M.Sc(CS) ,Third Semester Examination,2014-15
Subject: Theory of computation
M.Sc(CS)-304

Time: Three Hours] **[Maximum Marks : 60**

Note: Question Number 1 is compulsory. Answer any four questions from the remaining.

Q1. (Give answer in short) **Marks : 10X2**

---

i. What do you mean by left recursion?

If a grammar contains a pair of production of the form $A \rightarrow A\alpha \mid \beta$, then the grammar is a left recursive grammar. Left Recusrion can be eliminated from the grammar by replacing $A \rightarrow A\alpha \mid \beta$ with the production $A \rightarrow \beta B$ and $B \rightarrow \alpha B$ | null where A,B are variables, $\beta,\alpha \in (Vn \cup \sum)^*$

----

ii. Describe different type of automaton.

Deterministic finite automata (DFA)

Each state of an automaton of this kind has a transition for every symbol in the alphabet. Deterministic Finite Automata can be defined as $M=(Q,\sum,\delta,q0,F)$ where Q is the set of states

$\sum$ is the input symbols

δ is the transition function $Q \times \sum Q$

q0 is the start state

F is the final state

Non deterministic finite automata (DFA)

Non Deterministic Finite Automata can be defined as $M=(Q,\sum,\delta,q0,F)$ where

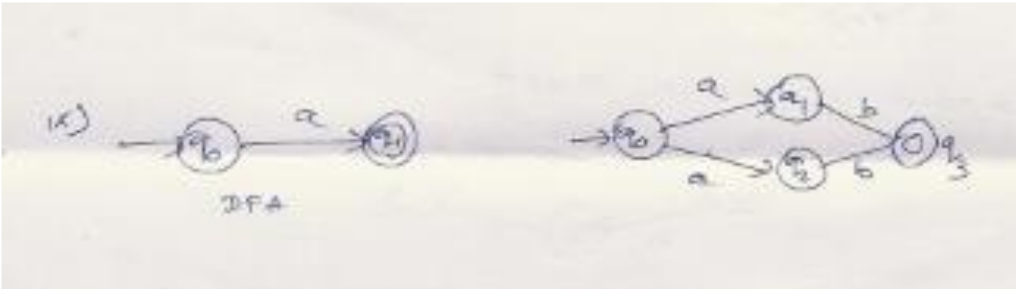
Q is the set of states

$\sum$ is the input symbols

δ is the transition function $Q \times \sum \rightarrow 2^Q$

q0 is the start state

F is the final state



----

iii. Explain left context and right context in a production.

In a production of the form φAψ→φαψ φ is called the left context of A and ψ is called right context of A. Example aAb→aBb left context of A =a Right context of A =b

----

iv. What do you mean by linear bounded automaton?

Linear bounded automation is described by the format
M=(Q,∑,Γ,δ,q0,b,C,$,F)

The input alphabet ∑ contains two special symbol C and $. C is called the left end marker and $ is called right end marker .The R/W never moves beyond the end mark.

Where Q= finite set of states

∑ = Input alphabet

Γ = is an alphabet called the stack

Q0= is the initial state q0 € Q

B is the blank symbol

F is set of final states F subset / equal to Q

----

v. What is the difference between Turing machine and universal turing machine?

Turing machines are designed to solve only one type of problem but Universal turing machines are designed to solve more than one problem. The inputs are different for both machines. Turing machine uses single tape but Universal turing machine uses multiple tapes.

----

vi. What is the difference between top down and bottom up parsing?

In top down parsing parsing takes place from the root node to terminal nodes . The derivation tree is traversed from the start of the grammar symbol to the terminal nodes. Ex: S→ AB

A→ a

B→ b

For string ab

S→AB

S→aB(A→a)

S→ab(B→b)

In bottom up parsing parsing takes place from the terminal nodes to the root node. In bottom-up parsing the derivation tree is traversed from the given input string to the start of the grammar symbol. Ex: S→AB

A→ a

B→b

For string ab

=aB (Using transition rule B→b)

=AB (Using transition rule A→a)

=S (Using transition rule S→AB)

----

vii. Define type 1 grammar with example.

If a grammar contains all its production as type 1 productions , the grammar is called type 1 grammar. The production S-> null is also allowed in a type 1 grammar but in this case S does not appear on tht right hand side of any production.

Type 1 production : A production of the form φAψ->φαψ is called a type 1 production if α not eqal to null

2A->1B

B->0

----

viii. Design a turing machine to compute 1's complement.

δ (q0,B)→(q0,B,R)
δ (q0,0)→(q1,1,R)
δ (q1,0)→(q1,1,R)
δ (q1,1)→(q1,0,R)
δ (q1,B)→q_f

---

ix. Define a Grammar

A grammar is a 4 tuple representation (V_N, ∑,P,S), where
V_N is a finite nonempty set whose elements are called variables.
∑ is a finite nonempty set whose elements are called terminals.
V_n U ∑ = **ϕ**
S is a special variable called the start symbol
P is a finite set whose elements are α→β where α and β arestring V_N U ∑

---

x. What do you mean by Chomsky Normal Form. Give examples.

A grammar whose productions are of the form
A→BC or A→a
Where A,B,C are non terminal and a is a terminal.
Example: S→AB
A→a
B→b

Marks: 4X10

Q-2 (a) Write a brief note on Turing Machine . 5

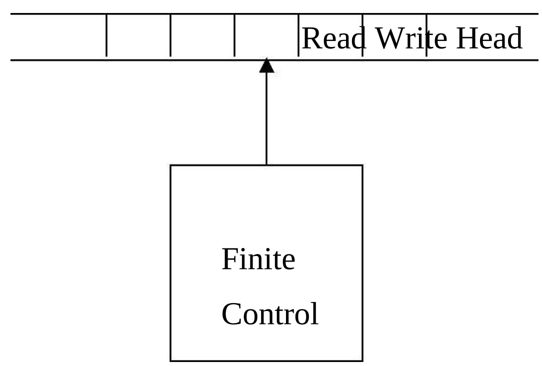
Basic model of a Turing machine consists of

i) a two way infinite tape,

ii) a read/write head and

iii) a finite control.

Input Tape



At any time, action of a Turing machine depends on the current state and the input symbol and involves (i) change of state (ii) writing a symbol in the cell scanned (iii) head movement to the left or right and (iv) Turing machine halts or not halts. A Turing machine may utilize the tape cells beyond the input limits and 'Blank' cell plays a significant role in the working of a Turing machine. Turing machine halts in any situation for which a transition is not defined. Unlike the previously dealt automata, it is possible that a Turing machine may not halt. At any state a Turing machine can halt or not halt. ie, it ends in accepting state if it successfully halts(accept halt). Otherwise it halts in any non accepting state (reject halt).

A turing machine M is a 7-tuple namely $(Q, \sum, \Gamma, \delta, q0, b, F)$ Where

Q is a finite nonempty set of states

Γ is a finite nonempty set of tape symbol

$B \in \Gamma$ is the blank

$\sum$ is a nonempty set of input symbols and is a subset of Γ and b does not belongs to $\sum$

Δ is the transition function mapping (q,x) onto (q',y,D)

$Q0 \in Q$ is the initial state

F is a subset or equal to Q

Left move :

Suppose δ (q,xi) =(p,y,L)

Id before processing

X1,x2…………………….xi-1 q xi ………………..xn

After processing

X1,……………….xi-2 p xi-1 y xi+1……………xn

Right move:

Suppose δ (q,xi) =(p,y,R)

Id before processing

X1,x2…………………….xi-1 q xi ………………..xn

After processing

X1,……………….xi-2 xi-1 y p xi+1……………xn

transition Table/Diagram example

Design a Turing machine to recognize all strings consisting of even number of 1's.

Solution: (i) q1 is the initial state. M enters state q2 on scanning 1 and writes b.

1) If M is in state q2 and scans 1, it enters q1 and writes b.

q1 is the only accepting state.

So M accepts a string if it exhausts all input symbols and finally in state q1. Symbolically,

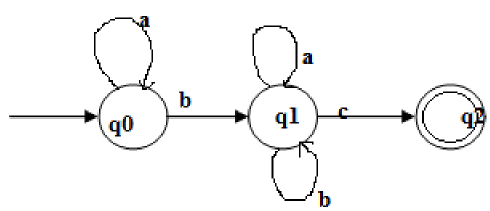
M=({q1,q2},{1},{1,b}, □ , q1,b,{q1})

Where δ is defined by

Present state	Input symbols	
	1	**B**
*q1	XRq2	BLq1
q2	XRq1	

(b) Write grammar for the regular expression. 5

a*b(a+b)*c



A0→aA0

A0→bA0

A1→aA1

A1→bA1

A1→cA2

A1→a

Q-3 (a) Explain types of Grammar with examples. 6

Types of grammar:

A type 0 grammar is any phase structure grammar without any restriction

A->a

A grammar is called type 1 or context dependent if all its production are type 1 productions. The production S-> null is also allowed in a type 1 grammar but in this case S does not appear on tht right hand side of any production.

Type 1 production : A production of the form φAψ->φαψ is called a type 1 production if α not eqal to null

2A->1B

B->0

A grammar is called a type 2 grammar if it contains only type 2 productions .It is also called a context free grammar A language generated by a context free grammar is called a type 2 language or a context free language

A type 2 production is a production of the form A->α where A € V_n and α $(V_n \cup \sum)^*$

Example S-> Aa, A->a

A grammar is called a type 3 grammar if it contains only type 3 productions . The production S-> null is also allowed in a type 1 grammar but in this case S does not appear on tht right hand side of any production.

A type 3 production is a production of the form A->a or A-> aB where A , B € V_n and a ϵ∑

Example B-> aC, A->a

(b) **Obtain the regular expressions for the following sets:**

(i)The set of all strings over {*a, b*} beginning and ending with '*a*'. **4**
(ii) $\{b^2, b^5, b^8, \ldots\}$

i) a(a+b)*a
ii) bb(bbb)*

Q-4 (a) Consider the CFG S→SaSaS | b . Find any three possible derivation for the grammar and draw parse tree for each. 6

i)S→SaSaS

S→baSaS(S→b)

S→babaS(S→b)

S→babab(S→b)

ii) $S \rightarrow SaSaS$

$S \rightarrow SaSaSaSaS(S \rightarrow SaSaS)$

$S \rightarrow baSaSaSaS(S \rightarrow b)$

$S \rightarrow babaSaSaS(S \rightarrow b)$

$S \rightarrow bababaSaS(S \rightarrow b)$

$S \rightarrow babababaS(S \rightarrow b)$

$S \rightarrow babababab(S \rightarrow b)$

iii) $S \rightarrow a$

(b) Show that the following languages are not regular

i) L= { $0^n 1 0^{2n}$ | n>=0}
ii) L= { $0^i 1^j$ | j is a multiple of i }

.

i) L={ null, 0100, 0010000,….}
Let w = 0100
x = 01
y=0
z=0
xy^2z =010000 does not ϵ L

ii) L={ 0011,001111,00111111……}
Let w=0011
x=00
y=1
z=1
xy^2z = 00111 which does not ϵ L

Q-5 (a) Construct a Moore machine equivalent to the Mealy machine M defined by the following table 5

Present state	Next State			
	a=0		a=1	
	state	output	state	output
->q1	q1	1	q2	0
q2	q4	1	q4	1
q3	q2	1	q3	1
q4	q3	0	q1	1

q2 is associated with outputs 0 and 1

q3 is associated with outputs 0 and 1

q2is converted into two state q20qnd q21

q3 is converted into two state q30 qnd q31

Present state	Next State			
	a=0		a=1	
	state	output	state	output
->q1	q1	1	q20	0
q20	q4	1	q4	1
q21	q4	1	q4	1
q30	q21	1	q31	1
q31	q21	1	q31	1
q4	q30	0	q1	1

Present state	Next State a=0	a=1	Output
->q1	q1	q20	1
q20	q4	q4	0
q21	q4	q4	1
q30	q21	q31	0
q31	q21	q31	1
q4	q30	q1	1

(b) What do you mean by context free grammar and parse tree? Explain with examples. 5

A CFG can be defined as G=(V,T,P,S) where V is the set of non terminals, T is the set of terminals, S is the start symbol and P is the set of productions of the form A->α where A belongs to V_N, α∈(VUT)*.

Derivation tree (Parse tree)

The derivation in a CFG can be represented by using trees called 'derivation tree' or 'parse tree'. A derivation tree for a CFG is a tree satisfying the following :

i) every vertex has a label which is a variable (non terminal) or terminal.
ii) The root has label which is non terminal
iii) The label of an internal vertex is a variable.

ie, a derivation tree is a labeled tree in which each internal node is labeled by a non terminal and leaves are labeled by terminals. Strings formed by labels of the leaves traversed from left to right is called the 'yield of the parse tree'. Ie, the yield of a derivation tree is the concatenation of the labels of the leaves without repetition in the left-to-right ordering.

Eg: Let G=({S,A},{a,b},P,S) where P is defined as

S-> aAS/a, A->b

S->aAS->aaASS->aabaa

Q-6 (a) Design a PDA to accept the language L= { wcw^R | w ϵ {a,b}*} 6

$\delta(q0,a,Z_0) = \{(q0,a\ Z_0)\}$

$\delta(q0,a,a) = \{(q0,a\ a)\}$

$\delta(q0,a,b) = \{(q0,a\ b)\}$

$\delta(q0,a,b) = \{(q,ab\ )\}$

$\delta(q0,c,a) = \{(q1,a)\}$

$\delta(q0,b,Z_0) = \{(q0,b\ Z_0)\}$

$\delta(q0,b,a) = \{(q0,ba)\}$

$\delta(q0,b,b) = \{(q0,bb)\}$

$\delta(q0,c,b) = \{(q1,b)\}$

$\delta(q0,c,Z_0) = \{(q1,\ Z_0)\}$

$\delta(q1,a,a) = \delta(q1,b,b)=\{(q1,null)\}$

$\delta(q1,null,Z_0) = \{(qf,\ Z_0)\}$

(b) Write a brief note on Universal Turing Machine. 4

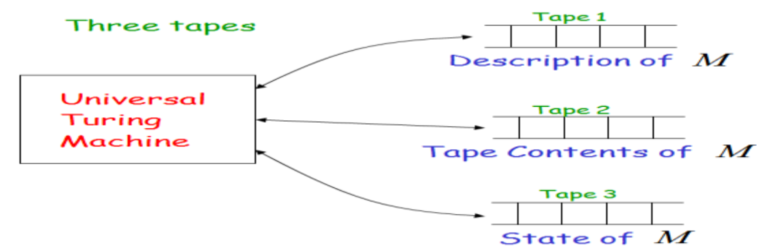
A limitation of Turing Machines:

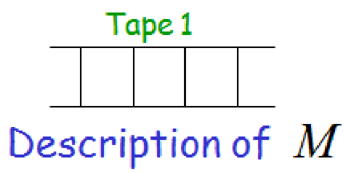
Turing Machines are "hardwired" they execute only one program but a universal turing machine can execute many types of program.

Attributes of universal turing machine:

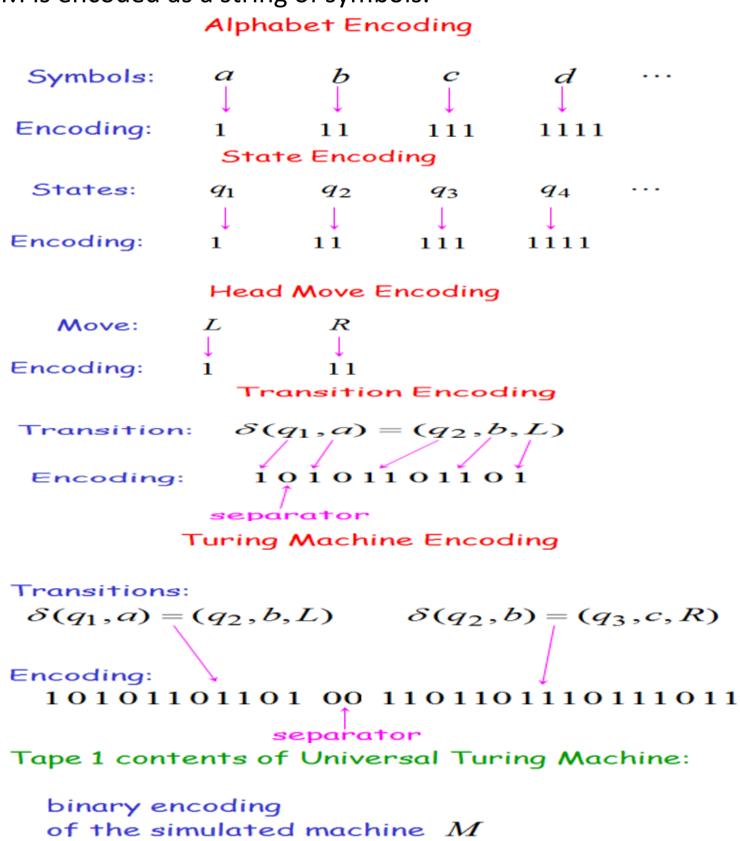
1. Reprogrammable machine
2. Simulates any other Turing Machine

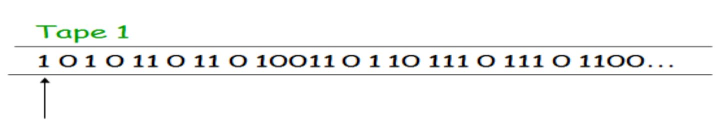
Universal Turing Machine simulates any Turing Machine M
Input of Universal Turing Machine:
Description of transitions of M
Input string of M





A Turing machine M can be described as a string of symbols:
M is encoded as a string of symbols.





Q-7 a) Find a reduced grammar equivalent to the grammar G whose productions are

S→AB | CA, B→BC | AB, A→a, C→aB | b 5

W1={ A,C}

As C→b and A→a

W2={A,C,S} S→CA

As CA appears on the right side of S

Step-II

S→CA

C→b and A→a

---

b) Proof the following identities for regular expressions by taking different strings

i) (PQ)*P=P(QP)*
. Let P=12 Q=34
L.H.S=(PQ)*P=$(1234)^2$12=1234123412
R.H.S=P(QP)*=12$(3412)^2$=1234123412

ii) (P+Q)R=PR + QR 5
Let P=12 Q=34 R=56
LHS = (P+Q)R=(12+34)56=1256
RHS = PR+QR=1256+3456=1256

Q-8 (a) Minimize the DFA given below. **5**

State	Inputs	
Q	0	1
->q0	q1	q2
q1	q4	q3
q2	q4	q3
*q3	q5	q6

*q4	q7	q6
q5	q3	q6
q6	q6	q6
q7	q4	q6

Divide the states in groups : Set of non final and final states

G1={q0,q1,q2,q5,q6,q7}   G2={q3,q4}

Apply input 0 to G1

δ (q0,0)=q1 ϵ G1

δ(q1,0)=q4 ϵ G2

δ (q2,0)=q4 ϵ G2

δ(q5,0)=q3 ϵ G2

δ (q6,0)=q6 ϵ G1

δ(q7,0)=q4   ϵ G2

Now G1={q0,q6} G2={q1,q2,q5,q7} G3={q3,q4}

Apply input 0 to G1

δ (q0,0)=q1ϵ G2

δ(q6,0)=q6 ϵ G1

G1=q0   G2=q6 G3={q1,q2,q5,q7} G4={q3,q4}

δ (q1,0)=q4 ϵ G4

δ(q2,0)=q4 ϵ G4

δ (q5,0)=q3 ϵ G4

δ(q7,0)=q4 ϵ G4

All belongs to same group for input a so apply input 1 to G3

δ (q1,1)=q3 ϵ G4

δ(q2,1)=q3 ϵ G4

δ (q5,1)=q6 ϵ G2

δ(q7,1)=q6 ϵ G2

G1=q0 G2=q6 G3={q1,q2} G4={q5,q7} G5={q3,q4}

δ (q1,0)=q4

δ(q2,0)=q4

δ (q1,1)=q3

δ(q2,1)=q3

All belongs to same group for input o and 1 . No further bifurcation

δ (q5,0)=q3

δ(q7,0)=q4

δ (q5,1)=q6

δ(q7,1)=q6

All belongs to same group for input o and 1 . No further bifurcation

δ (q3,0)=q5

δ(q4,0)=q7

δ (q3,1)=q6

δ(q4,1)=q6

All belongs to same group for input o and 1 . No further bifurcation

Thus minimum state automation is

M= { {q0},{q6},{q3,q4},{q5,q7},{q1,q2}}

(b) Design a Turing Machine that accepts the language L={ $a^n b^{3n}$ where n >0 } 5

δ (q0,B) =(q0,B,R)
δ (q0,a) =(q1,B,R)

δ (q1,a) =(q1,a,R)
δ (q1,b) =(q1,b,R)
δ (q1,B) =(q2,B,L)
δ (q2,b) =(q3,X,L)
δ (q3,b) =(q4,B,L)
δ (q4,b) =(q5,B,L)
δ (q5,b) =(q5,b,L)
δ (q5,a) =(q5,a,L)
δ (q5,B) =(q0,B,R)
δ (q0,X) =q_f
