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

Esolution
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							

Sample Solution

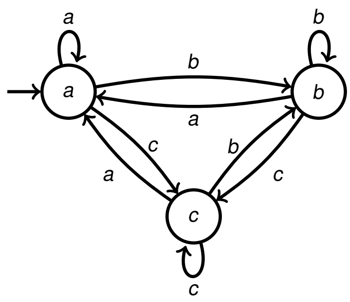
Working instructions

- 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 $\inf(w)$ denotes the set of letters occurring infinitely often in the infinite word w:

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

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.

Yes, $\{\{a\}, \{b\}\}$.

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.

No: there must be a pair $\langle F, G\rangle$ in the acceptance condition with $a, b \in F$, but then a^ω is accepted, contradiction.

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.

No: let F be the accepting states. Since $a^\omega \in L_2$, we have $a \in F$. Similarly $b \in F$. But then $(ab)^\omega$ is accepted, contradiction.

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.

Yes, $\{\{a\}, \{b\}, \{c\}, \{b, c\}, \{a, c\}\}$.

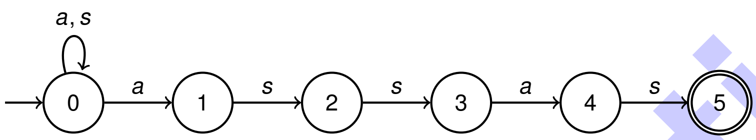
Sample Solution

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.

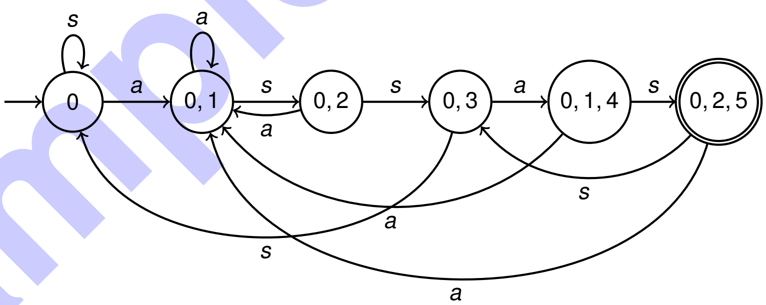
0
1



Sample Solution

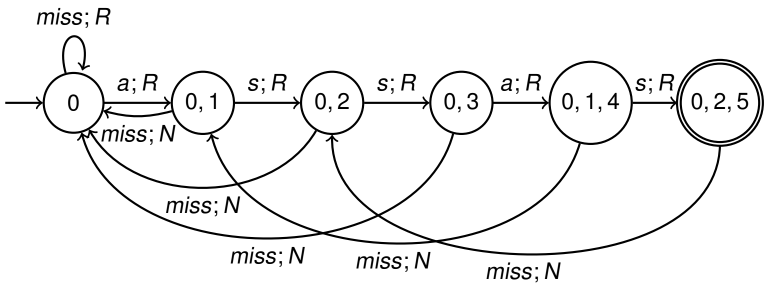
b) Construct the DFA B_p by applying the powerset construction on the NFA A_p.

0
1
2



0
1
2

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



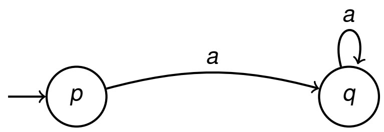
Sample Solution

Problem 3 Coordinators for DFA (8 credits)

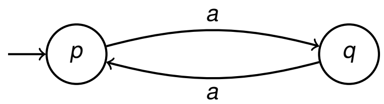
A word w is said to be a *coordinator* for a DFA $A = (Q, \Sigma, \delta, q_0, F)$ if there is a state $p \in Q$ such that **for all states** $q \in Q$, $\delta(q, w) = p$. Intuitively, the word w acts as a coordinating mechanism among all the states, in the sense that reading this word from *any* state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

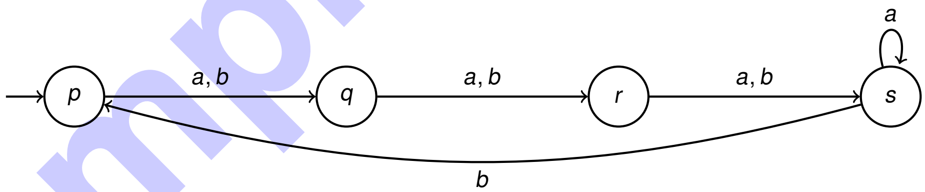
A is the following DFA.



B is the following DFA.



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.



Notice that for any state q_1 of A, there is another state q_2 such that the shortest path from q_2 to q_1 is of length 3. Hence, the shortest coordinator of A must be of length at least 3. Further, aaa is a coordinator of 3 and so A is the required DFA.

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).

Sample Solution

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA with state space Q. Let A' be the NFA given by $(Q, \Sigma, \delta, Q, F)$ and let $B = (\mathcal{Q}, \Sigma, \Delta, Q, \mathcal{F})$ be the DFA obtained by running *NFAtoDFA(A')*.
For any word w, recall that $\delta(q, w)$ denotes the set of states reachable from q after reading the word w in A and $\Delta(Q, w)$ denotes the (unique) state reachable from Q after reading the word w in B. Note that by definition of the construction of B, $\Delta(Q, w) = \cup_{q \in Q} \delta(q, w)$ for any word w.
Notice that w is a coordinator for A iff there is a state p in Q such that $\delta(q, w) = \{p\}$ for all $q \in Q$ which (since A is a DFA), is true iff $\cup_{q \in Q} \delta(q, w) = \{p\}$ which is true iff $\Delta(Q, w) = \{p\}$. Hence,

w is a coordinator for A iff there is a state $p \in Q$ such that $\Delta(Q, w) = \{p\}$ in the DFA B.

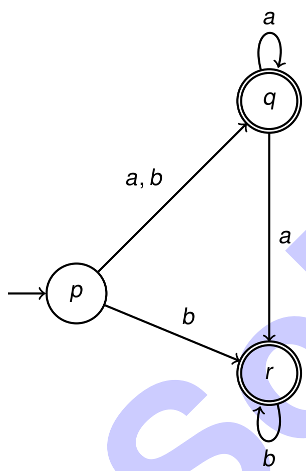
Hence, to check if A has a coordinator we only need to check if there is a state $p \in Q$ such that $\{p\}$ is reachable from the state Q in the DFA B. To do this, we first construct the DFA B. Then we iterate over all states p of A and then check if there is a path from Q to $\{p\}$ in B, which can be done by either a BFS or a DFS.

Sample Solution

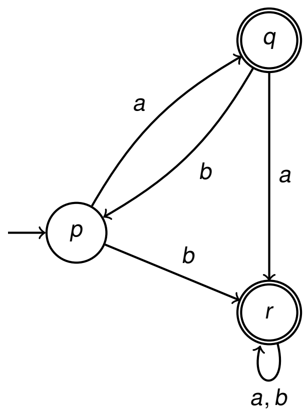
Problem 4 **1-loop automata (9 credits)**

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.



0 1 2 3

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.

Sample Solution

This claim is false. Let A be an NFA with initial state p and final state q over the alphabet $\{a, b\}$ with the following transitions: $\delta(p, a) = \{p, q\}, \delta(p, b) = \{p\}$. It is clear that A is a 1-loop NFA. If we let B be the DFA obtained by the powerset construction on the NFA A, then we get the cycle $\delta(\{p\}, a) = \{p, q\}$ and $\delta(\{p, q\}, b) = \{p\}$. So, B is not a 1-loop DFA.

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.

This claim is true. Let A and B be any two 1-loop NFAs and let C = *IntersNFA*(A, B). Suppose C is not a 1-loop NFA. Let $(p_0, q_0) \to (p_1, q_1) \to \ldots (p_k, q_k) \to (p_0, q_0)$ be a simple cycle which is not a self-loop in C. Hence, $(p_1, q_1) \neq (p_0, q_0)$ and so without loss of generality, we can assume that $p_1 \neq p_0$. By definition of C, this implies that $p_0 \to p_1 \to \ldots p_k \to p_0$ is a cycle in A and so there is a path from the state p_1 to p_0 in A. Since, $p_0 \neq p_1$, this immediately implies that there is a simple cycle from p_0 to p_0 which passes through $p_1 \neq p_0$ and so we have a simple cycle in A which is not a self-loop, which contradicts the fact that A is a 1-loop NFA.

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.

Sample Solution

The claim is false. We consider the same NFA A from the solution of the first subproblem which recognizes the language $(a+b)^*a$. Suppose the minimal DFA B which recognizes this language is also a 1-loop DFA. Let q_0 be the initial state of B and let q_1, q_2 be such that $q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2$. We note that the language of q_0 is, by definition, $(a+b)^*a$ and so the languages of q_1 and q_2 must be $\epsilon + (a+b)^*a$ and $(a+b)^*a$. Since B is minimal, this means that $q_2 = q_0$ and $q_1 \neq q_0$ and so there is a cycle between q_0 and q_1, which leads to a contradiction.

Sample Solution

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.

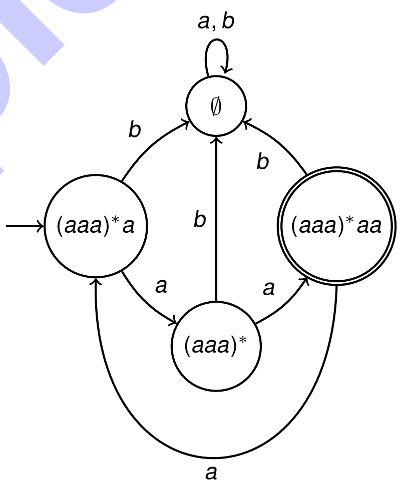
a) Give two regular languages L_1, L_2 such that $L_1 \neq L_2$ and $Reach(L_1) = Reach(L_2) = \{L_1, L_2\}$. Describe the languages as regular expressions.

For example, $L_1 = \Sigma(\Sigma\Sigma)^*$ and $L_2 = (\Sigma\Sigma)^*$. Indeed $L_1^a = L_1^b = L_2$ and respectively for the residuals of L_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.

The two sinks of G are Σ^* and $\emptyset$. They are clearly sinks as their residuals are equal to themselves. Let L be some sink of G. Assume L is not empty, and w is a word in L. By definition of being a sink, $L^a = L^b = L$. Thus $L^w = L$, so L contains the empty word ε. Now take any word u. Since $L^u = L$ by the same reasoning as above, we have $\varepsilon \in L^u$, and thus by definition of residuals $u\varepsilon = u \in L$. So $L = \Sigma^*$. Therefore the only two sinks of G are Σ^* and $\emptyset$.

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 $\Sigma = \{a, b\}$.



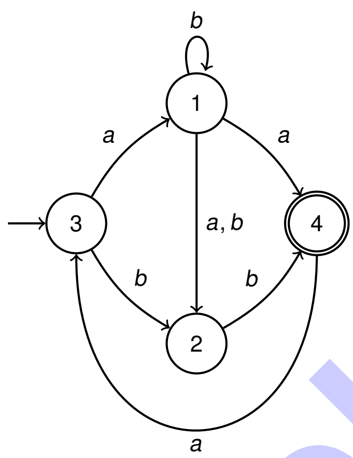
d) Prove or disprove: For every regular language L the set $Reach(L)$ is finite.

True: A regular language only has finitely many residuals. Since every language reachable from L in G is a residual, L can only reach finitely many other languages.

Sample Solution

Problem 6 Automata and regular expressions (7 credits)

Let A be the following NFA.



Sample Solution

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.

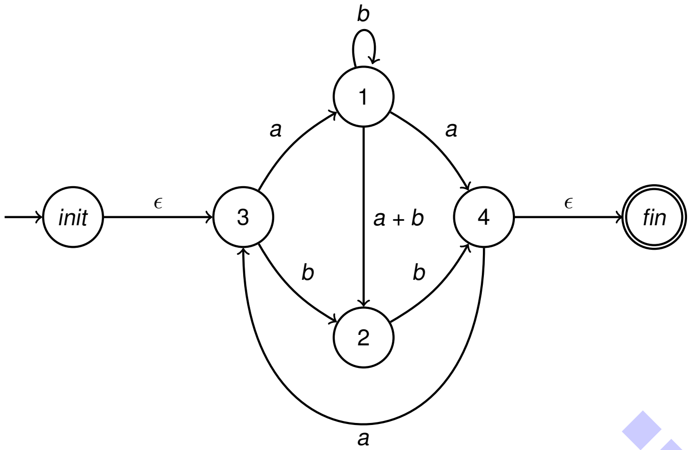
- While writing the solution, if you come across a long regular expression, you can abbreviate it by a variable and use this abbreviation. **For example**, you can let σ stand for the regular expression $(b^*a + a^*b)^*$ and then instead of writing $(b^*a + a^*b)^*$ throughout the solution, you can instead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

a)

0
1
2
3
4

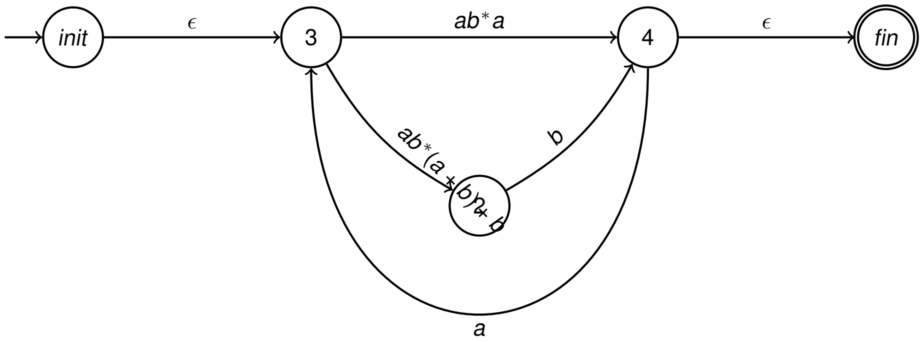
Use the *NFAtoRE* algorithm, as described in the lectures, to convert A into a regular expression. **The solution must contain the automaton after the preprocessing step and also the automata obtained after removing each state.**

We perform a preprocessing step and also merge the multiple transitions between states 1 and 2 into a single transition.

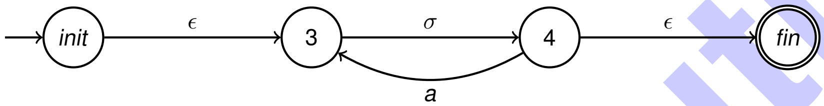


Sample Solution

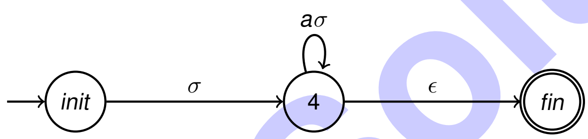
Now we remove state 1 and also merge any multiple transitions along the way.



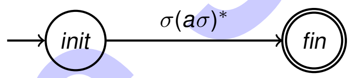
Now we remove state 2 and also merge any multiple transitions along the way. Let $\sigma = ab^*a + (ab^*(a + b) + b)b$ in the sequel.



Now we remove state 3.



And finally we remove state 4.



Hence, the final regular expression is $\sigma(a\sigma)^*$.

Sample Solution

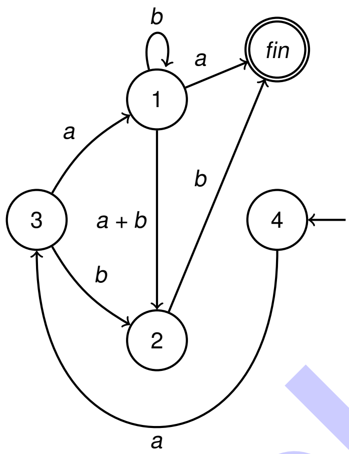
b)

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.

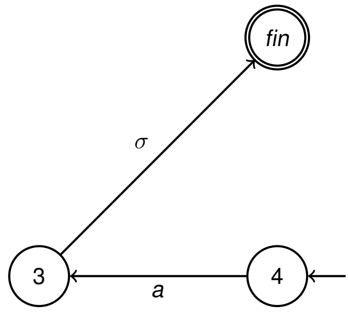
First, we need to compute $r_{3,4}$, i.e., a regular expression for the set of words with runs leading from state 3 to state 4 while visiting state 4 exactly once after leaving state 3. To do this, it suffices to compute a regular expression for the automaton B obtained by from A by deleting the transition $4 \xrightarrow{a} 3$. From the steps of the solution for the first subproblem, this is exactly σ.

Next, we need to compute $r_{4,4}$, i.e., a regular expression for the set of words with runs leading from state 4 to state 4 while visiting state 4 exactly once after leaving state 4. To do this, we need to redirect all the incoming arrows of 4 to a new state *fin* and take the initial state to be 4 and the final state to be *fin*. This results in the following automaton.



Sample Solution

Removing states 1 and 2 is exactly similar to the solution of the first subproblem and so we get

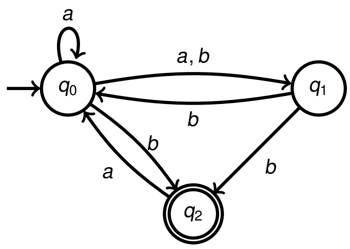


Removing state 3 gives us that the resulting regular expression is $a\sigma$.
Hence, the ω-regular expression for A is $\sigma \cdot (a\sigma)^{\omega}$.

Sample Solution

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.

- $\left[\begin{matrix}2\\0\\0\end{matrix}, \{q_0, q_2\}\right]$
- $\left[\begin{matrix}\bot\\5\\5\end{matrix}, \emptyset\right]$
- $\left[\begin{matrix}1\\\bot\\0\end{matrix}, \{q_1, q_2\}\right]$

- Yes.
- No: q_2 is accepting and must have even rank.
- No: q_1 should not be in the owing states as $lr(q_1) \notin [0, 2n]$.

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

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

Sample Solution

- No: $q_2 \xrightarrow{a} q_0$ so rank cannot increase.
- No: $q_0 \notin \delta(q_0, b)$.
- Yes.
- No: q_0 should be in the owing states.
- No: q_0 should have a rank since $q_1 \xrightarrow{b} q_0$. Another reason is: $\left[\begin{array}{c} \bot \\ 3 \\ 3 \end{array}, \emptyset\right]$ is not a valid state since q_2 should not have odd rank.

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

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a uniform DBA recognizing L. Consider the DBA $B = (Q, \Sigma, \delta, q_0, Q \setminus F)$. Notice that B is a uniform DBA. We claim that B recognizes the complement of L.
Let w be any infinite word and let ρ be the unique run of A on the word w (unique because A is deterministic). Notice that ρ is also the unique run of B on the word w. Let $inf(\rho)$ be the set of states which appear infinitely often in ρ.
Notice that if $p, q \in inf(\rho)$ then p and q are mutually reachable in A. Indeed, let i, j, k be positions along the run ρ such that $i < j < k$, p appears at position i, q appears at position j and p appears at position k. (Such positions exist because $p, q \in inf(\rho)$). This implies that there must be a path from p to q and a path from q to p, which enables us to conclude that p and q are mutually reachable. Hence, all the states in $inf(\rho)$ are mutually reachable from one another. Since A is uniform, this implies that either $inf(\rho) \subseteq F$ or $inf(\rho) \subseteq Q \setminus F$.
If $inf(\rho) \subseteq F$, this means that w is accepted by A. Since ρ is also the unique run of B on w and since the accepting states of B is $Q \setminus F$, it follows that B rejects the word w.
If $inf(\rho) \subseteq Q \setminus F$, this means that w is rejected by A. Since ρ is also the unique run of B on w and since the accepting states of B is $Q \setminus F$, it follows that B accepts the word w.
It follows that B recognizes the complement of L.

Notice that a Büchi automaton may accept words w that are not accepted by a run that ends in a cycle/ that are not of the form $w = u(v)^\omega$ with $u, v \in \Sigma^*$. For example, the Büchi automaton that accepts $(0 + 1 + ... + 9)^\omega$ accepts the word w of the decimals of $\sqrt{2}$.

Sample Solution

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

Sample Solution

Sample Solution

Sample Solution