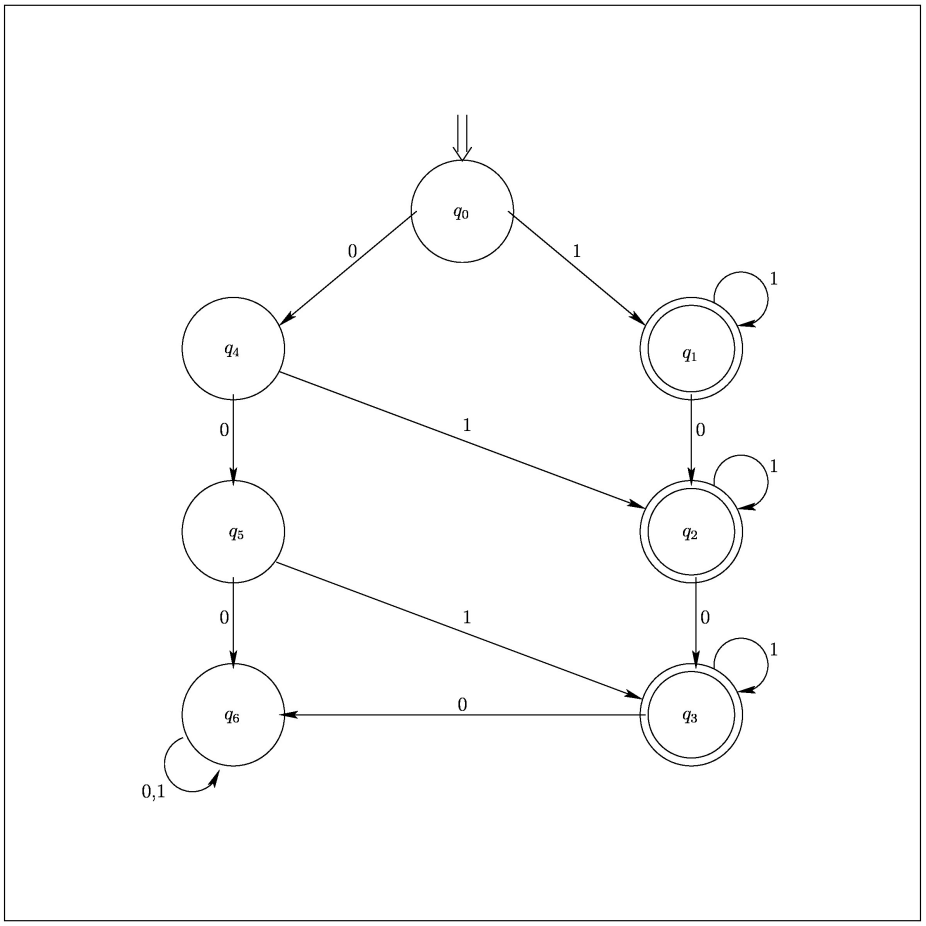
CS340
Assignment 1

Siddharth Agrawal(150716)

August 20, 2017

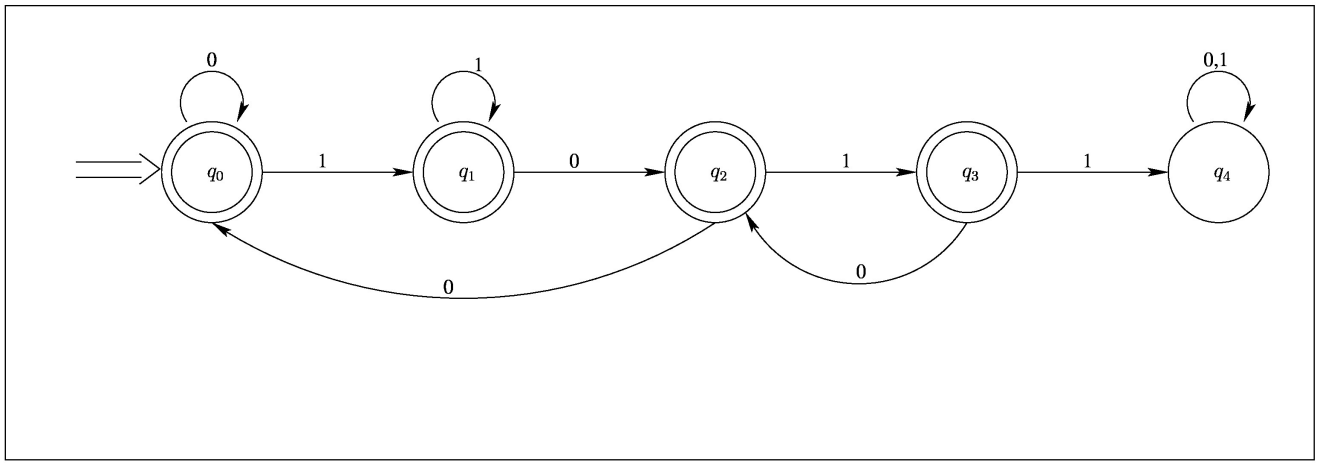
Question 1

Part a



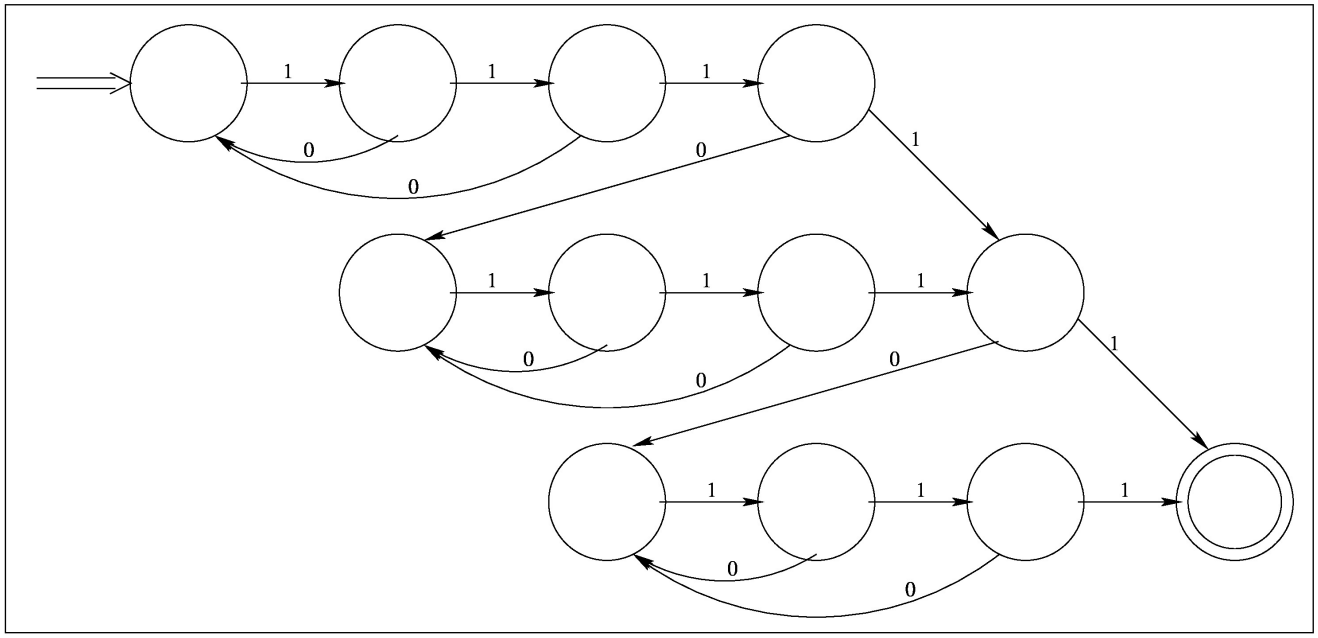
q_0 := ϵ-string
q_1 := string containing one or more 1's and no 0's
q_2 := string containing one or more 1's and one 0
q_3 := string containing one or more 1's and two 0's
q_4 := string containing no 1's and one 0
q_5 := string containing no 1's and two 0's
q_6 := string containing more than two 0's

Part b



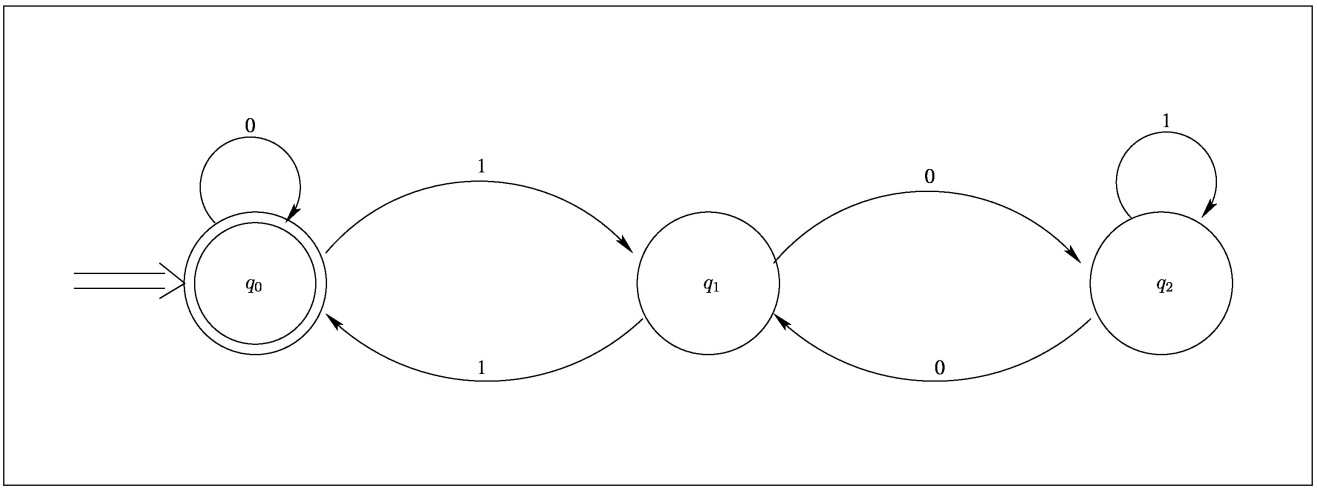
q_0 := ϵ-string or string ending with 0
q_1 := string ending with substring 1
q_2 := string ending with substring 10
q_3 := string ending with substring 101
q_4 := string containing substring 1011

Part c



Question 2

Part a



q_0 := string with RE = (0+1(01*0)*1)*
q_1 := string with RE = 0*1(01*0)*)
q_2 := string with RE = 0*101*

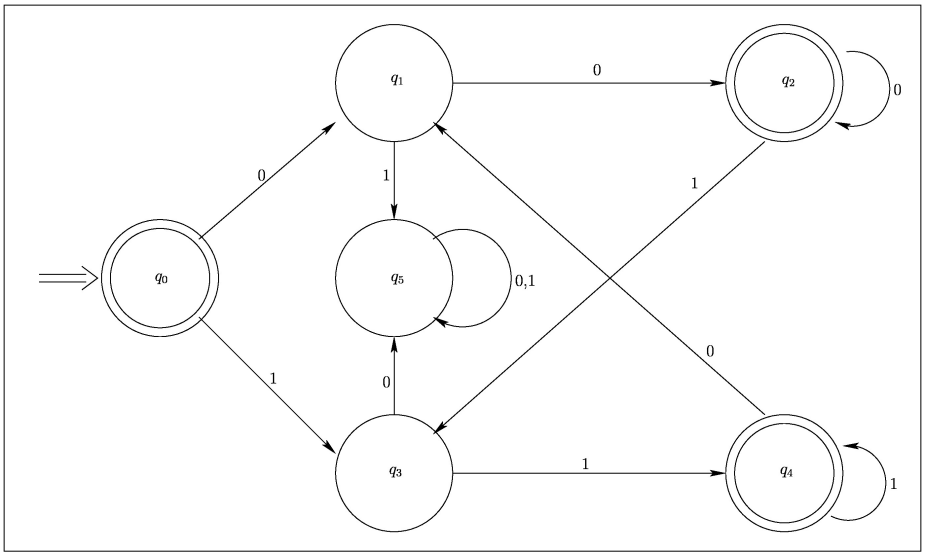
Consider the strings - '100','0'&'10'. Observe that these strings are distinguishable from the other two wrt the language L.
Using **"Distinguishablilty theorem[1]"** we can say that any DFA over L has atleast 3 states. Hence, we cannot have a DFA with lesser than 3 states.

Part b

q_0 := ϵ-string
q_1 := string with RE = (000*+111*)*0
q_2 := string with RE = (000*+111*)*00*
q_3 := string with RE = (000*+111*)*1
q_4 := string with RE = (000*+111*)*11*
q_5 := dump state

Note that all states in the given DFA are required. No two states can be combined.

- The dump state is obviously very important and exclusive to all other states
- The states - q_1 and q_3 (non-accept states) can obviously NOT be combined otherwise they might lose their definition and hence result in loss of important information.



- There are three accept states - q_0, q_2 and q_4. Note, these states CANNOT be combined otherwise it will be impossible to predict which alphabet to loop - '0' or '1'.

Hence, atleast 6 states are required by any DFA on the given language.

Question 3 [2]

Let $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ be the DFA for the language L_1.
Let $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ be the DFA for the language L_2.
Now, we define an NFA(N) which *non-deterministically* chooses which machine to run an input symbol on (i.e. N non-deterministically chooses whether to process δ_1 or δ_2).
Formally, $N = (Q, \Sigma, \delta, q_0, F)$ where -

- $Q = Q_1 \times Q_2$
- $q = (q_1, q_2)$
- $F = F_1 \times F_2$
- For $a \in \Sigma$, we define δ as -

$$\delta((p_1, p_2), a) = \{(\delta_1(p_1, a), p_2), (p_1, \delta_2(p_2, a))\}$$

where $p_1 \in Q_1$ *and* $p_2 \in Q_2$

Now, consider that on a string ω, machine N reaches the state (p_1, p_2).
$\therefore$By the definition of δ we can see that ω can be broken into two strings ω_1 and ω_2 such that machine M_1 reaches state p_1 on string ω_1 and machine M_2 reaches state p_2 on string ω_2.
Note:- if $\omega_1 = x_1x_2.....x_n$ *and* $\omega_2 = y_1y_2.....y_n$ *then* $\omega = x_1y_1x_2y_2.....x_ny_n$*; where* $x_1, x_2, ..., x_n, y_1, y_2, ...y_n \in \Sigma^*$.

Thus, machine N accepts the string ω iff machine M_1 accepts ω_1 and machine M_2 accepts ω_2.
Hence, it is clear that NFA N accepts Mix(L_1, L_2). So, Mix(L_1, L_2) is regular.

□

Question 4[3]

Part a

In this proof, we use the following theorem -

THEOREM 1: A language L is regular if and only if $L = L(R)$ for some regular expression R. Using theorem 1, let R be a regular expression such that $L = L(R)$.

Now, we define homomorphism as an operation on regular expressions as follows -

- $h(\emptyset) = \emptyset$
- $h(\epsilon) = \epsilon$
- $h(a) = h(a)$
- $h(R_1R_2) = h(R_1)h(R_2)$
- $h(R_1 + R_2) = h(R_1) + h(R_2)$
- $h(R^*) = (h(R))^*$

Now, we have to show that $h(L)$ is regular.
Or, we have to show that $h(L(R))$ is regular.
Or, we have to show that $h(L(R)) = L(h(R))$.

$h(R)$ is a regular expression which should have a unique language.
We just have to show that this language is actually same as $h(L)$
which will prove its regularity using theorem 1.

Claim: $h(L(R)) = L(h(R))$
PROOF: We prove this using induction on R.

Base Case:

- For $R = \epsilon$ or $\emptyset$, $h(R) = R$ and $h(L(R)) = L(h(R))$
- For $R = a, L(R) = \{a\}$, and $h(L(R)) = \{h(a)\} = L(h(a)) = L(h(R))$
- So, claim holds

Inductive Hypothesis:
$h(L(R_i)) = L(h(R_i))$
Inductive Step:

- For $R = R_1 + R_2$

$$\begin{aligned} h(R) &= h(R_1) + h(R_2) \\ h(L(R)) &= h(L(R_1) \cup L(R_2)) \\ &= h(L(R_1)) + h(L(R_2)) \\ &= L(h(R_1)) \cup L(h(R_2)) && \text{using inductive hypothesis} \\ &= L(h(R_1) + h(R_2)) \\ &= L(h(R)) \end{aligned}$$

- For $R = R_1 R_2$

$$\begin{aligned} h(R) &= h(R_1)h(R_2) \\ h(L(R)) &= h(L(R_1).L(R_2)) \\ &= h(L(R_1))h(L(R_2)) \\ &= L(h(R_1))L(h(R_2)) && \text{using inductive hypothesis} \\ &= L(h(R_1)h(R_2)) \\ &= L(h(R)) \end{aligned}$$

- For $R = R_1^*$

$$\begin{aligned} h(R) &= (h(R_1))^* \\ h(L(R)) &= h(L(R_1^*)) \\ &= (h(L(R_1)))^* \\ &= (L(h(R_1)))^* && \text{using inductive hypothesis} \\ &= L(h(R_1^*)) \\ &= L(h(R)) \end{aligned}$$

□

Part b

We prove this using construction of a DFA for $h^{-1}(L)$, i.e., if $\exists$ a DFA M' for $h^{-1}(L)$, then it is regular.

Now, given a DFA M recognizing L, construct a DFA M' which accepts $h^{-1}(L)$.
Clearly, the states for both the machines needs to be same (the start and finish state too need to be same), only the transition function needs to be modified such that for $\omega \in \Sigma^*$, M' accepts ω iff M accepts $h(\omega)$.
Formally-
$M = (Q, \Delta, \delta, q_0, F)$ accepts $L \subseteq \Delta^*$
Define $M' = (Q', \Sigma, \delta', q_0', F')$, where

- $Q' = Q$
- $q_0' = q_0$
- $F' = F$
- $\delta'(q, a) = \delta(q, h(a))$ where $a \in \Sigma$

Claim: $\delta'(q, \omega) = \delta(q, h(\omega))$ *where* $\omega \in \Sigma^*$ PROOF: We prove this using induction on $|\omega|$

Base Case: $\omega = \epsilon$
$\delta'(q, \epsilon) = \epsilon$, and $\delta(q, h(\epsilon)) = \delta(q, \epsilon) = q$
Inductive Step: Let $\omega = xa$, assume Induction Hypothesis(IH) for x

$$\begin{aligned} \delta'(q, \omega) &= \delta'(\delta'(q, x), a) \\ &= \delta'(\delta(q, h(x)), a) && \text{using IH} \\ &= \delta(\delta(q, h(x)), h(a)) && \text{using definition of DFA M'} \\ &= \delta(q, h(x)h(a)) \\ &= \delta(q, h(\omega)) \end{aligned}$$

Using above claim, we can say that $\forall h(\omega)$ accepted by M, ω is accepted by M'.
So, M' accepts $h^{-1}(L)$. Hence, $h^{-1}(L)$ is a regular language.

□

References

[1] ddunham. Distinguishability. URL https://www.d.umn.edu/~ddunham/cs3512s11/notes/l15.pdf.

[2] K. Draeger. stackexchange. URL https://cs.stackexchange.com/questions/42751/show-that-regular-languages-are-closed-under-mix-operations.

[3] ullman. rs2.pdf. URL http://infolab.stanford.edu/~ullman/ialc/spr10/slides/rs2.pdf.