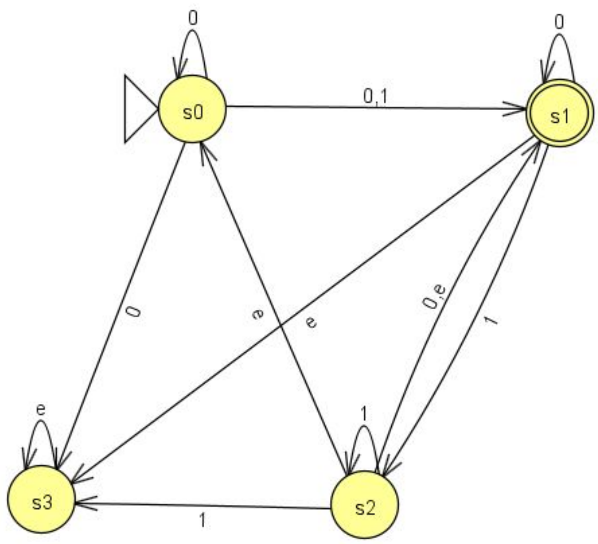
Automata, Computability, and Complexity

Hamza Bouhelal, Mohamed Reda Arsalane, Mahdi Ouchrahou

SHEET #2:

Excercise 1:

a)



b)

$N_2 = (\{Q_0, Q_1, Q_2, Q_3, Q_4\} , \{0,1\} , \delta , Q_0 , \{Q_1\})$

where the transition function δ is:

	0	1	e
Q_0	$\{Q_4\}$	$\{Q_0, Q_1, Q_4\}$	$\{Q_1, Q_4\}$
Q_1	$\{Q_2\}$	$\{Q_2\}$	$\{Q_2\}$
Q_2	$\{Q_0\}$	$\varnothing$	$\varnothing$
Q_3	$\{Q_1\}$	$\{Q_1\}$	$\{Q_4\}$
Q_4	$\{Q_1, Q_4\}$	$\{Q_4\}$	$\varnothing$

Excercise 2:

a)

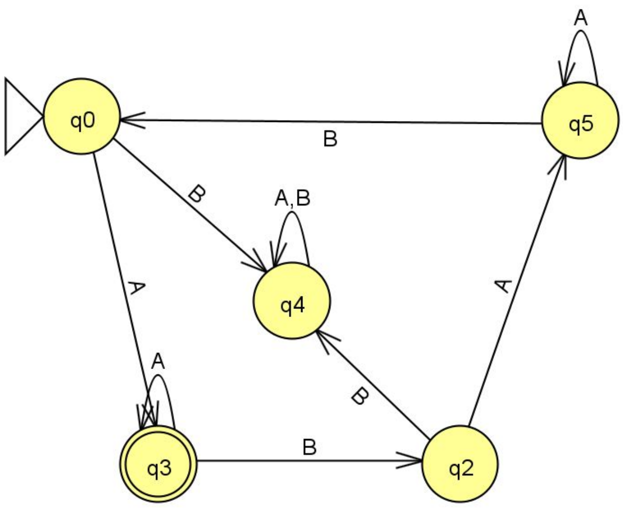
L(M1) = {w | w is a string starting with 0 and ending with 1}

b)

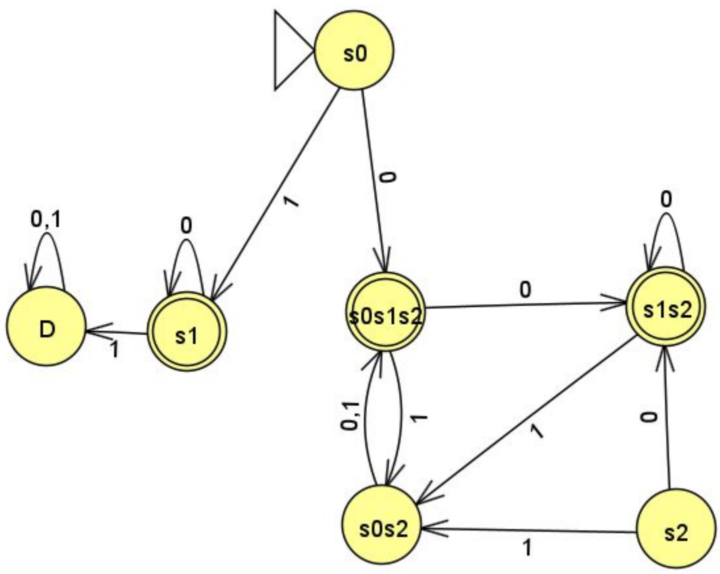
L(M2)= {w | w is a string that contains ab n times after the first b or a string   no b}

Excercise 3:

a)



b)



Excercise 4:

The complement language L□= {w ∈ Σ* |w ∉ L} is the set of words created from the alphabet Σ* and that are not included in L.

Since we can convert every NFA to a DFA like so:
Suppose there is an NFA N < Q, ∑, q0, δ, F > which recognizes a language L. Then the DFA D < Q', ∑, q0, δ', F' > can be constructed for language L as:
Step 1: Initially Q' = ɸ.
Step 2: Add q0 to Q'.
Step 3: For each state in Q', find the possible set of states for each input symbol using transition function of NFA. If this set of states is not in Q', add it to Q'.
Step 4: Final state of DFA will be all states with contain F (final states of NFA)

Knowing that the DFA accepting language L is equivalent to complement of that language L We can find L□ and then construct the NFA accepting L□.