DECISION PROBLEMS FOR REGULAR LANGUAGES

Exercise 1:

Give algorithms to tell whether

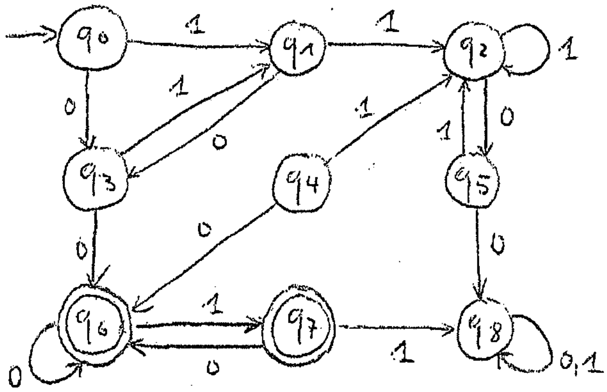
a) a given regular language L is universal. (i.e. $L = \Sigma^*$).

b) two regular languages have at least one string in common.

STATE MINIMIZATION

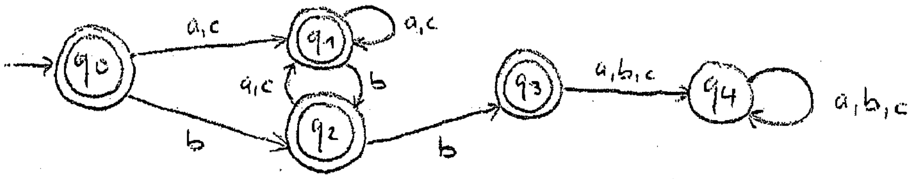
EXERCISE 2

Minimize the following DFA.



EXERCISE 3

Minimize the following DFA.



SOLUTIONS

1) a) If $L = \Sigma^*$, then $\overline{L} = \overline{\Sigma^*} = \emptyset$

Hence, we need to check whether $\overline{L}$ is empty.

Algorithm when L is given as a DFA D_L:

1) Construct a DFA $D_{\overline{L}}$ s.t. $\mathcal{L}(D_{\overline{L}}) = \overline{L}$ by swapping final and non-final states of D_L.
2) Check whether $D_{\overline{L}}$ is empty (by constructing the set of states reachable from the initial state, and checking whether it contains at least one final state)

Algorithm when L is given as an NFA N_L:

1) Determinize N_L, i.e. construct a DFA D_L s.t. $\mathcal{L}(D_L) = \mathcal{L}(N_L)$ (Note: D_L might have a number of states that is exponential in the number of states of N_L).
2) Proceed as in the case of a DFA

Algorithm when L is given as a RE E_L:

1) Construct an ε-NFA N_{ε_L} s.t. $\mathcal{L}(N_{\varepsilon_L}) = \mathcal{L}(E_L)$
2) Eliminate ε-transitions from N_{ε_L}, obtaining an NFA N_L s.t. $\mathcal{L}(N_L) = \mathcal{L}(N_{\varepsilon_L})$
3) Proceed as in the case of an NFA

1) b) To check whether two RLs L_1 and L_2 have at least one string in common, we can check whether $L_1 \cap L_2$ is nonempty.

Algorithm:

1) Construct a DFA / NFA / ε-NFA / RE for $L_1 \cap L_2$, starting from DFAs / NFAs / ε-NFAs / REs for L_1 and for L_2

2) Check whether $L_1 \cap L_2$ is not-empty.

Note: - to construct a DFA / NFA / ε-NFA / RE for $L_1 \cap L_2$, we can use De Morgan's law.

- $L_1 \cap L_2$ is still a RL, since RLs are closed under intersection.

E8.4

2) we start with the set of all states, and in step 0 the set will be separated into two sets of final and Non-final states. (we will use 0, 1, ..., 8 instead of $q_0, q_1, \ldots q_8$).

{0, 1, 2, 3, 4, 5, 6, 7, 8}

step0: {0, 1, 2, 3, 4, 5, 8} {6, 7}

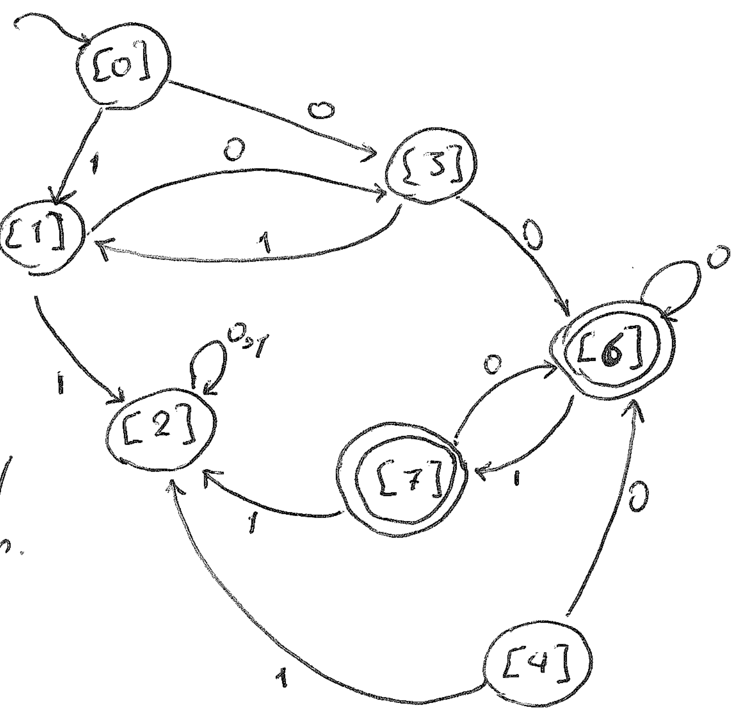
step1: {0, 1, 2, 5, 8} {3, 4} {6} {7}

step2: {2, 5, 8} {0, 1} {3, 4} {6} {7}

step3: {2, 5, 8} {0} {1} {3} {4} {6} {7}

step4: {2, 5, 8} {0} {1} {3} {4} {6} {7} <no change>

[2] = {2, 5, 8}



Since the State [4] is not reachable from the initial state, it should be eliminated from the minimized automaton.

Exercise 3) Same as previouse exercise we will have:

$\{0,1,2,3,4\}$

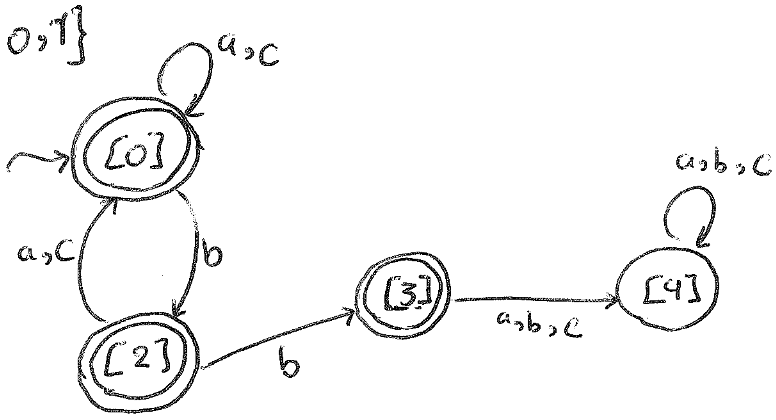
Step 0: $\{4\}$ $\{0,1,2,3\}$

Step 1: $\{4\}$ $\{3\}$ $\{0,1,2\}$

step 2: $\{4\}$ $\{3\}$ $\{0,1\}$ $\{2\}$

step 3: $\{4\}$ $\{3\}$ $\{0,1\}$ $\{2\}$

$[0] = \{0,1\}$



attention: Although none of final states are reachable from the state [4], it shouldn't be eliminated because we are looking for a DFA, and by eliminating [4], the automaton will not be a DFA anymore!

All the states are reachable from the initial state, so we don't need to eliminate any state and the DFA is minimum.