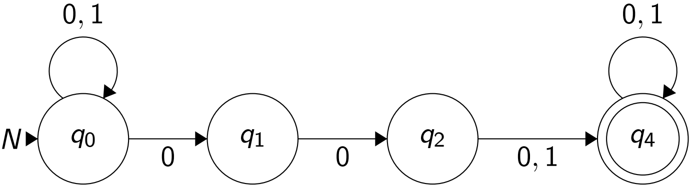
Recit-2

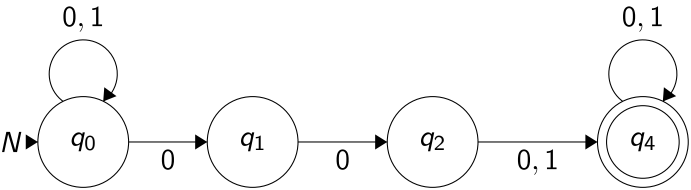
October 19, 2020

Q1



- *a*) Write a regular expression E for the language L accepted by N.
- *b*) Write a regular expression E' for the language L^c which is the comlement of the L. Then write down a ϵNFA N' that accepts E'.

Q1.a)



- Language L accepted by N is the language including all string which contains either 000 or 001 at least once. (strings with sub-strings u starting with 00 and $3 \leq \mathit{length}(u)$)
- Informally the language ...001... $\cup$ ...000...
- $(0+1)^*.0.0.0.(0+1)^* + (0+1)^*.0.0.1.(0+1)^*$ which is equal to $(((0+1)^*.0.0).(0+1)).(0+1)^*$

Q1.b)

Language L is the all string which contains either 000 or 001 at least once.
$(0+1)^*.0.0.0.(0+1)^* + (0+1)^*.0.0.1.(0+1)^*$

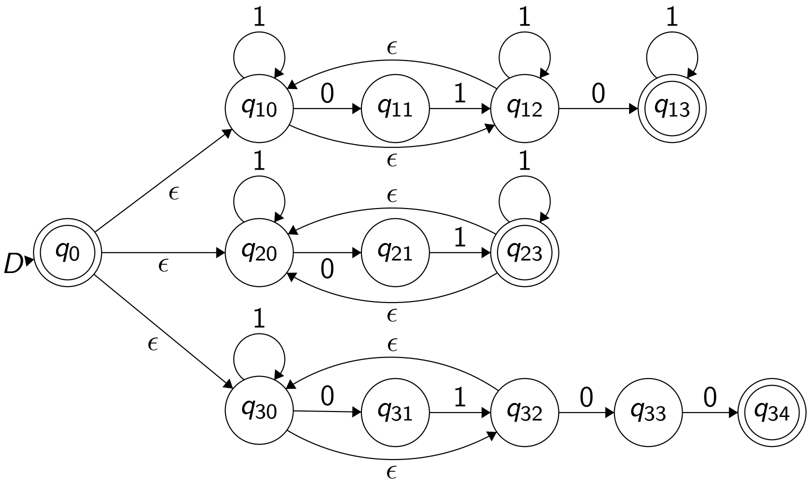
- Complement of the language L is L^c which is the all strings that do not contain any sub-string u starting with 00 and $3 \leq lenght(u)$.
- Examples: 00, 100, 111, 0, 101111, 111110 $\in L^c$.
  000, 1000000000, 111000 is in L.
- No 00 can appear as a infix.
- No 00 can appear as a prefix if $length(s) > 2$.
- Corresponding RE is: $1^*(0.1^+)^*.0.1^* + 1^*(0.1^+)^*.1^* + 1^*(0.1^+)^*.0.0$

  $*$[1]

[1] Not sure if it is the most simple RE representation of L^c

Q1.b)

$L^c := 1^*(0.1^+)^*.0.1^* + 1^*(0.1^+)^*.1^* + 1^*(0.1^+)^*.0.0$



Q2

- Informally; ϵ-closure of a state q is the all states can be reached from q by following ϵ transitions.
- Formally; recursive definition...
- In an ϵNFA, the states you can reach from q on input a is defined as to be $\bigcup_{q' \in \epsilon CLOSE(q)} \epsilon CLOSE(\delta_\epsilon(q', a))$

Q2.a

- $\epsilon Closure(q_1) = q_1, q_2$
- $\epsilon Closure(q_2) = q_2$
- $\epsilon Closure(q_3) = q_3$
- $\epsilon Closure(q_4) = q_1, q_2, q_4$

Then for $\sigma = 0$;

- $\delta_N(q_1, 0) = \epsilon Closure(\delta_\epsilon(q_1, 0)) \cup \epsilon Closure(\delta_\epsilon(q_2, 0)) = q_1, q_2, q_4$
- $\delta_N(q_2, 0) = \epsilon Closure(\delta_\epsilon(q_2, 0)) = q_1, q_2, q_4$
- $\delta_N(q_3, 0) = \epsilon Closure(\delta_\epsilon(q_3, 0)) = \emptyset$
- $\delta_N(q_4, 0) = \epsilon Closure(\delta_\epsilon(q_4, 0)) \cup \epsilon Closure(\delta_\epsilon(q_1, 0)) \cup \epsilon Closure(\delta_\epsilon(q_2, 0)) = q_1, q_2, q_4$

Q2.a

Then for $\sigma = 1$;

- $\delta_N(q_1, 1) = \epsilon Closure(\delta_\epsilon(q_1, 1)) \cup \epsilon Closure(\delta(q_2, 1)) = q_3$
- $\delta_N(q_2, 1) = \epsilon Closure(\delta_\epsilon(q_2, 1)) = \emptyset$
- $\delta_N(q_3, 1) = \epsilon Closure(\delta_\epsilon(q_3, 1)) = q_1, q_2, q_4$
- $\delta_N(q_4, 1) = \epsilon Closure(\delta_\epsilon(q_4, 1)) \cup \epsilon Closure(\delta_\epsilon(q_1, 1)) \cup \epsilon Closure(\delta_\epsilon(q_2, 1)) = q_3$

We know all transitions, sketch the automata.

Q2.b

q	σ	q_{next}
q_0	0	q_1, q_2, q_4
	1	q_3
q_1, q_2, q_4	0	q_1, q_2, q_4
	1	q_3
q_3	0	$\emptyset$
	1	$1, 2, 4$

Rename q_0: A, q_1, q_2, q_4: C, q_3: B, $\emptyset$: D.

Only final state is C since q_4 appears in it.

Q3

Write a regular expression for the set of strings of 0's and 1's with at most one pair of consecutive 1's.

- Every 1 must be followed by a 0, unless it is the only consecutive pair of 1.
- We might have a 11, but it is not necessary.
- $(0 + 1.0)^*(1.1 + \epsilon)(0 + 0.1)^*$

Alternatively, if you can not see it directly, longer but step-by-step methods is the following;

- Write down an *NFA* (with a similar logic as in the BABA example)
- Convert it to a DFA
- Convert it to a RE (state elimination)

Q4

Write down a NFA or DFA that accept the language consisting of all strings over the alphabet $1, 2, 3, 4, 5, 6, 7, 8, 9, 0$ s.t. final digit has appeared before.

- 10 cases, where the final digit is the one of the elements of alphabet.
- For each distinct case, we must make the NFA to check if the 'final digit' appears before the final state.

Q4