UNIT-1

Worksheet-1

Mathematical Concepts

MCQ

1. According to principle of mathematical induction, if P(k+1) = m(k+1) + 5 is true then _____ must be true.

a) $P(k) = 3m^{(k)}$

b) $P(k) = m^{(k)} + 5$

c) $P(k) = m^{(k+2)} + 5$

d) $P(k) = m^{(k)}$

2. A proof that $p \rightarrow q$ is true based on the fact that q is true, such proofs are known as ___________

a) Direct proof
b) Contrapositive proofs
c) Trivial proof
d) Proof by cases

3. For any positive integer m ______ is divisible by 4.

a) $5m^2 + 2$
b) $3m + 1$
c) $m^2 + 3$
d) $m^3 + 3m$

4. onlyhe "-if-part" of the statement of "H if and only if S" is __________.
a) if S then H
b) if not S then H.
c) if H then S
d) if not S then not H.

DESCRIPTIVE QUESTIONS

1. Show that 2^{2n}-1 is divisible by 3 using the principles of mathematical induction.
2. Prove that if for an integer a, a^2 is divisible by 3, then a is divisible by 3 using the proof by contradiction.
3. For any two integers a and b, (a+b) is odd if and only if exactly one of the integers a or b is odd. Prove the above statement.
4. notShow by counter example the given statement P is always true.

   $P = 2n^2-16n+31$ is always positive for all of n.
5. Prove using mathematical induction for n>=5, $2n>n^2$.
6. Prove that the sum of n squares can be found as follows
   $1^2+2^2+3^2+...+n^2=n(n+1)(2n+1)/6$

UNIT-1

Worksheet-2

1.The Regular expression101*10 is generically stated as

(i) Set of all strings starting and ending with ‘10’ and any number of 1’s in between ‘10’.

(ii) Set of all strings starting with ‘10’ and ending with ‘10’.

(iii) Set of all strings starting with ‘10’ and ending with ‘10’ and ‘1’ in between them.

(iv) Set of all strings starting with ‘10’ and ending with ‘10’ and ‘10’ in between them.

2. Choose the RE for a language of any combination of 0’s & 1’s containing 1001 as a substring

(i) L=(01)*1001(01)*

(ii)L=(0+1)*1001(0+1)*

(iii)L=(01)*1001(0+1)*

(iv)L=(0+1)*1001(01)*

3. Which pair is equivalent regular expression?

(i) (ab)* and a*b* (ii) a(aa)* and (aa)*a (iii) a+ and a*a

a. Only (i)
b. Only (ii)
c. (ii) and (iii)
d. (i)(ii) and (iii)

4. NFA’s accept

i. **Regular Languages**
ii. More languages than a DFA can accept
iii. Languages that are not regular
iv. Context Free Languages

5. Which one of the following languages over the alphabet {0, 1} is described by the regular expression (0 + 1) *0(0 + 1) *0(0 + 1) *?

(a) The set of all strings containing the substring 00
(b) The set of all strings containing at most two 0’s
(c) The set of all strings containing at least two 0’s
(d) The set of all strings starting and ending with 0 or 1

Part-B

1.Describe the Language generated by the following Regular Expression (0)*(101)*11.

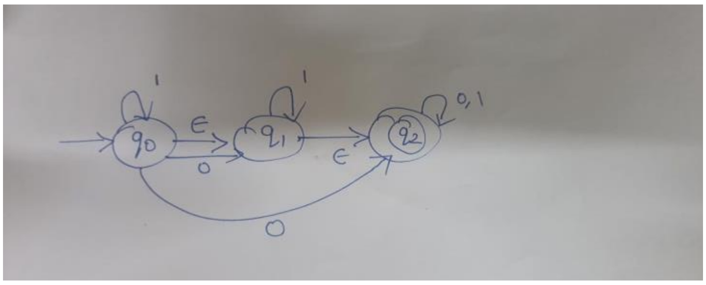
2. **Identify the Regular Expression for the following:**

A language consists of any combination of 0’s & 1’s, beginning and ending with the string ‘01’.

3.Justify whether Regular expression exist for the following scenario.

Seetha wants to write the Regular expression for the set of all strings which contain repeated substrings of any length>1[E.g., “aba” Substring ‘a’ Repeats].

4.Recognize the term Epsilon (ε) – closure. Identify the Epsilon (ε) closure of the state qo in the following NFA.



5.Memorize the 5 tuple structure of DFA and NFA.

UNIT-1

Worksheet-3

Construction of DFA, NFA, ε-NFA and equivalence of NFA and DFA

1. What is the minimum number of states to recognise the language L={w/w ϵ (0+1+2)+}?
   a) 1 b) 2 c) 3 d) 4
2. What is the minimum number of states required by the DFA that accepts the language? L={a | a is a number divisible by n} ?
   a) n b) n+1 c) n-1 d) 2^n
3. _______ is the maximum number of states that an ε-NFA can have on ε moves.
   a) n
   b) 0
   c) Infinite
   d) 1f

4. The FSA to recognize the words "infrared" and "infrastructure" has _____ number of states.

   a) 20
   b) 22
   c) 15
   d) 17

5. NFA with ϵ transitions _______

a) Increases computations

b) Decreases computations

c) Decreases number of states

d) Increases uncertainty

6. What are the maximum number of output states for any input state (n) in a NFA?

a) n

b) n+1

c) 2n

d) n-1

7. I: DFA's can be constructed for all the languages

II: The strings accepted by DFA will be accepted by NFA

What can be said about these two statements?

a) Only II is false

b) Only I is false

c) I is false and II is true

d) II is true and I is false

8. What can be told about the recognising capability of NFA, ε-NFA and DFA?

a) All three are equally powerful

b) ε-NFA is more powerful and flexible

c) ε-NFA is less powerful and flexible

d) DFA is more powerful

9. What is the minimum number of states for NFA that accepts the language {01n 01 | n >=0}?

a) 5

b) 4

c) 6

d) 16

10. Which of the given languages are accepted by Non Deterministic PDA but not by Deterministic PDA?

a) Language generating strings that contain at least one symbol repeated at least twice

b) Even Palindromes

c) Strings ending with a particular symbol

d) Strings starting with particular symbol

PART-B

1. Construct a DFA that can recognise the six-symbol password over the input Σ={a, b, c} with the following conditions:
   a) Password should start with 'ab'
   b) Password should not end with 'bb'.
2. Construct a DFA that accepts the numbers that are multiples of five in its binary form.
3. Construct a DFA and NFA that accepts strings that starts with 'abb' and ends with any number of 'a'.
4. Ramesh has to create an FSA that accepts string over {a, b, c} in such a way that the fourth symbol from the right is always 'c'. Can he construct both NFA and DFA? Justify your answer.
5. Is it possible to create an NFA and ε-NFA over {0,1} that accepts L={0^n 1^m 2^o| n, m, o >=0}? If so, give the construct.
6. Is it possible to create an NFA and ε-NFA over {0,1} that accepts L={0^n 1^m 2^o| n, m, o >0}? If so, give the construct.

7. Design a NFA that recognises the strings ‘abc’, ‘abd’, ‘aacd’ over the input Σ={a, b, c, d}.
8. Convert the following NFA to DFA:

δ	0	1
->Q0	{Q1, Q2}	{Q0}
Q1	{Q1, Q2}	Φ
*Q2	Q1	{Q1, Q2}

18CSC301T -Formal Language and Automata Theory

Worksheet 4 – Unit 1, Sessions - S7 and S8

Topics: ∈-NFA and NFA equivalence, DFA-NFA equivalence, DFA minimization.

Part – A

1. A language if accepted by DFA, will be accepted by the equivalent NDFA also. Justify your answer. 2. If a NFA has n states, the the corresponding DFA has _____ states

3. In Qx∑ = P(Q), the power set of Q has a maximum of __________ elements for a DFA and _____________ elements for a NFA.

4. For NFA with epsilon moves, which of the following is correct?

a) ∂ : Qx∑ = P(Q)

b) ∂ : Qx (∑ U $∑^+$) = P(Q)

c) ∂ : Qx($∑^+$) = P(Q)

d) ∂ : Qx(∑ U ∈) = P(Q)

5. The final state(s) of a NFA converted from the corresponding ∈-NFA is

a) same final state as ∈-NFA

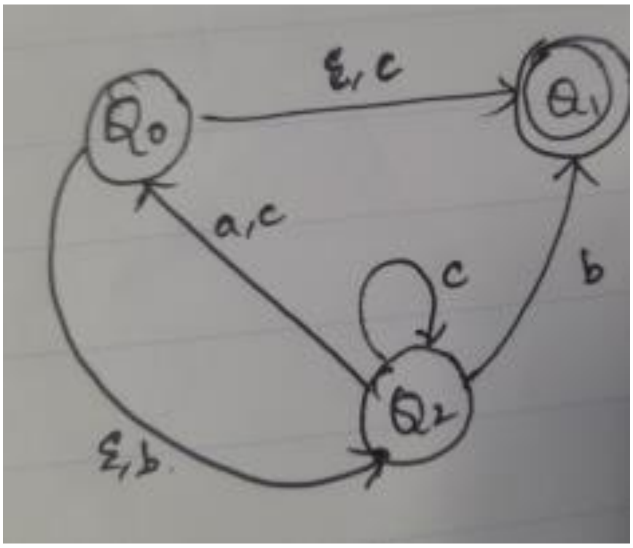
b) ∈-closure (final state of NFA)

c) All states that can reach the final state of ∈-NFA only by seeing an ∈

d) final state as the corresponding MDFA

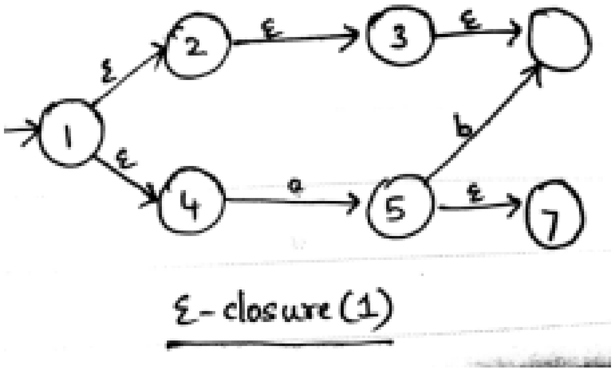
Part – B

1. Convert the below NFA to a DFA with minimal states.



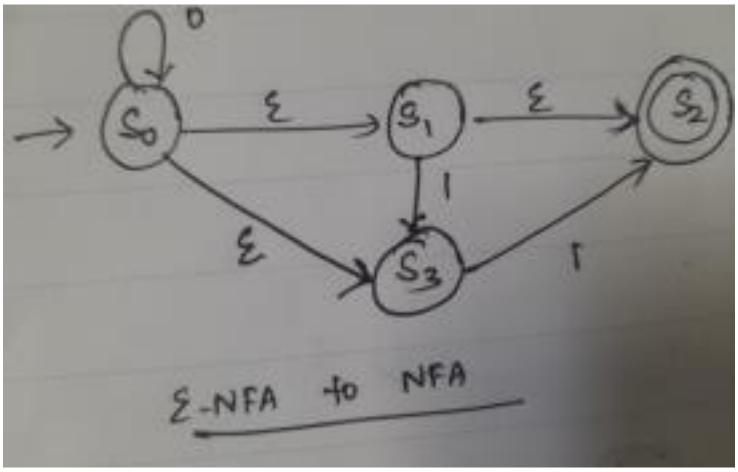
2. Given L_1=01* and L_2 = 0*1, M1 and M2 are the machines recognizing L1 and L2 respectively as shown in the figure below, design automata recognizing L_1 U L_2, L_2 and L_1*.

3. Find the epsilon closure(1) in the following ∈-NFA.

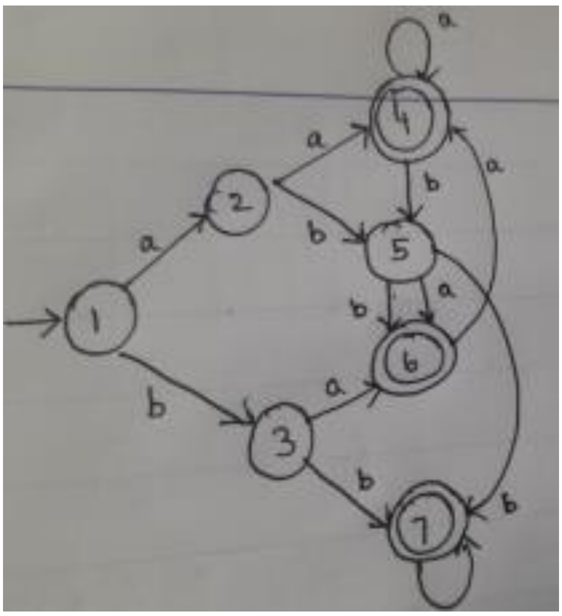


4. Convert a DFA that accepts binary number 9 to a NFA that accepts both binary equivalents of both the numbers 5 and 9.

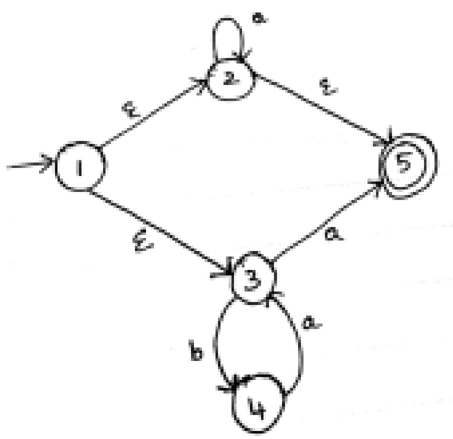
5. Convert the ∈-NFA in the below figure to a NFA .



6. Check whether the below DFA is minimized, if not minimize it.



7. Find the minimum state DFA accepted by the following ∈-NFA.



a
2
ε
ε
1
5
ε
a
3
a
b
4

UNIT-1

Worksheet-5

Problems related to Equivalence of Finite Automata and Regular Languages and Regular Grammars

PART A

1. Regular expression for all strings starts with ab and ends with bba is.
   a) aba*b*bba
   b) ab(ab)*bba
   c) ab(a+b)*bba
   d) All of the mentioned
2. Under which of the following operations, NFA is not closed?
   a) Negation
   b) Kleene
   c) Concatenation
   d) None of the mentioned
3. Ragu is asked to make an automaton which accepts a given string for all the occurrences of ‘1001’ in it. How many number of transitions would John use such that the string processing application works?
   a) 9
   b) 11
   c) 12
   d) 15

4. Which of the following does not represent the given language? Language: {0,01}
   a) 0+01
   b) {0} U {01}
   c) {0} U {0}{1}
   d) {0} ^ {01}

5. Which among the following looks similar to the given expression?
   ((0+1). (0+1)) *
   a) {xϵ {0,1} *|x is all binary number with even length}
   b) {xϵ {0,1} |x is all binary number with even length}
   c) {xϵ {0,1} *|x is all binary number with odd length}
   d) {xϵ {0,1} |x is all binary number with odd length}

6. RR* can be expressed in which of the forms:
   a) R+
   b) R-
   c) R+ U R-
   d) R

7. Which of the following represents a language which has no pair of consecutive 1’s if ∑= {0,1}?
   a) (0+10)*(1+ε)
   b) (0+10)*(1+ε)*
   c) (0+101)*(0+ε)
   d) (1+010)*(1+ε)

8. Let the class of language accepted by finite state machine be L1 and the class of languages represented by regular expressions be L2 then
   a) L1<L2
   b) L1>=L2
   c) L1 U L2 = .*
   d) L1=L2

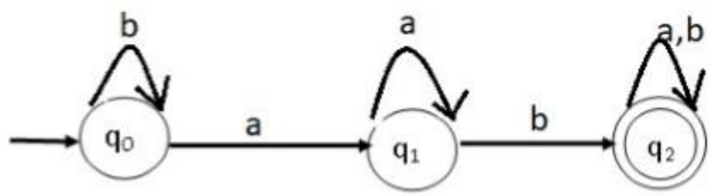
9. Let N (Q, ∑, δ, q0, A) be the NFA recognizing a language L. Then for a DFA (Q’, ∑, δ’, q0’, A’), which among the following is true?

a) Q’ = P(Q)
b) Δ’ = δ’ (R, a) = {q ϵ Q | q ϵ δ (r, a), for some r ϵ R}
c) Q’ = {q0}
d) All of the mentioned

10. If L1 and L2′ are regular languages, L1 ∩ (L2′ U L1′)’ will be
a) regular
b) non regular
c) may be regular
d) none of the mentioned

PART-B

1. Describe a Regular Expression. Write a Regular Expression for the set of strings that consists of alternating 0’s and 1’s.
2. Examine whether the language L=(0^n 1 n | n>=1) is regular or not? Justify your answer.
3. Construct Finite Automata equivalent to the regular expression (ab+a)*
4. Construct NDFA for given RE using Thomson rule.
   i) a(a+b)* ab
   ii) (a.b)*
   iii) (a+b)
5. Find the Regular Expression equivalent for the given Finite Automata.



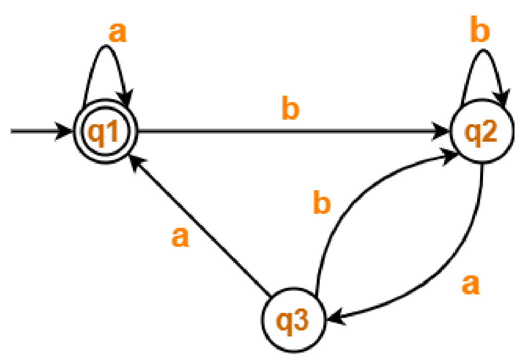
6. Evaluate the equalities for the following RE and prove for the same

   (i) b+ab* +aa*b+aa*ab*

   (ii) a*(b+ab*).

   (iii) a(a+b)*+aa(a+b)*+aaa(a+b)*

7. Find the Regular Expression equivalent for the given Finite Automata.



8. Construct a DFA which is equivalent to the following regular expression:
   00 ∪ (1 ∪ 01)(11 ∪ 0)∗10 ∗

UNIT-1

Worksheet

Question 1:

"The Enchanted Forest Maze"

In a mystical land, there lies an Enchanted Forest Maze. The maze is said to have magical properties, and it changes its pathways whenever someone enters it. Inside the maze, explorers can find hidden treasures, but only if they can navigate the maze correctly.

The maze is represented as a set of interconnected rooms, and each room has two doors: one labeled '0' and the other labeled '1'. When a brave explorer enters a room, they face a decision. They can choose to go through the door labeled '0', which leads to one or more other rooms , or they can go through the door labeled '1', which leads to a different set of rooms.

The initial starting point for all explorers is a room called "The Foyer", represented by the state q0. From "The Foyer," explorers can choose either door '0' or door '1', and they may face different options in their journey.

The transition rules for the Enchanted Forest Maze are as follows:

From "The Foyer" (q0), if an explorer chooses door '0', they may find themselves in either "The Foyer" again (q0) or in "The Magic Garden" (q1).

From "The Foyer" (q0), if an explorer chooses door '1', they may end up in "The Dark Cave" (q2).

From "The Magic Garden" (q1), if an explorer chooses door '1', they will find themselves in "The Enchanted Lake" (q2).

From "The Dark Cave" (q2), regardless of the chosen door, there are no further transitions.

The explorers aim to ensure their successful reach of the treasures hidden in "The Enchanted Lake" by transforming the mystical maze into a deterministic model. They intend to avoid making any unambiguous decisions on which door to choose, considering the input they encounter (0 or 1) in each room.

Question 2:

Scenario: "The Time Travel Machine"

In the secret laboratory of a brilliant scientist, there is a mysterious invention called "The Time Travel Machine." This machine has the ability to transport people through time, allowing them to explore various historical eras. However, the machine's operation is controlled by a complex non-deterministic finite automaton (NFA).

The NFA, inside the Time Travel Machine, has three states: "The Present" (P), "The Past" (T), and "The Future" (F). When someone activates the machine, they find themselves in "The Present" (P) state. From there, they have two choices: they can input either '0' or '1' into the machine.

The transition rules for The Time Travel Machine's NFA are as follows:

From "The Present" (P), if someone enters '0', they may either stay in "The Present" (P) again or travel to "The Past" (T).

From "The Present" (P), if someone enters '1', they will be transported to "The Future" (F).

From "The Past" (T), if someone enters '1', they will find themselves back in "The Present" (P).

From "The Future" (F), regardless of the input ('0' or '1'), there are no further transitions.

The scientist is eager to convert the given NFA transition rules into a deterministic finite automaton (DFA) to ensure more controlled and predictable time travel experiences. Your task, as a talented student, is to help the scientist convert the NFA into a DFA transition table.

Question 3:

In a popular social networking platform, users are required to create usernames according to a set of criteria. The username must start with the character 'p', followed by any combination of 'p' and 'q' (including an empty string), and it must end with the characters 'qp'. To ensure that usernames adhere to these rules, develop a deterministic model that accepts valid usernames and rejects invalid ones.

Question 4:

In a binary data processing application, users need to input binary strings following specific criteria. The binary string should only consist of the characters '0' and '1' and must contain the subsequence "10" at some point within the string. To validate these binary strings, design a deterministic finite automaton (DFA) that accepts valid strings and rejects invalid ones.

Question 5:

You are developing a web application that validates and processes URLs submitted by users. The URLs must follow a specific format to be considered valid.

The language L represents the set of valid URLs that adhere to the following pattern:

A URL starts with the protocol (e.g., "http://" or "https://").

Following the protocol, there is the domain name, which can include alphanumeric characters and can have multiple levels, separated by periods (e.g., "www.example.com" or "subdomain.example.com").

After the domain name, there can be an optional path, representing specific pages or resources on the website (e.g., "/page" or "/directory/page.html").

Additionally, the URL can include query parameters, represented by a question mark followed by key-value pairs (e.g., "?key1=value1&key2=value2").

The language L is defined as follows:

L = {protocol://domain[optional_path][query_parameters] | protocol is a valid protocol, domain is a valid domain, optional_path is a valid path (optional), and query_parameters are valid key-value pairs (optional)}

For example, some strings in L are: "http://www.example.com", "https://sub.domain.com/directory/page.html", "https://example.com/?key=value", etc.

Question:

Assuming that the language L of valid URLs is regular, apply the Pumping Lemma to prove that the language is not regular. Specifically, choose a sufficiently long URL from L and demonstrate that it cannot be pumped while remaining in L.

Hint:

To apply the Pumping Lemma, consider a long valid URL from L and break it down into three parts: s = xyz, such that the conditions of the lemma are satisfied. Then, examine what happens when you pump the middle part 'y' up or down in length. Does the resulting string remain in the language L, representing a valid URL?

Question 6:

Scenario: Regular Language of Valid Parentheses

You are working on a compiler for a programming language, and you need to validate parentheses expressions in the source code. The expressions can contain opening and closing parentheses and must be balanced.

The language L represents the set of valid parentheses expressions that adhere to the following pattern:

A valid parentheses expression can contain any number of opening '(' and closing ')' parentheses.

The parentheses must be balanced, meaning that each opening '(' must have a corresponding closing ')', and they must appear in the correct order.

The language L is defined as follows:

L = { w | w is a balanced parentheses expression }

For example, some strings in L are: "()", "(())", "(()(()))", "()(())", etc.

Question:

Assuming that the language L of valid parentheses expressions is regular, apply the Pumping Lemma to prove that the language is not regular. Specifically, choose a sufficiently long parentheses expression from L and demonstrate that it cannot be pumped while remaining in L.

Hint:

To apply the Pumping Lemma, consider a long valid parentheses expression from L and break it down into three parts: s = xyz, such that the conditions of the lemma are satisfied. Then, examine what happens when you pump the middle part 'y' up or down in length. Does the resulting string remain in the language L, representing a balanced parentheses expression?

Suppose you have a stack of identical-sized building blocks. The blocks are arranged in a straight line, and you want to prove that you can build a tower with 'n' blocks, such that each block is placed directly on top of the previous one.

Question 7:

Using mathematical induction, prove that you can build a tower with 'n' blocks in such a way that the height of the tower is the sum of the heights of the individual blocks (assuming each block has a height of 1 unit).

In other words, prove that the height of a tower with 'n' blocks is given by the formula:

TowerHeight(n) = n

Question 8:

You have a collection of postage stamps of two denominations: 3 cents and 5 cents. You want to prove that it's possible to form any postage amount of 8 cents or more using only these two denominations.

Using mathematical induction, prove that for any natural number 'n' greater than or equal to 8, you can form the postage amount 'n' using only 3-cent and 5-cent stamps.

Hint:

Base Case

For n=8, you can form the amount using two 3-cent stamps: 3 + 3 + 2 = 8. This matches our initial condition.

Inductive Hypothesis

Assume that it's possible to form the postage amount 'k' using only 3-cent and 5-cent stamps, for some arbitrary natural number 'k' where k >= 8.

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
DEPARTMENT OF COMPUTATIONAL INTELLIGENCE
18CSC301T- FORMAL LANGUAGE AND AUTOMATA

Worksheet 4 - Pumping lemma

(i) Each street will be given a unique symbol.

(ii) Yes, because normally in roadway networks there are cycles.

(iii) L = { uxw | u and w denote strings of symbols, x denotes Mount Road and |u| = |w| and uxw denotes a path from central station to Marina beach through Mount Road, and no two adjacent symbols are equal further u and w do not contain x}

(iv) Pumping lemma for regular languages.

(v) We apply pumping lemma as an adversarial argument.

Player 1 chooses to prove L is non-regular

Player 2 chooses pumping length k.

Payer 1 picks uxw such that |u| >= k.

Player 2 breaks up uxw as abc such that |b| >= 1

Player 1 claims that ab^2c is not in L

Case 1 if b is partially in u and includes x, then b^2 includes two x's and is therefore $ab^2c \notin L$.

Case 2 if b is partially in w and includes x, then argument is similar to Case 1

Case 3 partially in u, w, and includes x, argument is similar to case 1.

Case 4 if b is entirely in the u part, then ab^2c would increase the length of the part before mount road, so $ab^2c \notin L$.

Case 5 if b is entirely in the w part, then ab^2c would increase the length of the part after mount road, so $ab^2c \notin L$.

Case 6 if b = x, then ab^2c has two x, so $ab^2c \notin L$.

In each case we have proved $ab^2c \notin L$, therefore L is not a regular language.

(vi) We assumed that there is a uxw such that |u| >= k.

Worksheet 5 - Properties of regular languages

(i) Non-empty. Because ‘0’ belongs to the language
(ii) DFA for complement - simply make q0 final state and q1 non-final state.
(iii) DFA for reversal - first reverse the arcs, interchange start state and final state, you will get NFA. Then convert to DFA. You will get DFA below. (q0 and q1 do not correspond to q0 and q1 of original DFA)

	0	1
→q0	q1	q2
*q1	q1	q1
q2	q2	q2

(iv) DFA. In linear time you can test for membership just by following transitions and testing if you reached final state. For NFA, you will take polynomial time because there may be ∈-transitions and you need to compute next state from all current states. Regular expression you need to convert to NFA before you test for membership.

(v) DFA/NFA to regular expressions is most difficult to perform because we increase the size of the regular expressions 4 fold with each iteration, and we have n^3 expressions for a total time of $O(n^3 4^n)$

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
DEPARTMENT OF COMPUTATIONAL INTELLIGENCE
18CSC301T- FORMAL LANGUAGE AND AUTOMATA

WORKSHEET - 1

1. Prove that the sum of cubes of n natural numbers is equal to ([n(n+1)]/2)2 for all n natural numbers with mathematical induction.

2. By mathematical induction show that 1 + 3 + 5 + ... + (2n−1) = n2.

3. Distinguish between inductive proof and formal proof.

4. Write down the regular expression to denote a language L which accepts all the strings which begin or end with either 00 or 11.

5. Construct a regular expression which accepts all the strings with at least 2 b's over the

   ∑ ={a,b}.

6. Write the regular expression to denote the Language L over ∑ ={a,b} such that all the strings do not contain the substring a,b.

7. Write a regular expression to denote the language L over ∑ ={a,b} such that 3 character from the right end of the string is always "a".

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
DEPARTMENT OF COMPUTATIONAL INTELLIGENCE
18CSC301T- FORMAL LANGUAGE AND AUTOMATA
WORKSHEET - 2

1. Write the Regular Language for the following regular expression

i) R= {a * b* c}

ii)R= {1(0+1)* 0 }

2. Describe the language denoted by the following regular expression:

R = {(b*(aaa)*b*)*}

3. Draw a FA which checks whether the given binary number divisible by 4 and give the acceptance for the input sequence 11000.

4. Design a FA that reads a string made up of letters in the word "AUTOMATION" and recognise those strings that contain the word "MAT" as a substring.

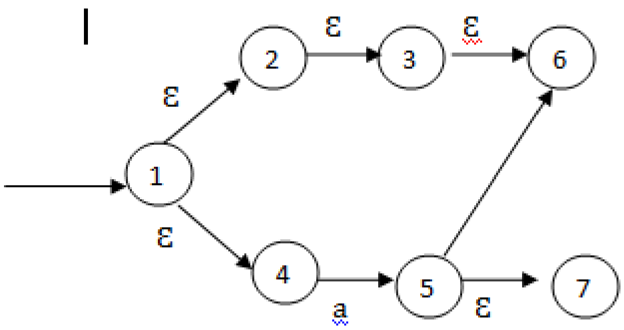
5. Design a DFA which accepts all the strings not having more than two a's over ∑ ={a,b} and convert the DFA diagram in to transition table.

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
DEPARTMENT OF COMPUTATIONAL INTELLIGENCE
18CSC301T- FORMAL LANGUAGE AND AUTOMATA

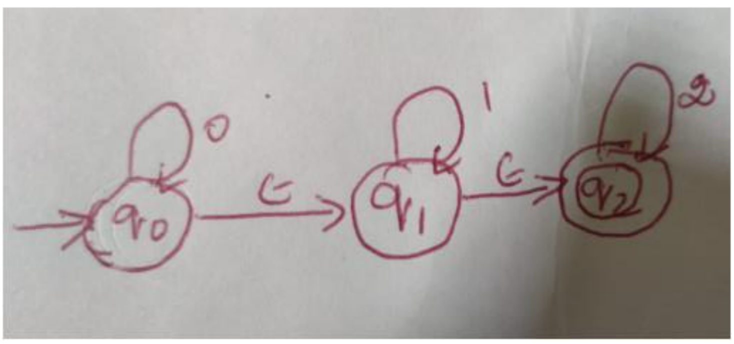
WORKSHEET - 3

1. Construct a NFA for even number of 0's or odd number of 1's also give the Regular expression for even number 0's or odd number of 1's.

2 . Find the Ɛ-closure of the all states in the following transition diagram.



3. Convert the given NFA with Ɛ to NFA without Ɛ



4. Draw the NFA for the given string ab (a+b) * using Thomson construction method.

5. Convert the given NFA to DFA.

States/inputs	**0**	**1**
-> q0	{q0,q1}	{q0}
Q1	{q2}	{q1}
Q2	{q3}	{q3}
Q3 [Final state]	φ	{q2}

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
DEPARTMENT OF COMPUTATIONAL INTELLIGENCE
18CSC301T- FORMAL LANGUAGE AND AUTOMATA
WORKSHEET – 4

Pumping Lemma

Consider that you are given the map of Chennai.

(i) How would you encode all possible paths (sequences of roads along path) from Central Station to Marina Beach using a language?

(ii) Do you think the language of (i) contains / does not contain an infinite number of strings and why / why not?

(iii) You are told to encode all paths from Central to Marina that pass-through Mount Road once with an equal number of roads traversed before and after Mount Road. Write the formal notation for such a language.

(iv) What approach would you use to prove that the language in (iii) is not regular?

(v) Describe the approach to prove that language in (iii) is not regular.

(vi) State any assumptions you made while proving (v)

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
DEPARTMENT OF COMPUTATIONAL INTELLIGENCE
18CSC301T- FORMAL LANGUAGE AND AUTOMATA
WORKSHEET – 5

Properties of Regular languages

Consider the following DFA shown in table below:

	0	1
→q0	q1	q0
*q1	q1	q1

(i) Is the language of the DFA empty or non-empty? Substantiate.

(ii) Obtain the DFA for complement of the language.

(iii) Obtain the DFA for reversal of the language.

(iv) Among the 3 models that can be used to represent regular languages, which one is most suitable to test for membership and why?

(v) Among the 3 models used to represent regular languages, which interconversion between the models is difficult to perform? Substantiate.

Worksheet 1

(1) Prove that the sum of cubes of 'n' natural numbers is equal to $\left([n(n+1)]/2\right)^2$ for all n natural numbers.

Soln:

In the given statement we are asked to prove:

$$1^3+2^3+3^3+\dots+n^3 = \left([n(n+1)]/2\right)^2.$$

Step 1:

With the help of principle of induction in maths, let us check the validity of the given statement P(n) for n=1

$$P(1) = [(1(1+1)]/2)^2 = (2/2)^2 = 1^2 = 1$$

This is true.

Step 2:

Now as the given statement is true for n=1, we shall move fwd for proving n=k (ie)

$$1^3+2^3+3^3+\dots+k^3 = ((k(k+1))/2)^2$$

Step 3:-

Try to establish that P(k+1) is also true

$$1^3+2^3+3^3+\dots+k^3+(k+1)^3 = \left[\frac{k(k+1)}{2}\right]^2 + (k+1)(k+1)^2$$

$$\Rightarrow (k+1)^2\left[k^2/4 + (k+1)\right]$$

$$= (k+1)^2\left(\frac{k^2+4k+4}{4}\right)$$

$$= (k+1)^2\left(\frac{k+2}{2}\right)^2$$

$$= \left[\frac{(k+1)(k+2)}{2}\right]^2$$

2

Worksheet-1.

(2) → Prove that $1+3+5+\dots+(2n-1)=n^2$

Let $P(n): 1+3+5+\dots+(2n-1)=n^2$ is the given statement

Step 1:

put $n=1$

Then LHS $=1$

RHS $=(1)^2=1$

$\therefore$ LHS = RHS

$\Rightarrow P(n)$ is true for $n=1$

Step 2

Assume that $P(n)$ is true for $n=k$.

$\therefore 1+3+5+\dots+(2k-1)=k^2$

adding $2k+1$ on both sides, we get

$1+3+5+\dots+(2k-1)+(2k+1)=k^2+(2k+1)=(k+1)^2$

$\therefore 1+3+5+\dots+(2k-1)+(2(k+1)-1)=(k+1)^2$

$\Rightarrow P(n)$ is true for $n=k+1$

$\therefore$ by the principle of mathematical induction

$P(n)$ is true for all natural numbers 'n'.

hence, $1+3+5+\dots+(2n-1)=n^2$, for all $n \in \mathbb{N}$.

3) The deductive proof consists of sequence of statements given with logical reasoning in order to prove the first or initial statement. The initial statement is called hypothesis.

The inductive proof is a recursive kind of proof which consists of sequence of parameterised statements that use the statement itself with lower values of its parameter.

④ RE for regular language L which accepts all the strings which begin or end with either 00 or 11.

Ans The regular expression can be categorised into two subparts.

$R = L_1 + L_2$

L_1 = The string which begin with 00 or 11.

L_2 = The string which end with 00 or 11.

$L_1 = (00+11)$ (any no of 0's + 1's)

$L_1 = (00+11).(0+1)^*$

L_2 = (any number of 0's + 1's).$(00+11) = (0+1)^*(00+11)$

$R = [(00+11)(0+1)^*] + [(0+1)^*(00+11)]$.

5) RE which accepts all the strings with at least 2b's over $\Sigma = \{a, b\}$.

$R = (a+b)^* b (a+b)^* b (a+b)^*$.

6) RE to denote the language L over $\Sigma = \{a, b\}$ such that all the strings do not contain substring "ab"

$L = \{\varepsilon, a, b, bb, aa, ba, baa, \ldots\}$

$re = (b^* a^*)$

⑦ 3rd character from the right end of the string is always "a"

$r = (a+b)^* a (a+b)(a+b)$.

Worksheet 2

1) Regular Language for the regular expr

i) $R = \{a^* b^* c\}$.

L = ~~denotes the~~ {Language L over Σ^*, where $\Sigma = \{a, b, c\}$ in which every string will be such that any number of a's is followed by any number of b's is followed by c.

ii) $R = \{1(0+1)^* 0\}$

L = ~~denotes the~~ {Language accepting the strings which are starting with 1 and ending with 0, over the set $\Sigma = \{0, 1\}$.

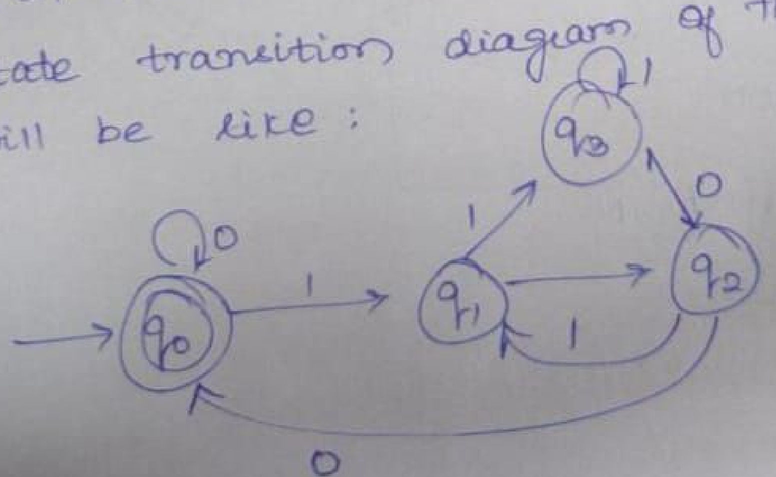
2) Language for RE = $R = \{(b^*(aaa)^* b^*)^*\}$.

L = [The language consists of the strings in which a's appear tippled, there is no restriction on number of b's].

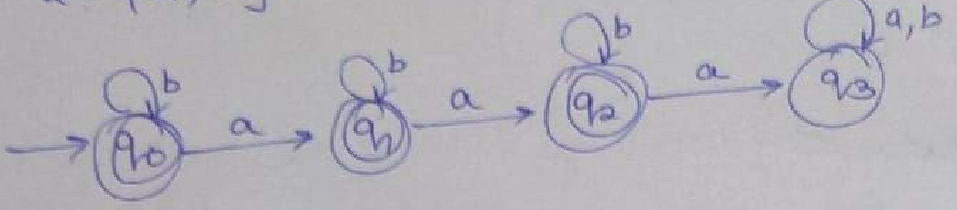
3) Consider the following inputs,

{0, 01, 10, 11, 100, 101, 110, ... }

State transition diagram of the language will be like:



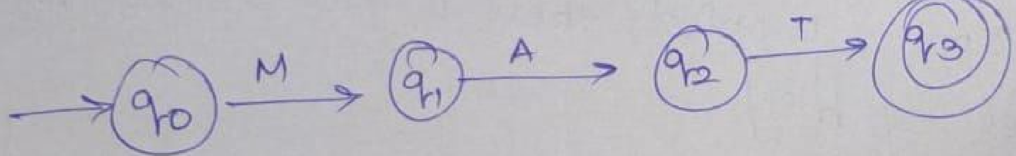
(5) DFA → having more than two a's over $\Sigma = \{a, b\}$.



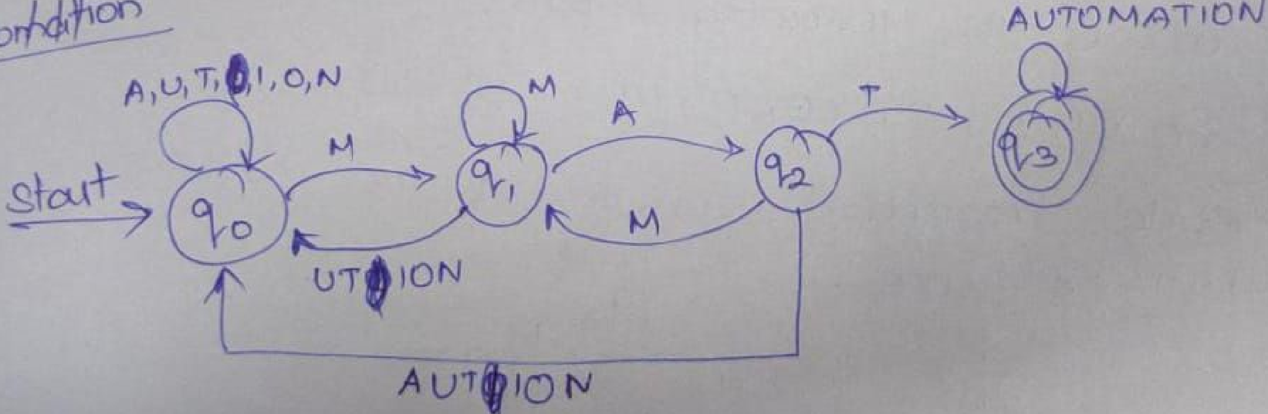
Transition table

state \ i/p	a	b
q_0	q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_2
q_3	q_3	q_3

(4) Step 1



Automation



WORK SHEET- 3

1) Construct a FA for even no of 0's and odd number of 1's. Given the regular expression for even no of 0's or odd number of 1's

⇒ (i) FA for even no of 0's or odd no. of 1's.

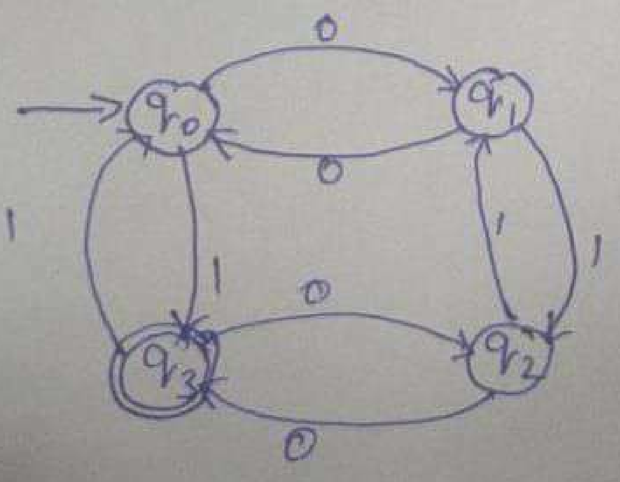
Consider the assumption of each state as:

q_0: State of even number of 0's and even number of 1's

q_1: State of odd number of 0's and even number of 1's

q_2: State of odd number of 0's and odd number of 1's

q_3: State of even number of 0's and odd number of 1's



(ii) Regular expression for even no of 1's or odd number of 1's

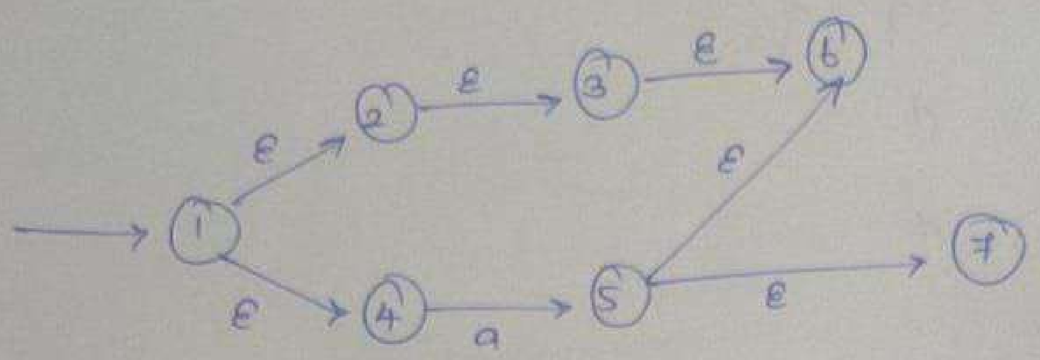
soln

$$(1(11)^* + (00)^* 1)$$

(or)

$$((00)^* 1 + (1(11)^*))$$

2)



ε-closure $(1) \rightarrow \{1, 2, 4, 3, 6\}$

ε-closure $(2) = \{2, 3, 6\}$

ε-closure $(3) = \{3, 6\}$

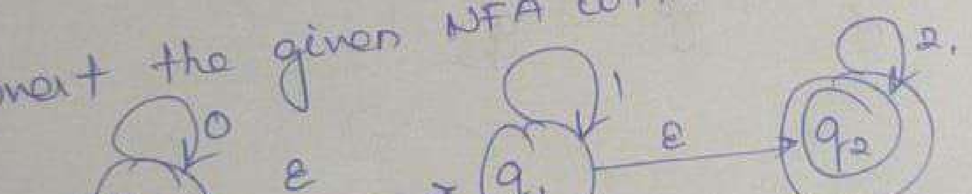
ε-closure $(4) = \{4\}$

ε-closure $(5) = \{5, 6, 7\}$

ε-closure $(6) = \{6\}$

ε-closure $(7) = \{7\}$.

3) Convert the given NFA with ε to NFA without ε.



ε-closure $(q_0) = \{q_0, q_1, q_2\}$

ε-closure $(q_1) = \{q_1, q_2\}$

ε-closure $(q_2) = \{q_2\}$

$\delta'(q_0, 0) = \{q_0, q_1, q_2\}$

$\delta'(q_0, 1) = \{q_1, q_2\}$

$\delta'(q_1, 0) = \phi$

$\delta'(q_1, 1) = \{q_1, q_2\}$

$\delta'(q_2, 0) = \phi$

$\delta'(q_2, 1) = \phi$

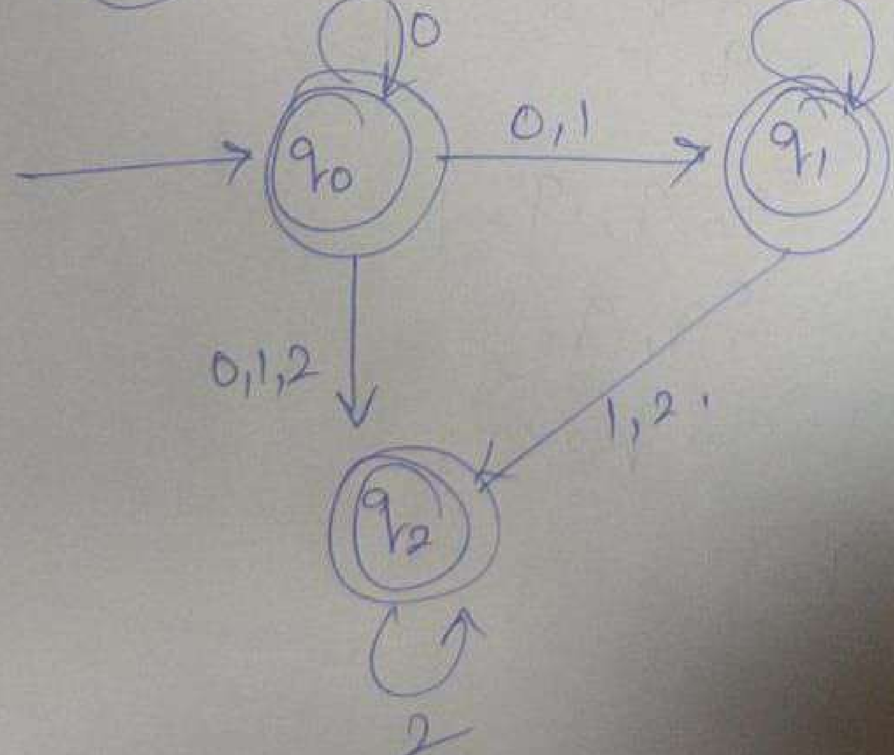
$\delta'(q_0, 2) = \{q_2\}$

$\delta'(q_1, 2) = \{q_2\}$

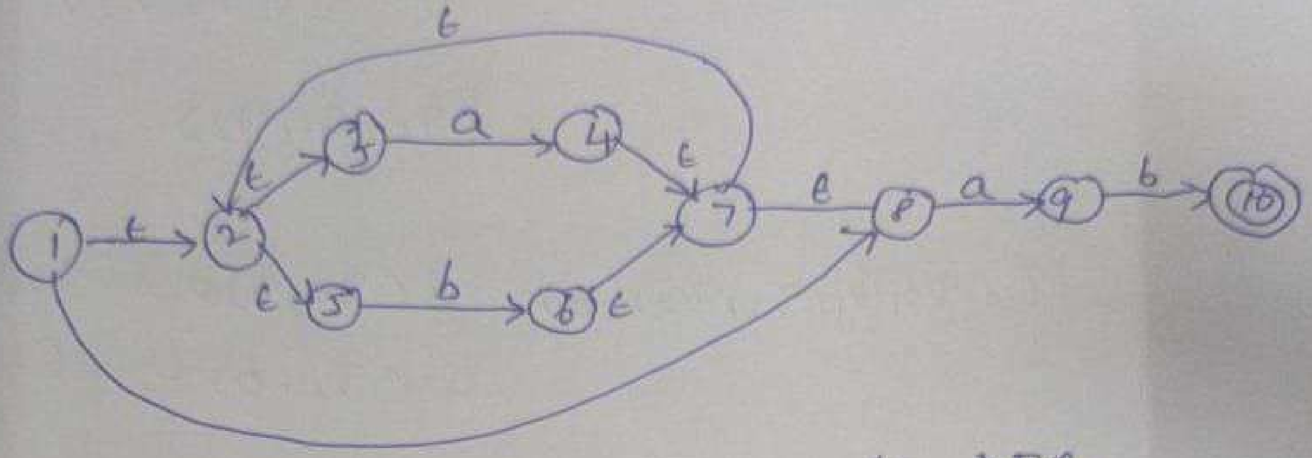
$\delta'(q_2, 2) = \{q_2\}$

transition table.

state \ i/p	0	1	2
q_0	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_2\}$
q_1	ϕ	q_1, q_2	q_2
q_2	ϕ	ϕ	q_2



4) NFA for $(a+b)^* ab$



5) convert the given NFA to DFA

state/inputs	0	1
→ q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\{q_1\}$
q_2	$\{q_3\}$	$\{q_3\}$
q_3	ϕ	$\{q_2\}$

$$\delta'((q_0, q_1), 0) = \delta(q_0, 0) \cup \delta(q_1, 0)$$
$$= \{q_0, q_1\} \cup \{q_2\}$$
$$= \{q_0, q_1, q_2\}$$

$$\delta'((q_0, q_1), 1) = \delta(q_0, 1) \cup \delta(q_1, 1)$$
$$= \{q_0, q_1\}$$

$$\delta'((q_0, q_1, q_2), 0) = \delta(q_0, 0) \cup \delta(q_1, 0) \cup \delta(q_2, 0)$$
$$= \{q_0, q_1, q_2, q_3\}$$

$$\delta'((q_0,q_1,q_2),1) = \delta(q_0,1) \cup \delta(q_1,1) \cup \delta(q_2,1))$$

$$= \{q_0, q_1, q_3\}$$

$$\delta'((q_0,q_1,q_2,q_3),0) = \delta(q_0,0) \cup \delta(q_1,0) \cup \delta(q_2,0) \cup \delta(q_3,0)$$

$$= \{q_0, q_1, q_2, q_3\}$$

$$\delta'((q_0,q_1,q_2,q_3),1) = \delta(q_0,1) \cup \delta(q_1,1) \cup \delta(q_2,1) \cup \delta(q_3,1)$$

$$= \{q_0, q_1, q_2, q_3\}$$

$$\delta'((q_0,q_1,q_3),0) = \delta(q_0,0) \cup \delta(q_1,0) \cup \delta(q_3,0)$$

$$= \{q_0, q_1, q_2\}$$

$$\delta'((q_0,q_1,q_3),1)) = \delta(q_0,1) \cup \delta(q_1,1) \cup \cancel{\delta(q_2,1)} \delta(q_3,1)$$

$$= \{q_0, q_1, q_2\}$$

we are ...

Transition table of DFA

state / input	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_1\}$
$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2, q_3\}$	$\{q_0, q_1, q_3\}$
$\{q_0, q_1, q_2, q_3\}$	$\{q_0, q_1, q_2, q_3\}$	$\{q_0, q_1, q_2, q_3\}$
$\{q_0, q_1, q_3\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$