|  |  |  |
|---|---|---|
| Subject Name | : | Theory of Computation |
| Subject code | : | XCS 352 |
| Class | : | M.Sc Information Technology |
| Semester | : | Fifth |

Prepared By

**R.I. HEAVEN ROSE**

Dept. of SE CT & IT

NICE

NOORUL ISLAM COLLEGE OF ENGINEERING
M.Sc Information Technology(5 years)
Fifth Semester
XCS 352 – Theory of Computing

**Part A questions and Answers**

1. Define DFA

   A Deterministic Finite Automata(DFA) is defined by five typles $(Q,\Sigma,q_o,F,\delta)$ where

   Q - finite, non empty set of states

   $\Sigma$ - an input alphabet

   $q_o \in Q$ - the start state

   $F \subseteq Q$ - a set of final states

   $\delta$ - a transition function

2. Define NFA

   The Non-deterministic Finite Automata (NFA) is defined by a five tuple $(Q,\Sigma,\delta,q_o,F)$ where

   Q - finite, non empty set of states

   $\Sigma$ - an input alphabet

   $q_o \in Q$ - the start state

   $F \subseteq Q$ - a set of final states

   $\delta$ - a transition function

3. Give the applications of Theory of computation.
   a. Compiler Design
   b. Robotics
   c. Artificial Intelligence
   d. Knowledge Engineering
4. Give any four applications of Finite Automata
   a. It is an useful tool in the design of Lexical analyzer
   b. Compiler design
   c. Pattern matching

    d. File Searching Program

5. Differentiate DFA with NFA

   A DFA is a deterministic Finite Automata where as NFA is a nondeterministic finite automata. In DFA, for a given state on a given input we reach to a deterministic and unique state. On the other hand, in NFA we may lead to more than one states for given input. The DFA is a subset of NFA.

6. Define star closure and positive closure. Star closure –zero or more. Occurrence positive closure – one or more occurance

   (eg) $a^* = \{\varepsilon, a, aa, \ldots.\}$

   $a+ = \{a, aa, \ldots.\}$

   $L^+ = L^* - \{\varepsilon\}$

7. Define NFA with ε-transition. Is the NFA's with ε-transitions are more powerful than the NFA's without ε-transition?

   The NFA with ε moves defined by 5tuple or quintyple as similarly as NFA, except δ $(Q,\Sigma,\delta,q_o,F)$ with all components as before and δ: $Qx(\Sigma U\{\varepsilon\}) = 2^Q$

   No, NFA with ε transition and NFA without ε-transition have the same power.

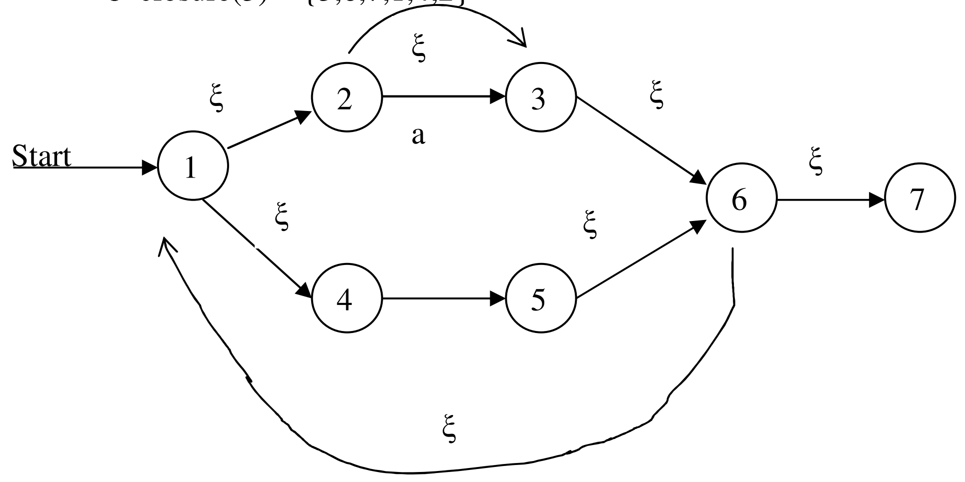
8. Define Finite State Systems

   The finite automaton is a mathematical model of a system, with disrete inputs and outputs and a finite number of memory configuration called states.

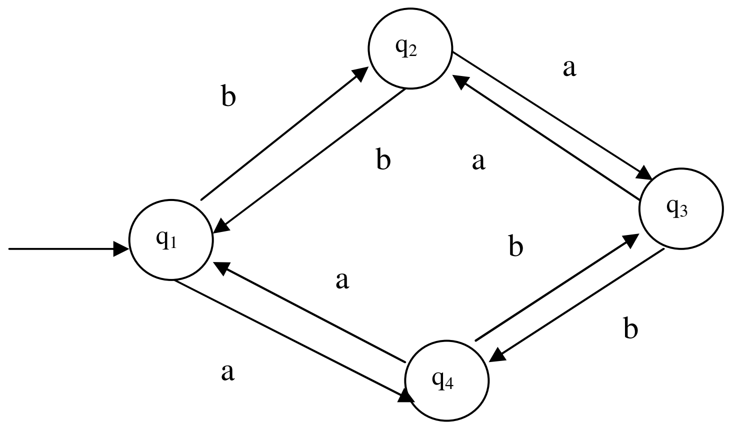
9. Find the ε-closure of the states 1, 3

   ε -closure(1) = {1,2,3,6,7,4}

   ε -closure(3) = {3,6,7,1,4,2}



10. Find the language accepted by the following automata



Some of the accepted strings are baba,baab,abab,…..

L={ω: ωЄ{a,b}*, strings contains even number of a's and even number of b's}

11. Describe the following sets by regular expressions.

(a) $L_1$=the set of all strings of o's and 1's ending in 00.

(b)$L_2$=the set of all strings of o's and 1's beginning with o and ending with1,

L1=(0+1)*00

L2=0(0+1)*1

12. Show that (r*)*=r* for a regular expressionr. (r*)*={ε,r,rr,….}=r*

13. Describe in the English language, the sets represented by the following r.e.

a(a+b)*ab

The set of all strings over {a,b} starting with 'a' and ending with'ab'.

14. Is it true the language accepted by NFA is different from the regular language? Justify your answer.

No, it is false. For every regular expression r there exists a NFA with ε-transition that accepts L(r).

15. From the given automata, M, Check whether the string ababba is accepted or not.

The string is not accepted by given automata.

16. Define Context-free Grammar.

A CFG is a way of describing languages by recursive rules (or) substitution rules called productions.

A CFG is defined as G=(V,T,P,S) where

V-Set of Non-terminals

T-Set of terminals

P-Production rule

S-Start symbol

17. Give the context free grammar for generating the ;amgiage

$L=\{a^n b^{2n};n\geq 1\}$

$S\rightarrow Sbb$

$S\rightarrow abb\xi$

18. What is a ambiguous grammar?

A context free grammar G is said to be ambiguous if it has more than one parse tree

e.g $E\Rightarrow id+id*id$.

19. Find the language generated by a CFG. $G=(\{S\},\{0,1\},\{S\rightarrow 0|1|\varepsilon, S\rightarrow 0S0|1S1|S)$

| | |
|---|---|
| $S\Rightarrow 0S0$ | $S\Rightarrow 1S1$ |
| $\Rightarrow 01S10$ | $\Rightarrow 10S01$ |
| $\Rightarrow 0I0I0$ | $\Rightarrow 10001$ |

L= { ω;ω is a palindrome}

20. Consider G whose productions are $S\rightarrow aAS|a$, $A\rightarrow SbA|SS|ba$. Show that $S\Rightarrow aabbaa$

| | |
|---|---|
| $S\Rightarrow Aas$ | $s\rightarrow aAS$ |
| $\Rightarrow aSbAS$ | $A\rightarrow SbA$ |
| $\Rightarrow aabAS$ | $S\rightarrow a$ |
| $\Rightarrow aabbaS$ | $A\rightarrow ba$ |
| $\Rightarrow aabbaa$ | $S\rightarrow a$ |

21. Optimize the CFG given below by reducing the grammar.

$S\rightarrow A|0C1, A\rightarrow B|01|10, C\rightarrow \varepsilon|CD$. Unit productions and useless symbols are

$A\rightarrow B$

$C\rightarrow CD$

So we can remove this productions. Eliminate ε for C in S.

$S\rightarrow A/01$

Since $A \rightarrow 01/10$

$S \rightarrow 01/10/01$

So, $S\rightarrow 10/01$

22. Define Pushdown Automata

A. Pushdown Automata M is defined by $(Q, \sum, \Gamma, \delta, q_0, z_0, F)$ Where

Q is a finite set of states

$\sum$ is an alphabet called input alphabet

$\Gamma$ is an alphabet called stack alphabet

q0 €Q is the start state

Z0 in $\Gamma$ is the start symbol on stack

$F \subseteq Q$ is the set of final state

$\delta$ is the transition relation

(ie) $\delta$ is a subset of $(Q \times (\sum\cup\{e\} \times \Pi$

23. Explain the following transition of PDA.

$\delta (P,a, \beta) = (q,\gamma)$

$((P, a, \beta), (q, \gamma)$ €$\delta$

It means that we

a. Read 'a' from the tape
b. Pop the string $\beta$ from the stack
c. More from state p to state q.
d. Push string $\gamma$ on to stack

24. Define the Instantaneous Description of PDA.

A PDA goes from configuration to configuration when consuming input.

Let P be a PDA $P=(Q,\sum,\Gamma,\delta, q_0,z_0,F)$.

The instantaneous descriptions of pushdown automata is such that $(q_1,aw,bx,) \vdash (q_2,w,yx)$ is possible if and only if $(q_2,y)$€ $\delta(q,a,b)$

(ie) $(q_1,a_1,a_2,\ldots\ldots a_n, z_m,\ldots\ldots z_2z_1) \vdash (q_2,a_2,\ldots\ldots a_n,\beta,z_m,\ldots\ldots z_2)$ is possible if and only if $\delta(q_1,a_1,z_1)$ contains $(q_2,\beta)$.

25. What are the different types of language acceptances by PDA and define them,

Two types

1. Acceptance by final state
2. Acceptance by empty stack

1. Acceptance by final state.

   Let M=(Q,∑,Γ,δ, $q_0$ ,$z_0$,F) be a PDA. Then, the language accepted by M is the set of strings such that

   L(M)=(ω| ($q_0$,ω,$z_0$) ├*(p, ε,γ),PЄF,γЄΓ*)
2. Acceptance by empty stack

   Let N(M) be the language accepted by M is given by N(M)={ω| $q_0$,ω,$z_0$) ├*(p, ε, ε) fsor some PEQ}

26. Is it true that the language accepted by a PDA by empty stock or that of final states are different in language?

    No, because the languages accepted by PDA's by final state are exactly the languages accepted by PDA's by empty stack.
27. What is the additional feature of PDA has when compared with NFA? Is PDA superior over NFA? Is PDA superior over NFA in the sense of language acceptance? Justify your answer:

    Additional features of PDA:

    a) PDA is eventually a FA with a stack.
    b) PDA can accept context free languages.
    c) PDA can recognize the context free languages which are not regular.

    Yes, PDA is superior over NFA in the sense of languages.
28. A=({$s_0$,p,q}, {a,b}, {a,b,e},δ,$s_0$,e,{q}) where δ is δ($s_0$,a,e)={(p,a)},δ(p,a,a)={(p,aa)} δ(p,a,e)={(p,a)},δ(p,b,a)={(P,E),(Q,E)} process string aaabbb.

    (So, aaabbb,e) ├(P,aabbb,ae) ├(P,abbb,aae) ├(p,bbb,aaae) ├(p,bb,aae) ├(p,b,ae) ├(p,e,e)

    Since q is a final state and stack is also become empty, given string is accepted by PDA.
29. Is it true the nondeterministic PDA is more powerful than that of deterministic PDA? justify your answer.

Yes, $ww^R$ is accepted by NPDA but not any DPDA in the sense of language acceptance.

30. Regular Language:

A Language can be described by DFA or NFA.

Context free language cannot be described by DFA (or) NFA – since there is no memory.

Pushdown Automata has a memory, which can be used to count the number.

Pushdown Automata can accept context free languages. It is essentially an NFA with a stack.

31. Definition of Deterministic Pushdown Automata(DPDA)

A pushdown automata $\Gamma=(Q,\sum,\Gamma,\delta,q_0,z,F)$ is said to be deterministic if

1. $\delta(q,a,b)$contains almost one element.

2. If $\delta(q,\lambda,b)$ is not empty, then $\delta(q,c,b)$ must be empty for overy $c\in\sum$, $q\in Q$, $b\in\Gamma$

The first condition says that for any given input symbol and any stack top, at most one move can be made.

The second condition say that when a $\lambda$ move is possible for some configuration, no input consuming alternative is available.

32. Definition

A language L is said to be deterministic context free language, if and only if there exists a deterministic pushdown automata (dpda) M such that L = L (M)

33. Definition of Turing Machine

Turning Machine T(M) is defined as $M = (Q,\sum,\Gamma,\delta,q_0,B,F)$ where

Q is the finite set of states

$\Pi$ is the finite set of allowable tape symbols

B is a symbol of $\Gamma$, is the Blank

$\sum$ a subset of $\Gamma$ not including B, is the set of input symbols

$\delta$ is the next more function, a mapping from

$\delta\ Q \times \Gamma \rightarrow Q \times \Gamma \times \{L,R\}$

$Q_0 \in Q$ is the initial state

$F \subseteq Q$ is the set of final states

34. What are the actions takes place in Turning Machine

In one move the turning machine, depending upon the symbol scanned by the tape head and the state of finite control,

1. Changes state
2. Prints a symbol on the tape cell scanned, replacing what was written there

Moves its head left (or) right one cell.

35. Representation of turning machines

We can describe Turing machine using

a. Instantaneous Descriptions
b. Transition table
c. Transition diagram

36. Describe Instantaneous descriptions and more of Turing machine

An Instantaneous Description (ID) of a Turing Machine (M) is denoted by $\alpha q \alpha_2$.

Here q€Q is the current state of M. $\alpha_1$, $\alpha_2$ is the string in $\Pi^*$ that is the contents of the tape upto the right most nonblank symbol (or) the symbol to the left of the head whichever is rightmost.

Move of a Turing Machine

We define a more of M as follows

Let $x_1x_2 \ldots\ldots x_{i-1}\, qx_i \ldots\ldots\ldots x_n$ be an ID

Let i-1 = n, then take $x_i$ as B

If i=1, the there is no next ID

If i>1, then we write

$x_1\, x_2 \ldots\ldots x_{i-1} q x_i \ldots\ldots x_n \vdash x_1 x_2 \ldots\ldots x_{i-2}\, p\, x_{i-1} y x_{i+1} \ldots\ldots x_n$

If $\delta\ (q,x_i) = (p,y,R)$, then change of ID is $x_1,x_2 \ldots\ldots x_{i-1} q x_i \ldots\ldots x_n \vdash x_1 x_2 \ldots x_{i-1} y p x_{i+1} \ldots\ldots x_n$

37. Explain the Basic Turning machine model

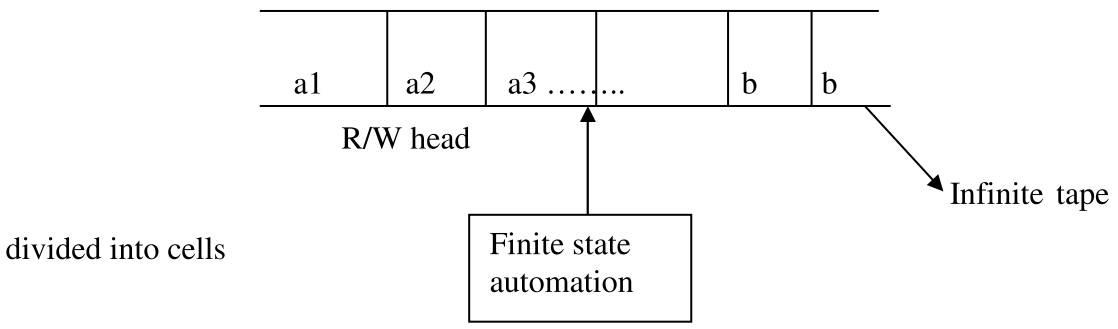


Fig. Turning machine model

The Turing machine can be thought of as a finite state automation connected to a R/W (Read/Write) head. It has an infinite tape which is divided into number of cells Each cell stores one symbol.

38. Definition

The language accepted by M denoted by L(M), is L(M) = {ω:ω€* and $q_o\omega \vdash^* \alpha_1 p \alpha_2$, for some p€F, $\alpha_1$, $\alpha_2$ € Γ*}

When a Turing machine processes on input string x, there are three possibilities.

a. Turing machine can accept the string by entering the accepting state
b. It can reject the string by entering non-accepting state
c. It can enter an infinite loop so that it never halts.

39. Explain the following more transition in TM,

$\delta(q_0,a) = (q_1,d,R)$

δ $(q_0,a)$ denotes the current state of control unit $q_0$ and the current tape symbol being processed or read is 'a'

$(q_1, d,R)$ denotes the result of new state $q_1$, in control unit, a new tape symbol which replace the old one, and a move symbol to the right.

40. Is it possible that a Turing machine could be considered as a computer of functions from intergers to integer? Is yes justify your answer.

Yes, the turning machine may be viewed as a computer of functions from positive intergers to positive integers. Each integer will be represented in an unary fashion. That is , the integer i≥0 is represented by the sting $O^i$ If a function has R arguments

$i_1, i_2 \ldots\ldots. i_k$ then these integers are initially placed on the tape separated by 1 has $O^{i1}1O^{i2}1O^{i3}\ldots\ldots\ldots1O^{ik}$

41. Explain Two-way infinite tape

A turning machine with two way infinite tape is denoted by M = (Q, ∑, Γ, δ, qo, B, F) as in the original model. The tape is infinite to the left as well as to the right.

42. Define Multitape Turing Machine

A multitape TM consists of a finite content with k tapes heads and k tapes, each tape is infinite in both directions. On a single move, depending on the state of the finite control and the symbol scanned by each of the tape heads. The machine can

a. Change state
b. Print a new symbol on each of the cells scanned by its tape head.
c. Move each of its tape heads, independently, one cell to the right or left or keep it stationary.

43. Is Tm is more powerful than FMs.

Yes, Turing machine is more powerful than finite state machine because it has the capability to remember arbitarary relay long sequences of input symbol.

44. When a language is said to be recursive or recursively enumerable?

A language L is Recursive (R) if and only if there is a TM that decides L. L is Recursively enumerable (RE) if and only if there is a TM that semi decides L.

45. When a problem is said to be undesirable?

Curve an example of a decidable problem.

A problem is undesirable if there is no algorithm that takes as input an instance of the problem and determines whether the answer to that instance, is yes or no.

Example for decidable problem

Curve two DFSM $M_1$ & $M_2$, L ($M_1$) = L ($M_2$)

46. Show that union of recursive language is recursive.

Refer the chapter 9-3, Theorem 3.

47. When a problem is said to be decidable and give an example of an undecidable problem?

A problem whose language is recursive is said to be decidable.

Example for undesirable

a. Halting problem
b. Le = {e(M); M is a TM and L (M) = Q}

48. Definite of Recursive Languages

Let M = (Q, ∑, δ, S,H) be a TM such that

H = {y,n}, where y means yes and n means no

$\sum_o \subseteq \sum$ - {B} is the input alphabet

L ⊆ ∑* is a language

(s,w) is the initial configuration

M decides L if, for all string $\omega \in \sum_0^*$

– either ω∈L, in which case M accepts ω (M halts in state y)

--or ω∈L, then M rejects ω (M halt in state n)

49. Definition of Recursively Enumerable languages

Let M = (Q,∑, δ, S,H) be a TM

H = {y,n}

$\sum_0 \subseteq \sum$ - {B}

$L \subseteq \sum_0$

We say M semidecides L if, for all strings $\omega \in \sum_o$

– either ω∈L, in which M halts on Input ω

–or ω∈L, in which case M does not halt

(of course it may stop or loop)

**Theorems**

50. The union of two recursive languages is recursive
51. The union of two recursively enumerable language is RE.
52. If a language L and its complement I are both recursively enumerable, the L is recursive
53. Is the complements of an RE language always RE? No
54. If L is recursive, then L is RE

55. If L is Re, then is L always recursive? No

56. Explain the modified Post's correspondents problem.

Given lists A and B of k strings each from $\sum^*$, A = $w_1, w_2$ …….wk and B = $x_1, x_2$………$x_k$ does there exist a sequence of integers $i_1, i_2$………$i_k$ such that $w_1$, $w_{i1}, w_{i2}$,……………$w_{ir}$ = x, $x_i$……..$x_{ir}$

The difference between the MPLP and PCP is that in the MPCP a solution is required to start with the first string on each list.

57. When a recursively enumerable language is said to be recursive. Is it true that the language accepted by a nondeterministic Turing machine is different from recursively enumerable language?

A language L is recursively enumerable if there is a TM that accepts L, and recursive if there is a TM that recognizes L.

No, the language accepted by non determinist Turing machine is same as secursievley enumerable language.

58. Give tow properties of recursively enumerable sets which are undesirable.

   a. Emptiness
   b. Finiteness
   c. Regularity
   d. Context freedom

59. What do you mean by universal Turing machine

Universal Turing machine is TM, that can be programmed to solved any problem that can be solved by any TM.

60. What is universal languages

The universal Lu (of the UTM u) consist of the set of binary strings in the form of pairs (M, ω), where M is a TM encoded in binary and ω is its binary input string, such that w€ L (M) ie Lu = L (u) where (M, ω) € L (u).

61. An FSM (Finite State Machine) can be considered to be a TM (Turning machine) of finite tapa length, without rewinding capability and unidirectional tape movement.

62. Define NP-Complete Problem

The language L is said to be NP-Complete if $L \subseteq$ NP and L is NP-hard.

63. NP- hard problem:

A language L is said to be NP-hard if $L1 \subseteq PL$ fsor every L, $\subseteq$ NP.

64. Example for NP-complete problems

    The directed Hamilton circuit problem.

    The Vertex cover problem.

65. Example for NP-hard problem

    Integer linear programming

66. Abstract machines

    The computer which performs computation are not actual computers they are abstract machine

67. Examples of abstract machine

    Turing machine, Finite automata.

68. Limitation of Finite automata.

    It can recognize only simple languages

69. Transition deagram of DFA.

    Transition diagram associated with DFA is a directed graph whose vertices corresponds to states of DFA. The edges are the transitions from one state to another.

70. The language accepted by finite automata's are called regular language.

71. Equivalence of DFA and NFA

    As every DFA is an NFA, the class of languages accepted by NFA's includes the class of languages accepted by DFA's.

    DFA can simulate NFA.

    For every NFA, there exists an equivalent DFA.

72. Some identities for Regular expressions

    $\Phi+R=R$

    $\varphi R=R\varphi=\varphi$

    $\lambda R=R\lambda=R$

    $\lambda^*=\lambda$ and $\varphi^*=\lambda$

    $R^*R^*=R^*$

73. Leftmost derivation:

    A derivation $A\Rightarrow^*\omega$ is called a leftmost derivation if we apply a production only to the leftmost variable at every step.

74. Rightmost derivation.

A derivation $A \Rightarrow^* \omega$ is called a rightmost derivation if we apply a production only to the right most variable at every step.

75. Simplification of context free grammar.

Variables are eliminated if it does not derive any terminal string.

Elimination of null production.

Elimination of unit production.

NOORUL ISLAM COLLEGE OF ENGINEERING
M.Sc Information Technology(5 years)
Fifth Semester
XCS 352 – Theory of Computing
**Part B questions and Answers**

1. Let $M=(\{q_0, q_1\},\{0,1\},\{x,z_0\},A,q_0,z_0,\Phi)$.
   Where $\delta$ is given by
   $\delta (q_0,0,z_0)=\{(q_0,xz_0)\}$
   $\delta (q_0,0,x)=\{(q_0,xx)\}$
   $\delta (q_0,1,x)=\{ (q_1, \varepsilon )\}$
   $\delta (q_1,1,x)=\{ (q_{1,\varepsilon)\}}$
   $\delta (q_1, \varepsilon, x)=\{ (q_{1,\varepsilon)\}}$
   $\delta (q_1, \varepsilon, z_0)=\{(q_{1,\varepsilon)\}}$
   Construct the CFG equivalent to this PDA.

**Solution**

$\delta (q_0,0,z_0)=\{(q_0,xz_0)\}$
This is in the form of
$\delta (q_i, a, A)=\{(q_j, BC)\}$
The CFG for this transition is
$(q_i,A,q_k) \rightarrow a((q_j,B,q_l) \ (q_l,C,q_k)$
So CFG for our transition is
k =0
l =0 $(q_0,z_0,q_0) \rightarrow 0((q_0,x,q_0) \ (q_0,z_0,q_0)$
l =1$(q_0,z_0,q_0) \rightarrow 0((q_0,x,q_1) \ (q_1,z_0,q_0)$

k =1
l =0$(q_0,z_0,q_1) \rightarrow 0((q_0,x,q_0) \ (q_0,z_0,q_0)$
l=1$(q_0,z_0,q_1) \rightarrow 0((q_0,x,q_1) \ (q_1,z_0,q_0)$

$\delta (q_0,1,x)=\{ (q_1, \varepsilon )\}$
This is in the form of
$\delta (q_i, a, A)=\{(q_j, \varepsilon)\}$

The corresponding CFG is
$(q_i,A,q_j) \rightarrow a$
So CFG for our transition is
$(q_o,x,q_1) \rightarrow 1$

2. Construct a PDA accepting $L= \{a^n b^m c^n;\ m, n >=1\}$by empty store
   **Solution**
   The language consist of a set of a's followed by any number of b's and followed by same set of c's.
   Steps:
   1. store all a's in the string until symbol 'b' is read in input string.
   2. on seeing 'b' in the i/p string, there is a change in state and there is no change in stack.
   3. once all the b's is in the i/p string is exhausted, the remaining c's in the i/p string in matching with the a's present in the stack.

3. Construct a PDA equivalent to the following grammar
   $S \rightarrow aAA$ , $A \rightarrow aS / bS / a$
   **Solution**
   To solve this problem you should remember the following points
      1. Insert a start symbol on the stack.
      2. Read $\lambda$ with the non terminal.
      3. Read input symbol
      4. Both input and top of stack are same.
      5. Read $\lambda$ with empty stack.(i.e) enter into final state

4. Show that the language $L=\{a^n b^{n+1} c^{n+2} ; n >=1\}$ is not CFL
   **Solution**
   Given $L=\{a^n b^{n+1} c^{n+2\}}$
   Let $Z = a^m b^{m+1} c^{m+2} \in L$
   By pumping lemma, Z can be written as
   uvwxy such that
   $u=a^m$
   $vwx=b^{m+1}$ where $| vwx| <=n$
   $vx = b^1$ where $|vx|>=1$
   $y = c^{m+2}$
   then $uv^i wx^i y = uvwx(vx)^{i-1} y$
   $=a^m b^{m+1} b^{1(i-1)} c^{m+2}$
   $= a^m b^{m+1+i-1} c^{m+2}$
   $= a^m b^{m+1} c^{m+2} \in L$ for all values of i
   Hence language L is not CFL.
5. Design a Turing machine M to recognize the language $L=\{ 1^n 2^n 3^{n\ ;\ n>=1}\}$.
   Solution

   Let us evolve the procedure for processing of input string 112233.

Step1: q1 is the initial state. The R/W head scans the leftmost replaces 1 by b, and moves to the right . M enters q2
Step 2 : On scanning the leftmost 2, the /w head scan replaces 2 by b, and moves to the right, M enters q3.
Step 3 : On scanning leftmost 3, the r/w header replaces 3 by b, and moves to the right m enters q4.
Step4: After scanning the rightmost 3 the r/w head moves to the left until it finds leftmost 1. As a result , the leftmost 1,2, and 3 replaced by 3.
Step5 :Steps 1 – 4 are repeated until all 1's , 2's, and 3's are replaced by blanks.

6. Explain Universal Turing Machine

   Universal TM's that can be programmed to solve any problem that can be solved by any turing machine
   Input to U : The encoding "M" of a TM M and the encoding "w" of the string w є *
   Behavior : U halts on input "M","w" if and only if M halts on input w.
   Encoding TM's and their input strings:
   **Generic move**
1. We call symols 0,1,B by X!,X2,X3 respectively.
2. The directions L and R are called by synonyms D1 , D2 respectively.
3. The generic move
   $\delta$ ($q_i$, Xj)=($q_k$,$X_l$,$D_m$) is encoded by the binary string form
   $0^i\ 1\ 0^j\ 1\ 0^k\ 1\ 0^l\ 1\ 0^m$
   The binary code of Turing machine M is
   111 code1 11 code2 11……….11 coden 111

7. Explain Halting Problem
   Theorem:
   The halting problem is un decidable that is
   H={"M","w" : TM M halts in string w} is not recursive.
   Proof Idea:
   - Assume that H is recursive
   - By the definition of recursive languages, there is a TM $M_H$ that decides H
   - Design a new TM that uses $M_H$ as a submachine to decide $H_1$
   - But this construction contradicts that non recursiveness of $H_1$. Hence , H is not recursive
   - We say $H_1$ is reduced to H.
8. The Language L is defined as {M1 ᴜ M2 :DFSMs M1, M2 and L1 ⊊ L2} is recursive
   Proof:
   The following TM ME decides L: On input "M1" and "M2" where m1 and M2 are DFSMs
   - Cnstruct DFSM M for the language L(M1∩ L(M2).
   - Apply M φ to the input string "M"
   - If Mφ halts in y, M1 halts in y.

- If Mφ halts in n, M1 halts in n.

9. Construct a TM for successive function?

   F: N → N, f(x) = x+1

   Solution :

   Assume that the input is encoded in UNARY form.

   Let M=(Q, E,I Q0,B,F)

   Where

   Q={q0,q1}q0 =start state q1 = final state

   = {0}

   = {0.B}

   F={q1}

   The transition function can be defined as

| State | tape symbol | |
|---|---|---|
| | 0 | B |
| q0 | (q0,0,R) | (q1,0,R) |
| q1 | - | - |

   Let us take the input x=3, This is encoded as 000

   (q0,$\underline{0}$00B) = (q0,0$\underline{0}$0B) = (q0,00$\underline{0}$B)= q0,000$\underline{B}$) =(q1,000$\underline{0}$B)

   The Machine halts an accepting state q1 computing the success of x.

10. Consider G whose productions are S →aAS /a, A→ SbA/SS/ba

    For the string w=aabbaa find left most derivation, rightmost derivation

    Solution

    Leftmost derivation

    S → aAS
    →aSbAS
    →aabAs
    →aabbaS
    →aabbaa

    Right most derivation

    S →aAS
    →aAa
    →aSbAa
    →aSbbaa
    →aabbaa