followed.

Problem 7 **Büchi Complementation (6 credits)**

Consider the following Büchi automaton $\mathcal{B}$



We denote by $\overline{\mathcal{B}}$ the complement of $\mathcal{B}$, defined with level rankings and owing states as in the lecture. We write the states of $\overline{\mathcal{B}}$ as the pairs $[lr, O]$ where lr is a level ranking and O is the set of owing states. We write lr with rank of q_0 on top, rank of q_1 below, and rank of q_2 on the bottom.

a) For the following pairs $[lr, O]$, say whether or not they are states of $\overline{\mathcal{B}}$. If they are not, give a justification.

- $\begin{bmatrix} \begin{matrix} 2 \\ 0 \\ 0 \end{matrix} , \{q_0, q_2\} \end{bmatrix}$
- $\begin{bmatrix} \begin{matrix} \bot \\ 5 \\ 5 \end{matrix} , \emptyset \end{bmatrix}$
- $\begin{bmatrix} \begin{matrix} 1 \\ \bot \\ 0 \end{matrix} , \{q_1, q_2\} \end{bmatrix}$

b) For the following transitions, say whether or not they are transitions of $\overline{\mathcal{B}}$. If they are not, give a justification.

- $\begin{bmatrix} \begin{matrix} 2 \\ 1 \\ 0 \end{matrix} , \{q_0, q_2\} \end{bmatrix} \xrightarrow{a} \begin{bmatrix} \begin{matrix} 1 \\ 0 \\ \bot \end{matrix} , \{q_1\} \end{bmatrix}$
- $\begin{bmatrix} \begin{matrix} 6 \\ 4 \\ 0 \end{matrix} , \{q_0\} \end{bmatrix} \xrightarrow{b} \begin{bmatrix} \begin{matrix} 4 \\ 3 \\ 0 \end{matrix} , \{q_0, q_2\} \end{bmatrix}$
- $\begin{bmatrix} \begin{matrix} 6 \\ 4 \\ 0 \end{matrix} , \{q_0\} \end{bmatrix} \xrightarrow{b} \begin{bmatrix} \begin{matrix} 4 \\ 3 \\ 0 \end{matrix} , \{q_2\} \end{bmatrix}$
- $\begin{bmatrix} \begin{matrix} 3 \\ 3 \\ \bot \end{matrix} , \emptyset \end{bmatrix} \xrightarrow{b} \begin{bmatrix} \begin{matrix} 2 \\ 0 \\ 0 \end{matrix} , \{q_1, q_2\} \end{bmatrix}$
- $\begin{bmatrix} \begin{matrix} \bot \\ 3 \\ 3 \end{matrix} , \emptyset \end{bmatrix} \xrightarrow{b} \begin{bmatrix} \begin{matrix} \bot \\ \bot \\ 2 \end{matrix} , \{q_2\} \end{bmatrix}$

c)

This question is for **2 bonus points**. Let A be any DBA. States p and q of A are said to be *mutually reachable* if p is reachable from q and q is reachable from p. A is said to be a *uniform* DBA if the following is true: For every pair of mutually reachable states p, q, either both p and q are accepting states or both p and q are rejecting states.

Prove the following: If A is a uniform DBA recognizing an ω-regular language L, then there is a uniform DBA B such that B recognizes the **complement of** L.

0
1
2

Additional space for solutions–clearly mark the (sub)problem your answers are related to and strike out invalid solutions.