CS 3331 – Assignment 1

Name: rd

Student Number: rd

For all questions, unless stated otherwise, let:

M represent an FSM.
K represent a set of states.
Σ represent an alphabet.
s represent an initial state of an FSM.
A represent a set of accepting states (in relation to an FSM)
δ represent a transition function (if M is a DFSM)
Δ represent a transition relation (if M is a FSM)
λ represent the $lambda\ or\ epsilon\ transition$

1.

a. $\{w^R w w^R \mid w \in \{a,b\}^*\}.$

i. *Prove whether it is regular or not*

L is not regular. Proof by contradiction of Pumping Lemma Theorem.

$Let\ v = a^k bba^k a^k b\ requirements\ satisfied -\quad |v| \geq k, v \in L$
$Let\ x = a^i, y = a^j, z = a^{k-i-j} bba^k a^k b \qquad x, y, z \in \Sigma^*$
Assume L is regular. Then:
$For\ some\ k \geq 1, all\ q \geq 0: xy^q z \in L, |xy| \leq k$
$Let\ j = 1, i = k-1, q = 2.$
$xy^q z = a^{k-1} a^2 a^{k-k+1-1} bba^k a^k b$
$= a^{k+1} bba^k a^k \notin L$

Therefore, there exists a $q \geq 0\ s.t. xy^q z \notin L\ for\ all\ k \geq 1.$
Therefore, the Pumping Theorem is not satisfied.
Therefore this language is not regular.

ii. *Construct a NDFSM*
iii. *Convert NDFSM →DFSM*
iv. *Minimize DFSM*
v. *Convert one of the machines into a regular expression*

b. $\{w \in \{0,1\}^* \mid w \text{ has } 1010 \text{ as substring}\}$

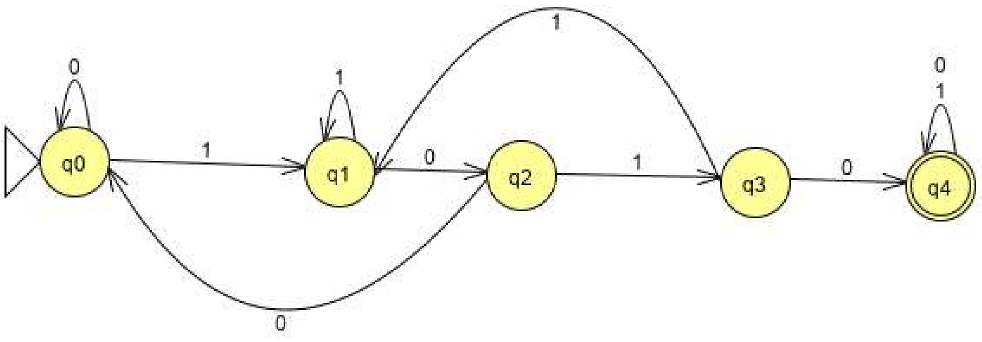
i. *Prove whether it is regular or not*

I will prove that the language is regular by constructing a DFSM. (part iii)

ii. *Construct a NDFSM*

I was able to construct the DFSM directly.

iii. *Convert NDFSM → DFSM*

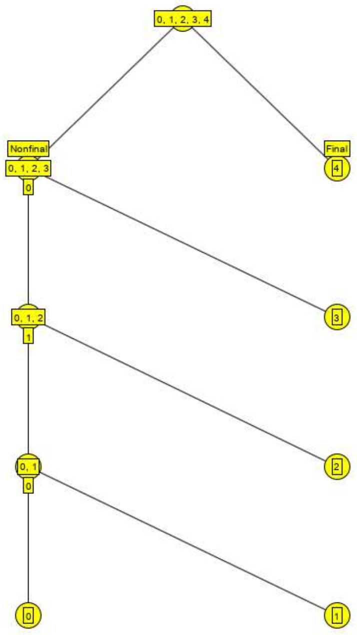


iv. *Minimize DFSM*

M is already minimal because K contains no redundant or unreachable states, as every state:

- Can be reached from the s through a set of transitions $\in \delta$.
- Is unique to every other state in K. There are no two states A, B $\in K$ such that all strings $\in \Sigma^*$ will face the same final outcome (accept or reject) regardless of whether they start at A or starting at B.

Further proof can be seen in the following diagrams representing the overclustering process, that shows there exists a string $\in \Sigma^*$ that reaches each state (illustrated by each state being represented in a leaf node) and that each state is unique (illustrated by each leaf node only containing one state):



v. *Convert one of the machines into a regular expression*

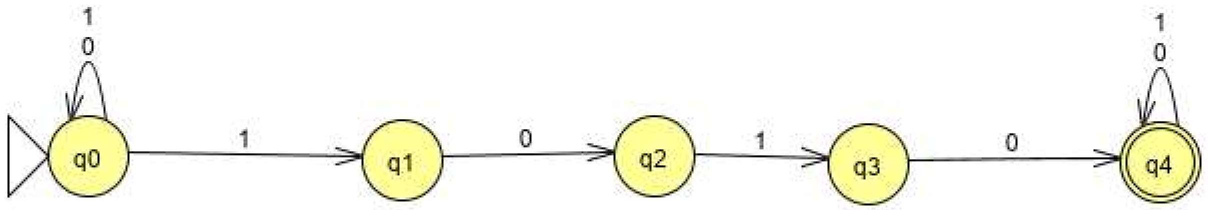
From the language, we can find the regular expression:

$\{w \in \{0,1\}^* \mid w \text{ has } 1010 \text{ as substring}\}$

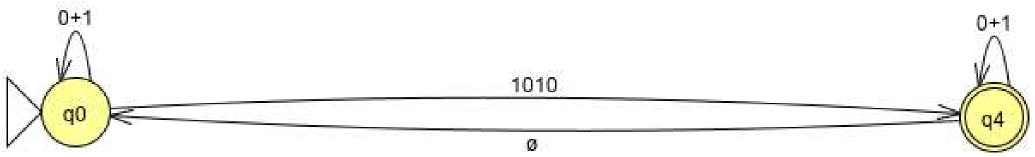
$= (0 \cup 1)^*(1010)(0 \cup 1)^*$

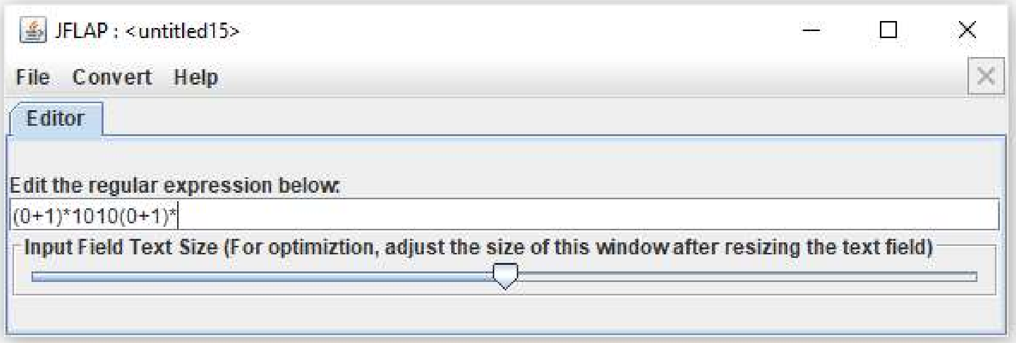
$= (0^*1^*)^*(1010)(0^*1^*)^*$

Further proof using NDFSM:



Collapse multiple transitions, add empty transitions between all states that don't have transitions between them, remove nonfinal, noninitial states:





Edit the regular expression below:

(0+1)*1010(0+1)*

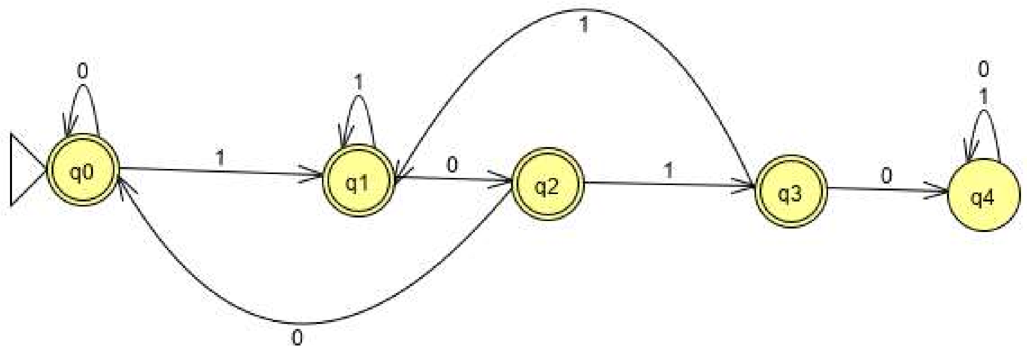
Input Field Text Size (For optimiztion, adjust the size of this window after resizing the text field)

c. $\{w \in \{0,1\}^* \mid w \text{ does not have } 1010 \text{ as substring}\}$

i. *Prove whether it is regular or not*
Since the language in part b is regular,
and since the language in b is the complement of this language,
and since (iff L is regular, then L's complement is regular),
therefore, this language is regular.

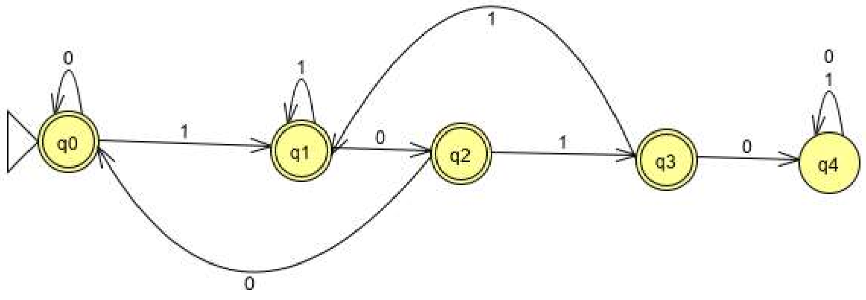
ii. *Construct a NDFSM*
I constructed the DFSM without the NDFSM.

iii. *Convert NDFSM → DFSM*

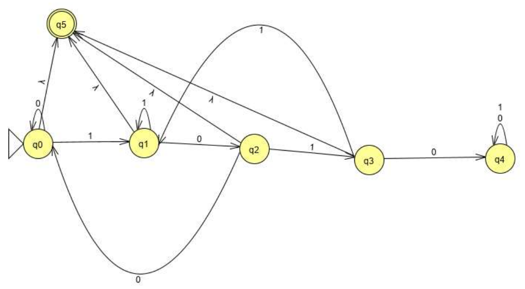


iv. *Minimize DFSM*
Since the complement DFSM is minimal and has 5 states, therefore the complement Language has 5 equivalency classes. Since the complement language has 5 equivalency classes, therefore this language has 5 equivalency classes. Since the minimal DFSM has a number of states equal to the number of equivalency classes, and this DFSM has 5 states, therefore this DFSM is already minimal.

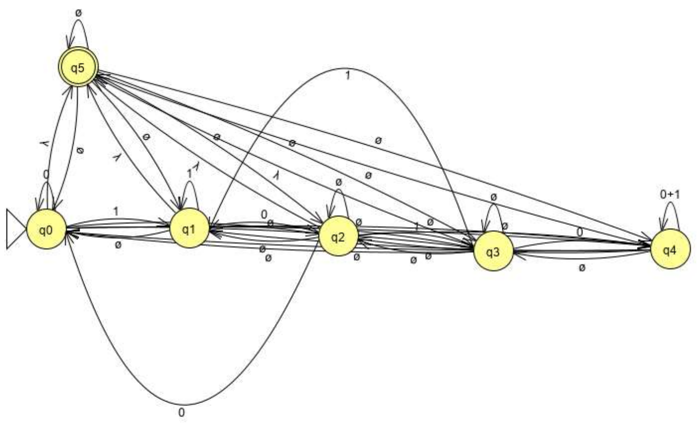
v. *Convert one of the machines into a regular expression*
DSFM:



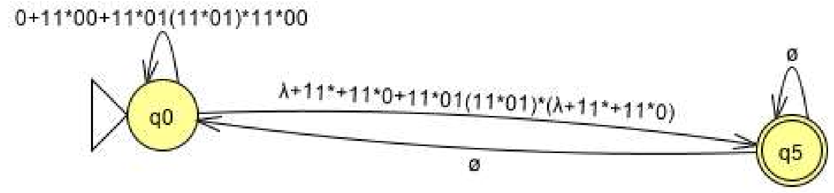
Create single Noninitial final state and put λtransitions from the old final states → new final states:



Collapse multiple transitions, add empty transitions between all states that don't have transitions between them



Remove nonfinal, noninitial states and replace the transitions with a regex:



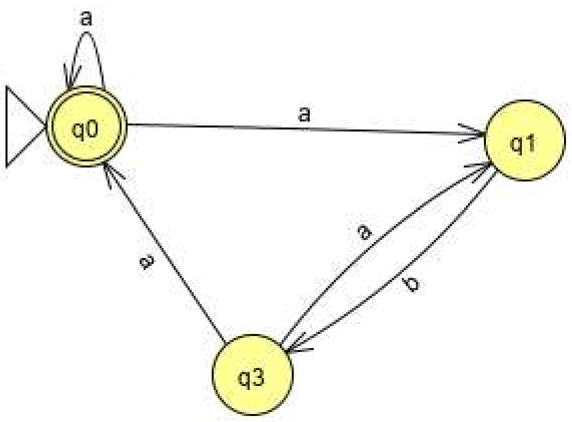
REGEX: (0+11*00+11*01(11*01)*11*00)*(λ+11*+11*0+11*01(11*01)*(λ+11*+11*0))

Simplified:
(0+11*0(0+1(11*01)*11*00))*(λ+1(1*+1*(0+01(11*01)*(λ+1(1*+1*0)))))

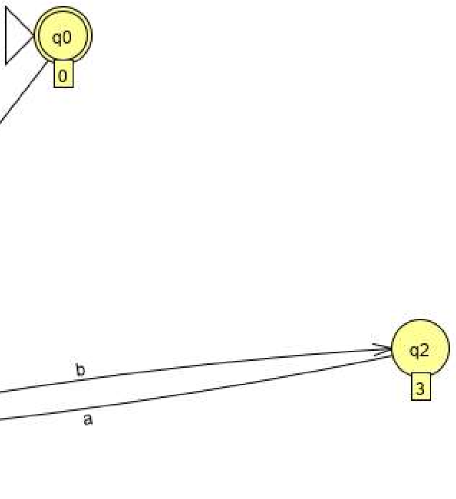
d. $\{w \in \{a,b\}^* \mid \text{ every } b \text{ in } w \text{ is immediately preceded and followed by } a\}.$

i. *Prove whether it is regular or not*
   Proof by NDFSM.

ii. *Construct a NDFSM*



iii. *Convert NDFSM → DFSM*



iv. *Minimize DFSM*
   This DFSM is already minimized. One way to argue this is to show |K| = # of equivalence classes.
   There are 3 collectively exhaustive equivalence classes of $\approx L.$:

   - In L. All the strings in L would be accepted if we added a, and all would be rejected if we added b, and would be accepted if we added the empty string..
   - Ending in b, in L otherwise. All the strings ending in b but that would be in L otherwise would be accepted if we added an a, rejected if we added a 'b' or the empty string.
   - Have one or more 'b's that are not preceded by an a. No matter what we add, the strings in this class will always be rejected.

   Since there are 3 equivalence classes, therefore the minimal DFSM has 3 states. Therefore, this DFSM is already minimal.

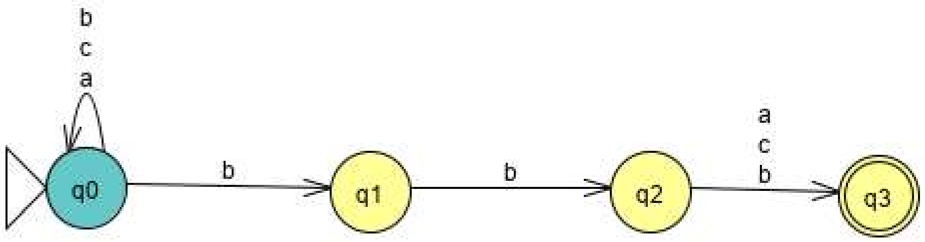
v. *Convert one of the machines into a regular expression*
   (a u ab(ab)*a)*
   = (a*(ab(ab)*a)*)*

e. $\{w \in \{a, b, c\}^* \mid \text{ the third and second from the last characters are } b\text{'s}\}$

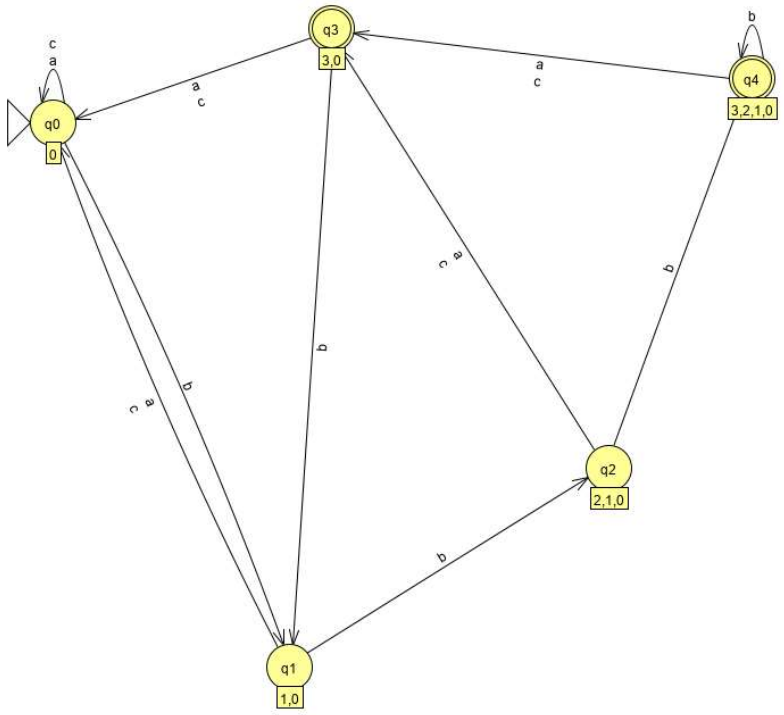
i. Prove whether it is regular or not

Proof by NDFSM.

ii. Construct a NDFSM

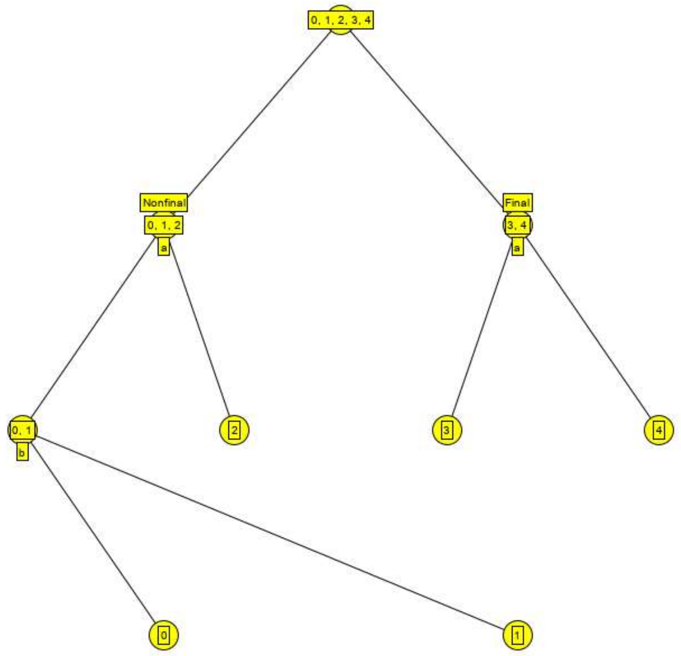


iii. Convert NDFSM → DFSM



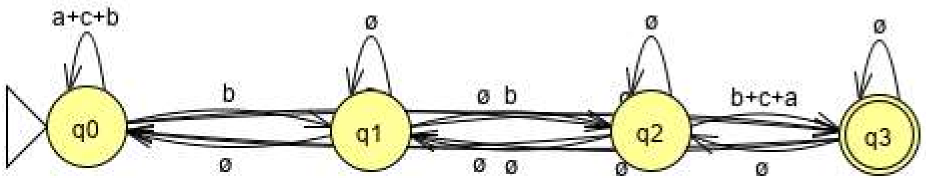
iv. *Minimize DFSM.*

This DFSM is already minimal as the overclustering algorithm made no changes to it. The fact that this DFSM is minimal can be illustrated through this diagram representing the overclustering process, that shows there is a path to each state (illustrated by each state being represented in a leaf node) and that each state is unique (illustrated by each leaf node only containing one state):

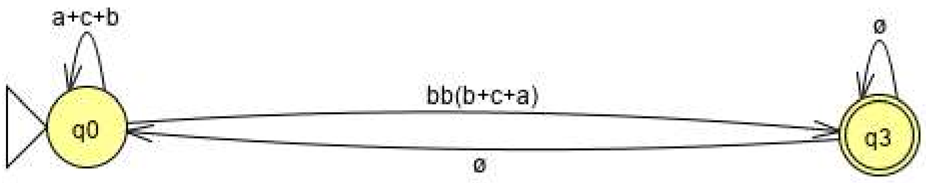


v. *Convert one of the machines into a regular expression*

Collapse multi-transitions and add empty transitions between stages that don't have a transition between them already:



Remove nonfinal, nonintial states (rip process):



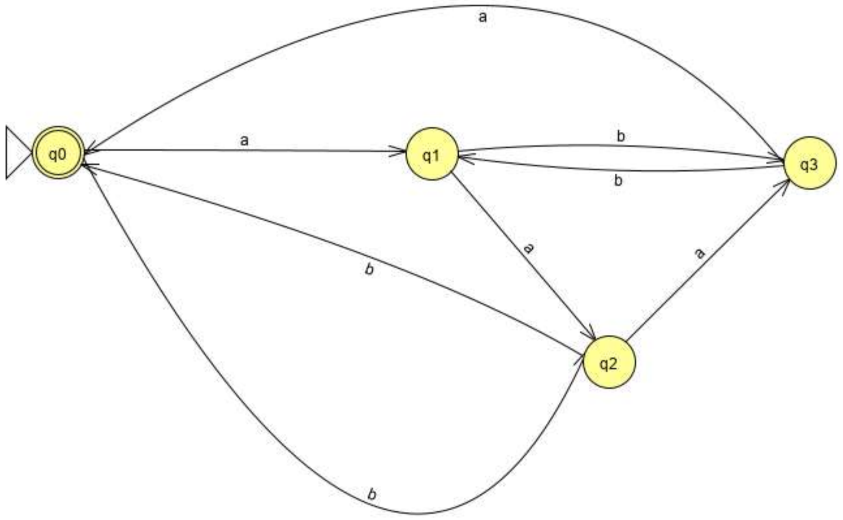
Regular expression: (a+c+b)*bb(b+c+a)

f. $\{w \in \{a,b\}^* \mid (\#_a(w) + 2\#_b(w)) \equiv 0 \pmod 4\}$. ($\#_a(w)$ is the number of a's in w).

i. *Prove whether it is regular or not*
Proof using DFSM.

ii. *Construct a NDFSM*
Constructed DFSM without NDFSM

iii. *Convert NDFSM → DFSM*



iv. *Minimize DFSM*
The DFSM is already minimal. I will argue this by showing that the number of equivalency classes = |K|. There are four equivalency classes of $\approx L$:

- num(a) + 2(num(b)) $\equiv 0 \; mod \; 4$
- num(a) + 2(num(b)) $\equiv 1 \; mod \; 4$
- num(a) + 2(num(b)) $\equiv 2 \; mod \; 4$
- num(a) + 2(num(b)) $\equiv 3 \; mod \; 4$

Since this DFSM has 4 states, and there are 4 equivelency classes, therefore this DFSM is minimal.

v. *Convert one of the machines into a regular expression*
((b U aa)b U (ab U (b U aa)a)(b(b U aa))(a U bab))**

g. $\{w \in \{a,b\}^* \mid \#_a(w) - 2\#_b(w) = 0\}$

i. Prove whether it is regular or not

Language is not regular. Proof by contradiction of Pumping Lemma.

$Let\ w = a^{2k}b^k; w \in L$
$Let\ x = a^i, y = a^j, z = a^{2K-i-j}b^k$
Assume L is regular. Then:
$Let\ w = xyz;$ x, y, z ∈ Σ^*
$For\ some\ k \geq 1, all\ q \geq 0, xy^qz \in L, |xy| \leq k$
$Let\ q = 2, let\ j = 1, let\ i = k - 1$. Then:
$xy^kz = a^{k-1}a^2a^{2K-k+1-1}b^k$
$xy^kz = a^{2k+1}b^k$

Therefore, there does not exist a k >= 1 s.t. for all q >= 0, $xy^qz \in L$
Therefore, this contradicts the pumping theorem
Therefore, this language is not regular.

ii. Construct a NDFSM
iii. Convert NDFSM → DFSM
iv. Minimize DFSM
v. Convert one of the machines into a regular expression

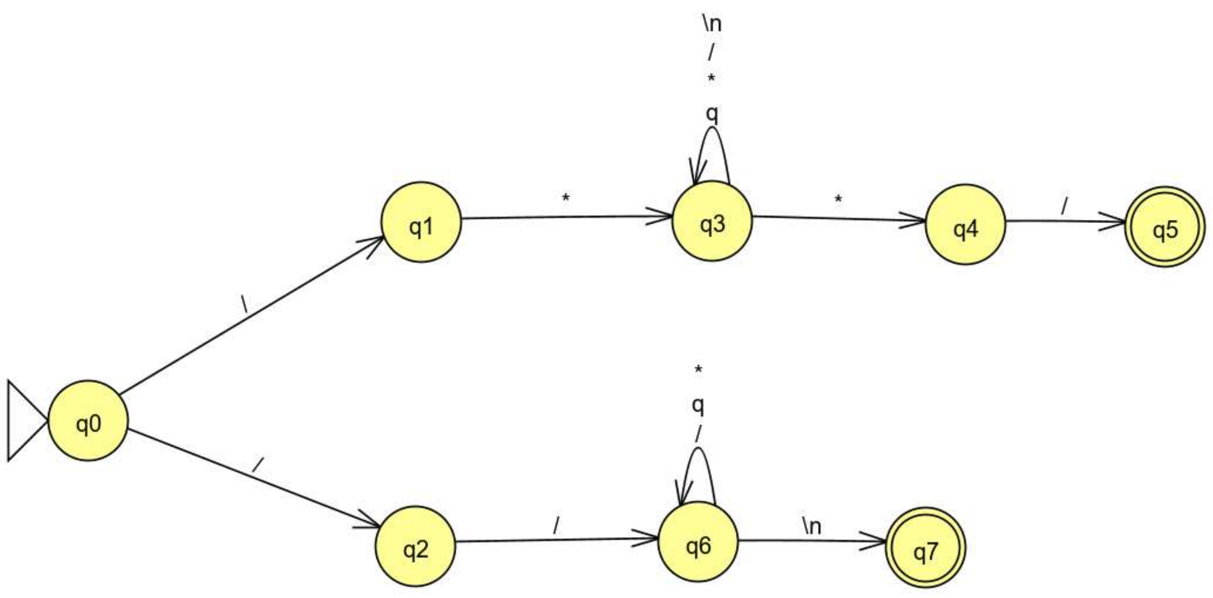
h. $\{w \in \Sigma^* \mid w \text{ is a C comments}\}$.

Assumptions:

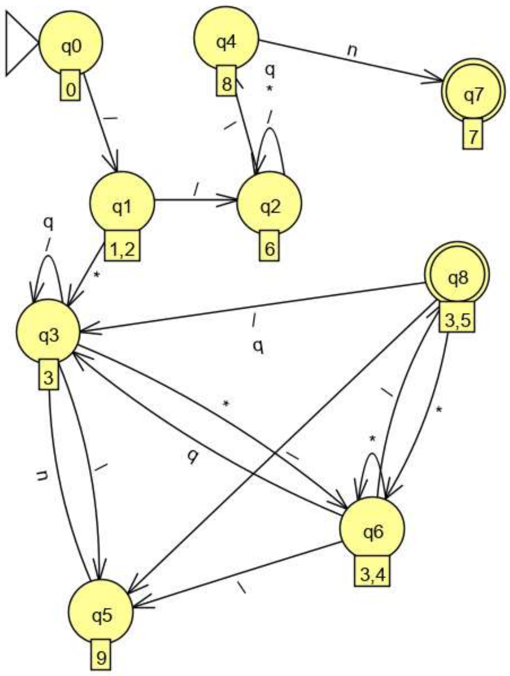
These are C comments so recognize nested comments aren't recognized. This language wouldn't be regular if nested comments were recognized.

I interpreted '\n' as one character, because typing '\' then typing 'n' won't close a single-line comment in C.

i. *Prove whether it is regular or not*
   Proof by NDFSM.
ii. *Construct a NDFSM* – **Let q represent all characters typeable from keyboard EXCEPT \n (enter key), /, ***

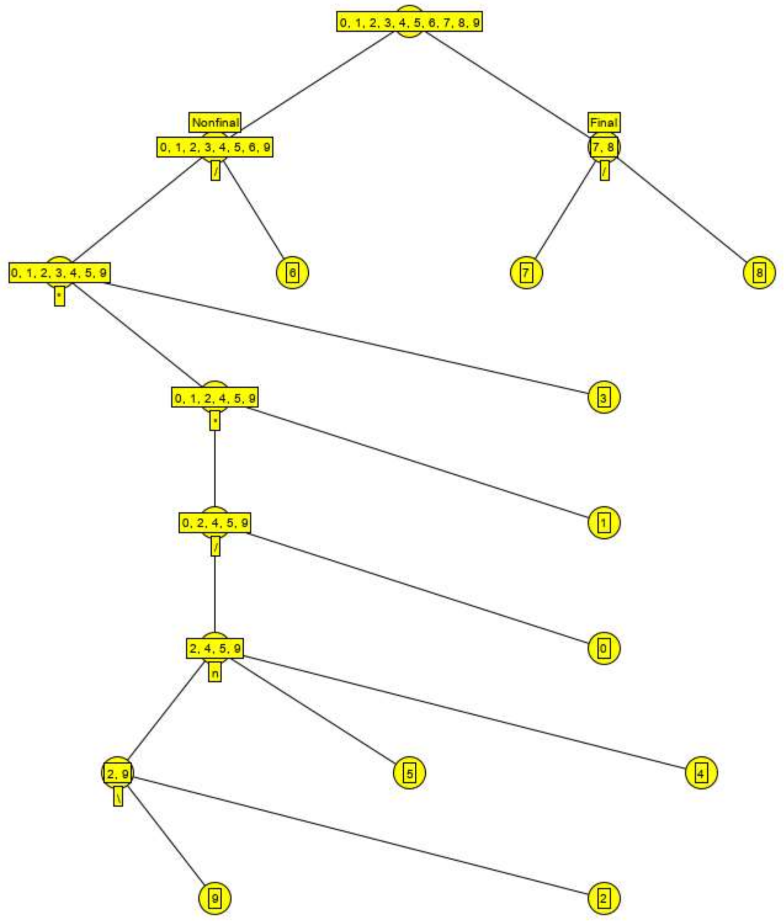


iii. *Convert NDFSM → DFSM*



iv. Minimize DFSM

This DFSM is already minimal as the overclustering algorithm made no changes to it. The fact that this DFSM is minimal can be illustrated through this diagram representing the overclustering process, that shows there is a path to each state (illustrated by each state being represented in a leaf node) and that each state is unique (illustrated by each leaf node only containing one state):



v. Convert one of the machines into a regular expression

/*(q+*+/+\n)**/+//(/+q+*)*\n

I changed the asterisks to dollar signs as to not confuse them with Kleene Star:

/$(q+$+/+\n)*$/+//(/+q+$)*\n

From above: q represents every typeable character other than \n, /, *