question -1

Pushdown Automata

Since the DFA's or NFA's can not count & can not store the input for future reference, we have a new machine called pushdown Automata (PDA):

Defⁿ of PDA: A pushdown Automata (PDA) is a seven tuple

$$M=(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$$

Where

Q – is set of finite states
Σ – set of input alphabets
Γ – set of stack alphabets.
δ – transition from $Q\times(\Sigma\cup\varepsilon)\times\Gamma$ to finite subset of $Q\times\Gamma^*$
δ is called the transition fnc of M
$q_0\in Q$ is the start state of machine
$Z_0\in\Gamma$ is the initial symbol on the stack
$F\subseteq Q$ is set of final states

The actions (i.e transitions) performed by the PDA depends on

1. The current state
2. The next input symbol.
3. The symbol on top of the stack.

The actions performed by the m/c consists of

1. changing the states from one state to another
2. Replacing the symbol on the stack.

question-2

Languages of PDA :-

There are two cases wherein string w is accepted by a PDA.

* PDA accepts its input by consuming it and entering an accepting state, this approach is called "acceptance by final state"
* The set of strings that cause PDA to empty its stack, this approach is called "acceptance by empty stack"

K.S. Sampada
Lecturer, CSE
PNSIT

Acceptance by final state :-

Let $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be a PDA. Then $L(P)$, the language accepted by P by final state is

$L(P) = \{ w \mid (q_0, w, Z_0) \vdash^* (q, \varepsilon, \alpha) \}$

for some state q in F and any stack string α. That is, starting in the initial ID with w waiting on the input, P consumes w from the i.l.p and enters an accepting state. The contents of the stack at that time is irrelevant.

Acceptance by Empty stack

For each PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, then $N(P)$, the language accepted by P by empty stack is

$N(P) = \{ w \mid (q_0, w, Z_0) \vdash^* (q, \varepsilon, \varepsilon) \}$

for any state q, $N(P)$ is the set of inputs w that P can consume and at the same time its stack is empty. The set of accepting states is irrelevant.

∴ in this we can write P as a six-tuple $(Q, \Sigma, \Gamma, \delta, q_0, Z_0)$

quesion-3

(Eg) Obtain a PDA to accept the language $L=\{a^n b^n / n \geq 1\}$ by a final state.

Soln

$\delta(q_0, a, z_0) = (q_0, az_0)$

$\delta(q_0, a, a) = (q_0, aa)$

$\delta(q_0, b, a) = (q_1, \varepsilon)$

$\delta(q_1, b, a) = (q_1, \varepsilon)$

$\delta(q_1, \varepsilon, z_0) = (q_2, z_0)$

So the PDA to accept the language

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

$M = (Q, \Sigma, T, \delta_0, q_0, F)$

Where

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

$\Sigma = \{a, b\}$

$T = \{a, z_0\}$

δ: is shown below

$\delta(q_0, a, z_0) = (q_0, az_0)$

$\delta(q_0, a, a) = (q_0, aa)$

$\delta(q_0, b, a) = (q_1, \varepsilon)$

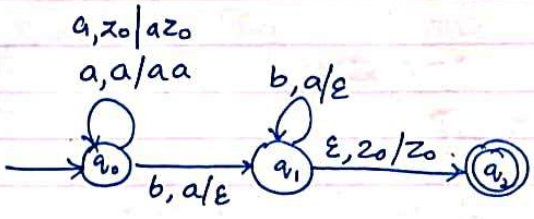
$\delta(q_1, b, a) = (q_1, \varepsilon)$

$\delta(q_1, \varepsilon, z_0) = (q_2, z_0)$

$q_0 \in Q$ is the start state of m/c

$z_0 \in T$ is the initial symbol on the stack

$F = \{q_2\}$ is the final state.



Eg obtain a PDA to accept the lang $L(M)=\{w \mid w \in$ and $n_a(w)=n_b(w)$ by a final state.

question-5

Sol?

δ:

$\delta(q_0, a, z_0) = (q_0, az_0)$

$\delta(q_0, b, z_0) = (q_0, bz_0)$

$\delta(q_0, a, a) = (q_0, aa)$

$\delta(q_0, b, b) = (q_0, bb)$

$\delta(q_0, a, b) = (q_0, \varepsilon)$

$\delta(q_0, b, a) = (q_0, \varepsilon)$

$\delta(q_0, \varepsilon, z_0) = (q_1, z_0)$

a
a
a
a
Sta

aaaba

So the PDA to accept the language

$L=\{w \mid n_a(w) = n_b(w)\}$

$M=(Q, \Sigma, T, \delta, q_0, z_0, f)$

$Q=\{q_0, q_1\}$

$\Sigma=\{a, b\}$

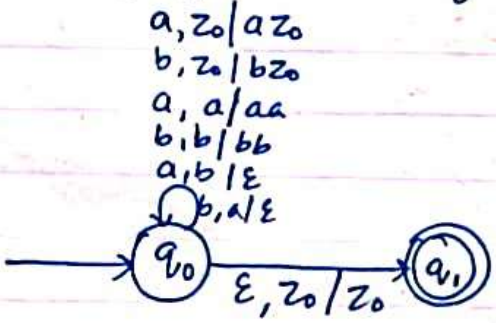
$T=\{a, b, z_0\}$

δ is shown above

$q_0 \in Q$ is the start state of m/c

$z_0 \in T$ is the initial symbol on the stack

$F=\{q_1\}$ is the final state

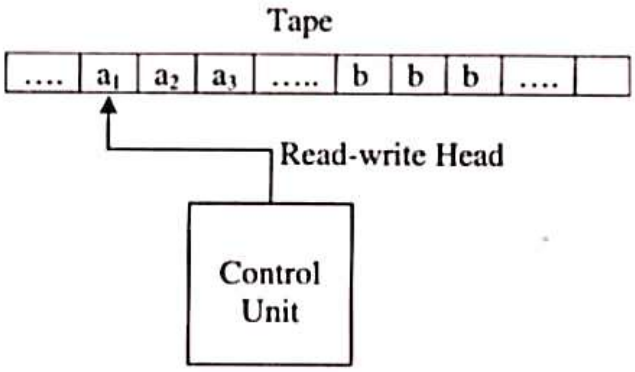


7.1. Turing Machine Model

question-7

The Turing machine model is shown in figure below. It is a finite automaton connected to read-write head with the following components:

- Tape
- Read-write head
- Control unit



Tape is used to store the information and is divided into cells. Each cell can store the information of only one symbol. The string to be scanned will be stored from the leftmost position on the tape. The string to be scanned should end with blanks. The tape is assumed to be infinite both on left side and right side of the string.

Read-write head: The read-write head can read a symbol from where it is pointing to and it can write into the tape to where it points to.

Control Unit: The reading from the tape or writing into the tape is determined by the control unit. The different moves performed by the machine depends on the current *scanned symbol* and the *current state*. The control unit consults *action table* i.e., *transition table* and carry out the tasks.

The read-write head can move either towards left or right i.e., movement can be on both the directions. The various actions performed by the machine are:

1. Change of state from one state to another state.
2. The symbol pointing to by the read-write head can be replaced by another symbol.
3. The read-write head may move either towards left or towards right.

If there is no entry in the table for the current combination of symbol and state, then the machine will halt. The Turing machines can be represented using various notations such as

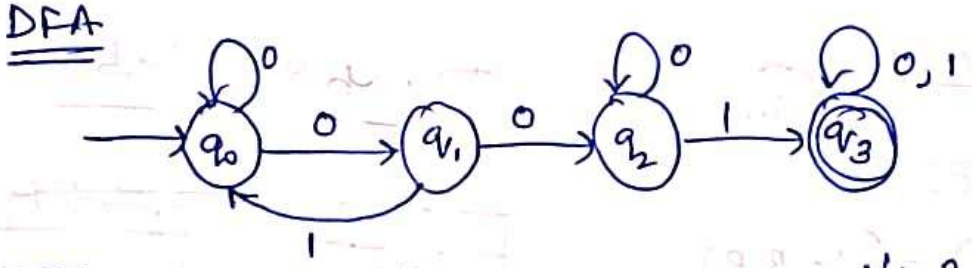
- Transition tables
- Instantaneous descriptions
- Transition diagram

question-12

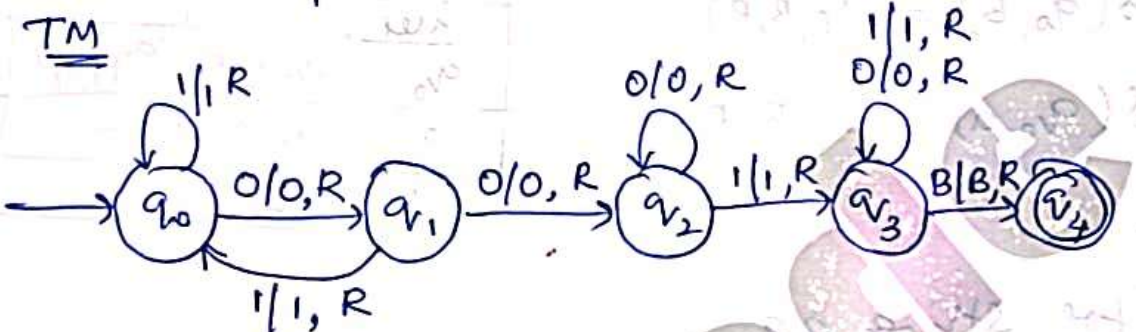
(Eg) Design a TM to accept the following language:
$L = \{w / w \in (0+1)^*\}$ containing the substring 001.

Soln:

DFA



TM



State	0	1	B
q_0	$q_1, 0, R$	$q_0, 1, R$	
q_1	$q_2, 0, R$	$q_0, 1, R$	
q_2	$q_2, 0, R$	$q_3, 1, R$	
q_3	$q_3, 0, R$	$q_3, 1, R$	q_4, B, R
q_4	–	–	–

Techniques for TM construction

The various techniques used for constructing a TM are

i) Turing machine with stationary head
ii) Storage in the state
iii) Multiple track turing Machine
iv) Subroutines.

i) Turing Machine with Stationary Head

In Standard turing we defined δ as

$$\delta(q, a) = (q', Y, D)$$

where D stands for direction. so D can be left or Right denoted by L or R. So the head moves to the left or right after reading an input symbol. If we want to include the option that head can continue to be in the same cell for some input symbol. Then we can define

$$\delta(q, a) = (q', Y, S)$$

This means that the TM on reading the input symbol a, changes the state to q' & writes Y in the current cell in place of a & continues to remain in the same cell.

The defn of TM with Stay-option is given by

$$M = (Q, \Sigma, I, \delta, q_0, B, F)$$

where

Q - Set of finite states

Σ - Set of input alphabet

I - Set of tape symbols

δ - transition fnc from $Q \times I$ to $Q \times I \times \{L, R, S\}$ indicating TM can move towards left or right or stay in the same position after updating the symbol on the tape.

q_0 - Start state

B - Special symbol indicating blank character

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

ii) Storage in the state:

In all the different types of machines we have studied such as: FSM or PDA or TM, we used states to remember things. we can use a state to store a symbol as well. So the state becomes a pair (q, a) where q is the state & a is the tape symbol stored in (q, a).

So the new set of state is given by: $Q \times I$.

iii) Multiple track turing machine

In a multiple track TM, a single tape is assumed to be divided into several tracks. If a single tape is divided into k tracks, then the tape alphabets consist of k-tuples of tape symbols. The only difference between standard TM & TM with multiple tracks is the set of tape symbols. In case of std TM, tape symbols are denoted by symbol I whereas in case of TM with multiple tracks, the tape symbols are denoted by symbol I^k. The working remains same

*Subroutines: We know that subroutines are used in programming languages whenever a task has to be done repeatedly. The same facility can be used in TM & complicated TM's can be built using subroutines. A TM subroutine is a set of states that performs some pre-defined task. The TM subroutine has a start state & a state without any moves. This state which has no moves serves as the return state & passes the control to the state which calls the subroutine.

question-10

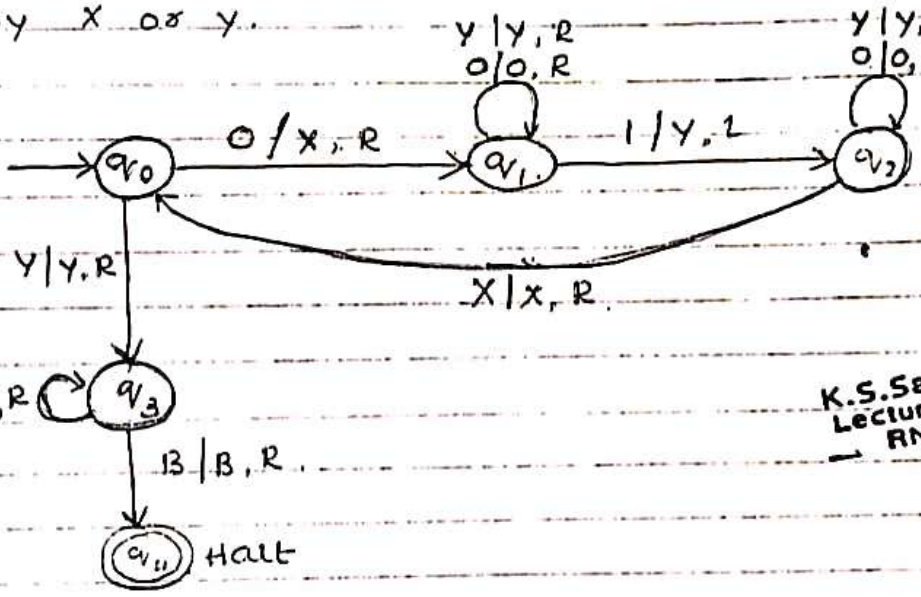
1. Design a TM to accept the language
$L = \{a^n b^n \mid n \geq 1\}$

Soln:

Replace the leftmost 0 by X and change the state to q_1, and then move the read-write head towards right. [because after 0 is replaced, corresponding 1 also needs to be replaced]

Search for the leftmost 1 and replace it by symbol Y and move towards left.

Repeat until all the symbols are replaced by X or Y.



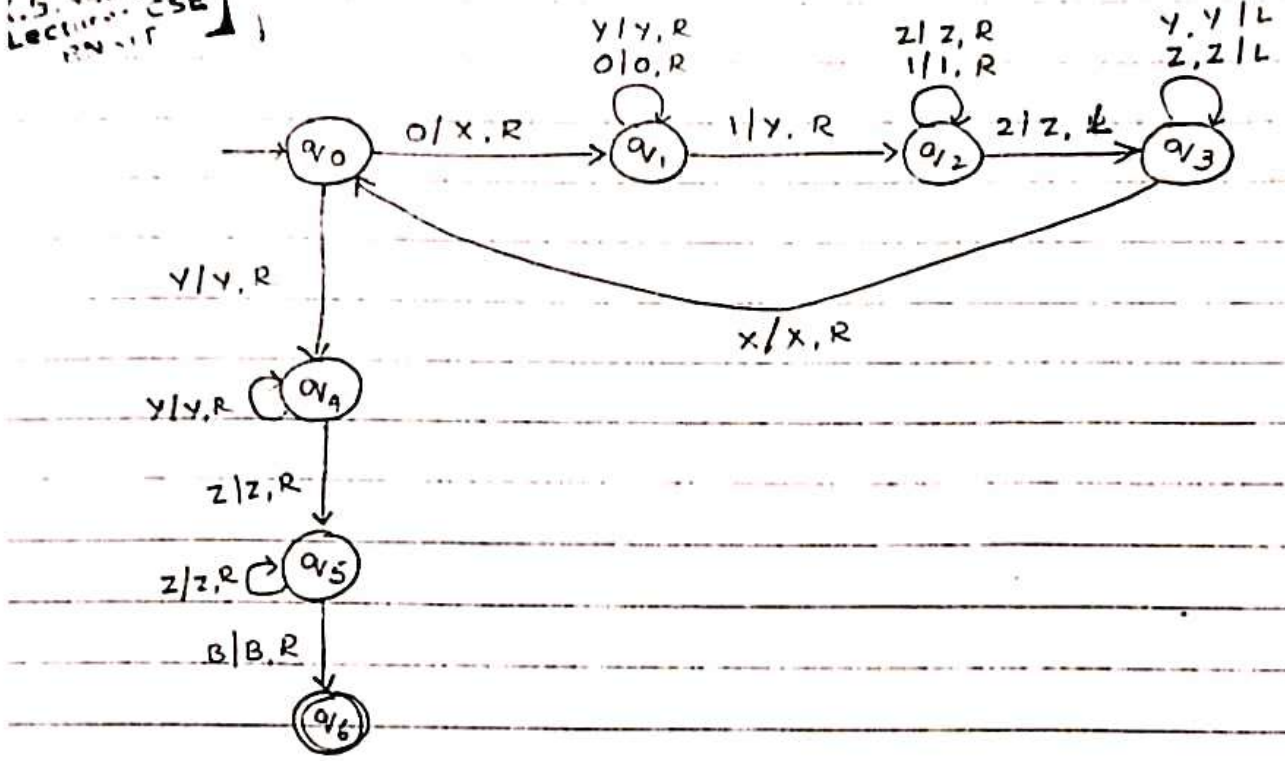
K.S.Sampada
Lecturer CSE
RNSIT

δ	Tape symbol				
States	0	1	X	Y	B
q_0	(q_1, X, R)	-	-	(q_3, Y, R)	-
q_1	$(q_1, 0, R)$	(q_2, Y, L)	-	(q_1, Y, R)	-
q_2	$(q_2, 0, L)$	ϵ -	(q_1, X, R)	(q_2, Y, L)	-
q_3	-	-	-	(q_3, Y, R)	(q_4, B, R)

question-11

3. Design a TM to accept $L = \{0^n 1^n 2^n \mid n \geq 1\}$.

000111222



δ	0	1	2	X	Y	Z	B
q_0	(q_1, X, R)	–	–	–	(q_4, Y, R)	–	–
q_1	$(q_1, 0, R)$	(q_2, Y, R)	–	–	(q_1, Y, R)	–	–
q_2	–	$(q_2, 1, R)$	(q_3, Z, R)	–	–	(q_2, Z, R)	–
q_3	–	–	–	(q_0, X, R)	(q_3, Y, L)	(q_3, Z, L)	–
q_4	–	(q_4, X, R)	–	–	(q_4, Y, R)	(q_5, Z, R)	–
q_5	–	–	–	–	–	(q_5, Z, R)	(q_6, B, R)
q_6							