TM

Turing Machine

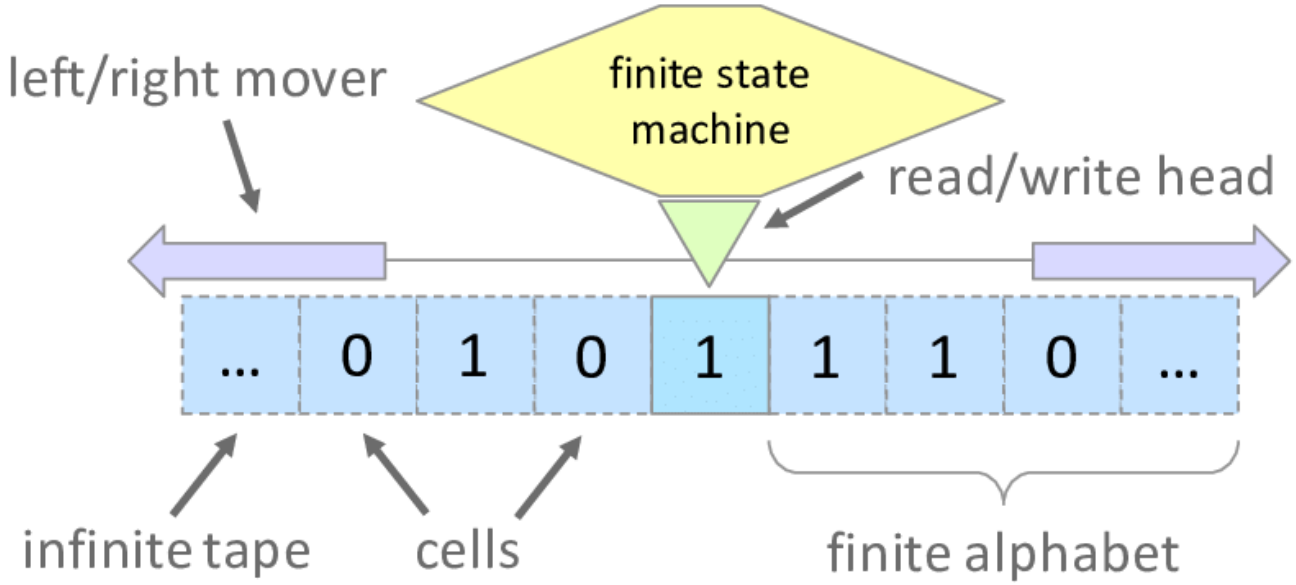
Consists of:-

- **A Tape**

Tape is infinite extending on either sides, divided into cells, each cell capable of holding one symbol

- **Head-**

Associated with tape is read-write head that can travel left or right on tape and read a single symbol on each move



Turing Machine

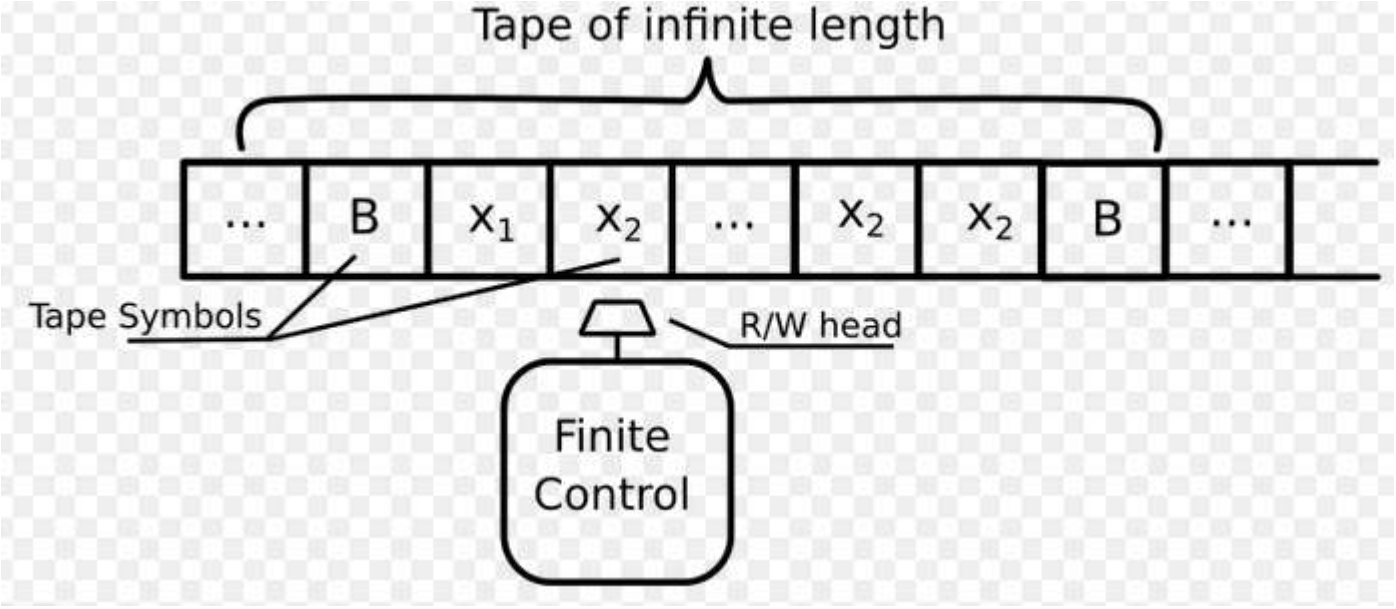
Consists of:-

- **Tape Symbols-**

Finite set of symbols, consists of lowercase letters, digits, usual punctuation marks and the Blank B

- **States-**

Set of states in which the machine can reside



Turing Machine

A Turing machine is a mathematical model of computation that defines an abstract machine[1] that manipulates symbols on a strip of tape according to a table of rules

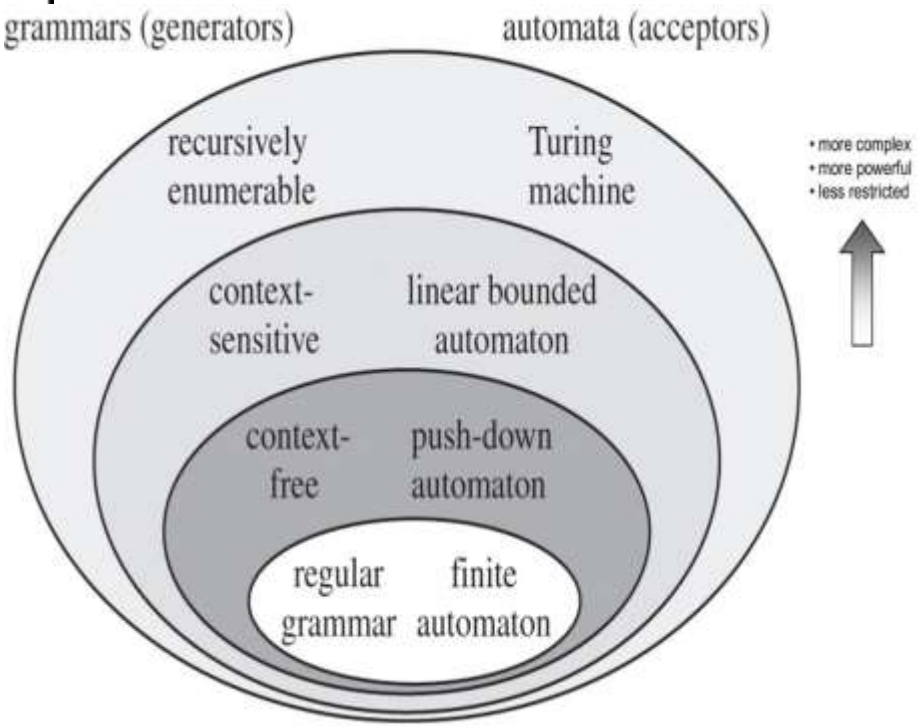
-Wikipedia

Turing Machine

- Turing Machine was invented by Alan Turing in 1936
- Accepts Recursive Enumerable Languages (generated by Type-0 Grammar).

The Hierarchy

Class	Grammars	Languages	Automaton
Type-0	Unrestricted	Recursive Enumerable	Turing Machine
Type-1	Context Sensitive	Context Sensitive	Linear-Bound
Type-2	Context Free	Context Free	Pushdown
Type-3	Regular	Regular	Finite



TM: Formal Definition

TM is denoted by 7 Tuple: $M=(Q, \sum, \Gamma, \delta, q_0, B, F)$ where

- **Q:Finite set of Internal states of the Control Unit**
- **∑:Finite input alphabet**
- **Γ: Finite Set of Tape Symbols**
- **δ:Q X r ->Q X r X {L,R,N}: Transition function(TF)**
- **q_0: Start state of Control Unit, q0 ∈ Q**
- **B : Blank Symbol**
- **F: Set of Final States/ Accept State, F ⊆ Q**
- **∑ ⊆ r -{B}**

δ Function

δ Function:-

- **δ:Q X Γ ->Q X Γ X {L,R,N}: Transition function(TF)**

If δ(q,X)=(p,Y,L)

- **If present state is q and the next symbol on the tape is X then the Turing Machine changes the state to p, replaces the symbol X by symbol Y and moves the tape head one position left.**

If δ(q,X)=(p,Y,R)

- **If present state is q and the next symbol on the tape is X then the Turing Machine changes the state to p, replaces the symbol X by symbol Y and moves the tape head one position right.**

TM Example 1

Transition Rules

Construct a TM to replace each occurrence of ‘a’ by ‘b’ for the strings over ∑={a,b}

TM Example 1

Construct a TM to replace each occurrence of ‘a’ by ‘b’ for the strings over ∑={a,b}

TM Example 2

Design a TM to replace all occurrences of ‘111’ by ‘101’ over ∑={0,1}

TM Example 2

Transition Diagram ,Table

Design a TM to replace all occurrences of ‘111’ by ‘101’ over ∑={0,1}

TM Example 3

Design a TM accept the language containing substring “aba” over ∑={a,b}

Logic-

During the process if TM discovers “aba” on the tape, it halts and thereby accepts entire input string

M=(Q, ∑, Γ, δ, q_0, B, F)

TM Example 3

Design a TM accept the language containing substring “aba” over ∑={a,b}

δ(q0,a)=(q1,a,R)

δ(q0,b)=(q0,b,R) self loop

δ(q1,a)=(q1,a,R) self loop

δ(q1,b)=(q2,b,R)

δ(q2,b)=(q0,b,R) go back

δ(q2,a)=(qf,a,N)

TM Example 4

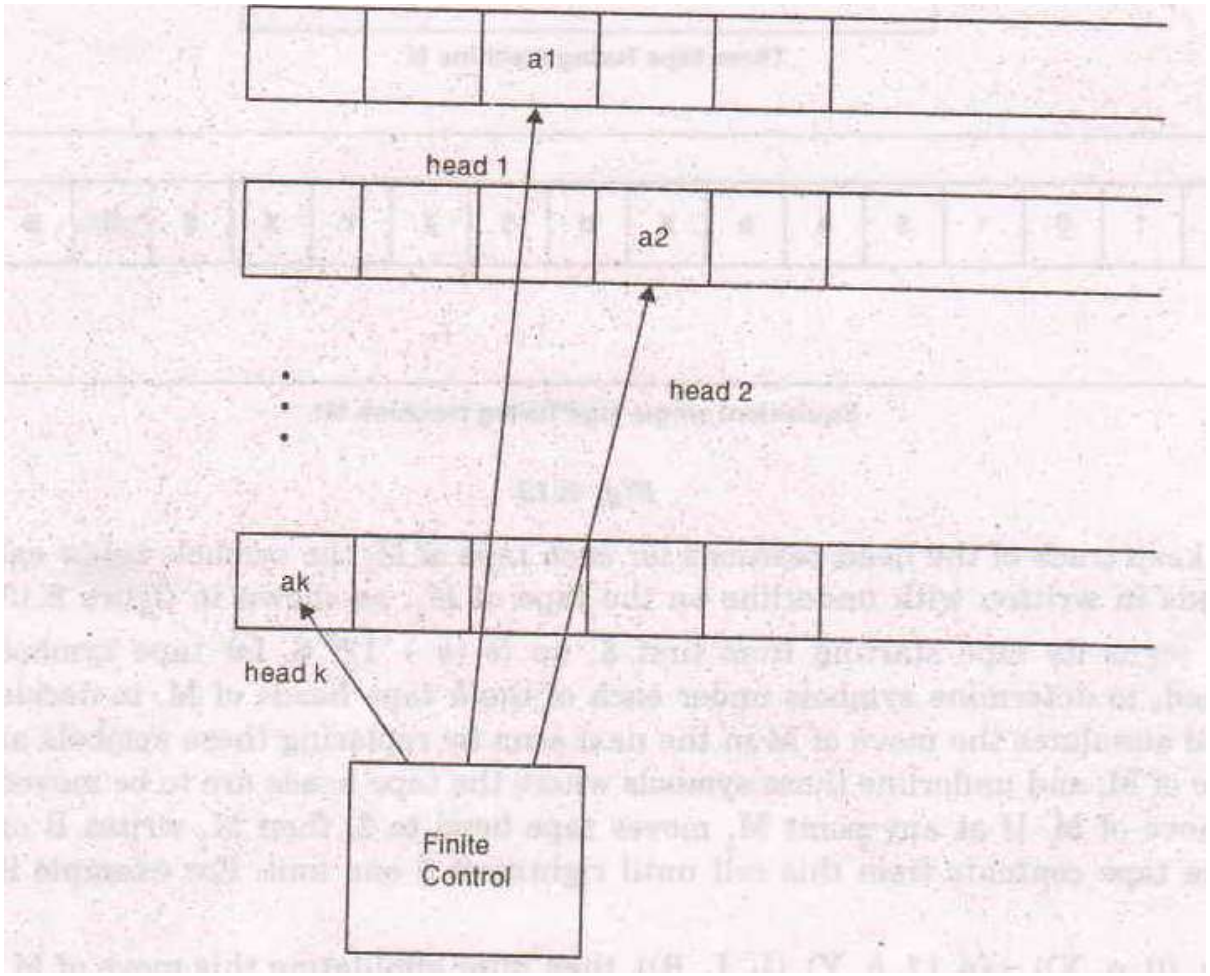
Design a TM to replace all occurrences of ‘101’ by ‘011’ over ∑={0,1}

Modifications of Turing Machines or Variants of Turing Machines

- **Multi Tape Turing Machine**
- **Non-Deterministic Turing Machine**

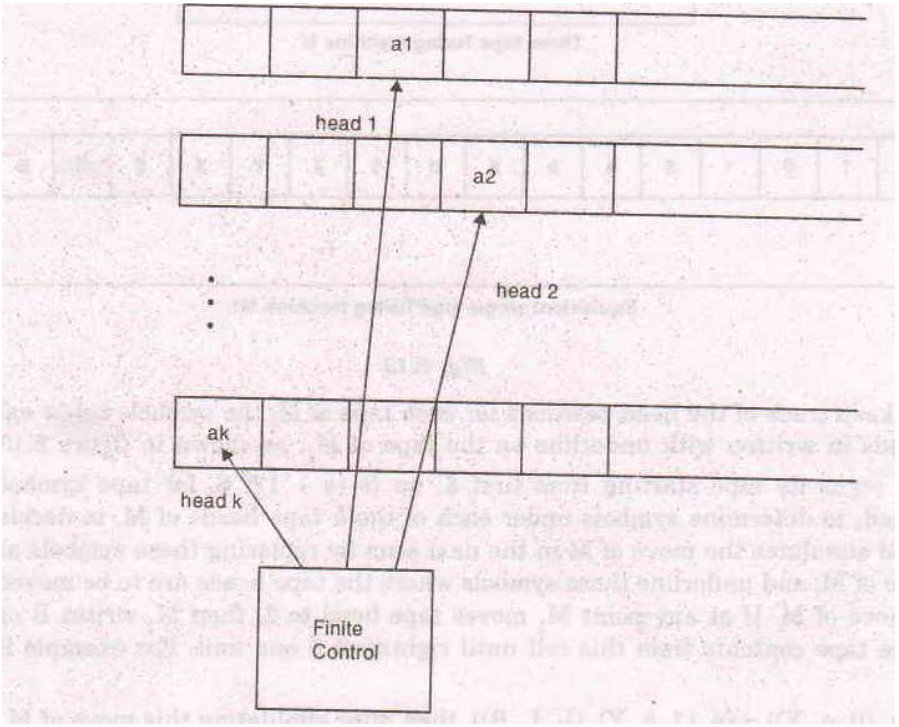
Multi Tape Turing Machine

- **A Multitape Turing Machine-**



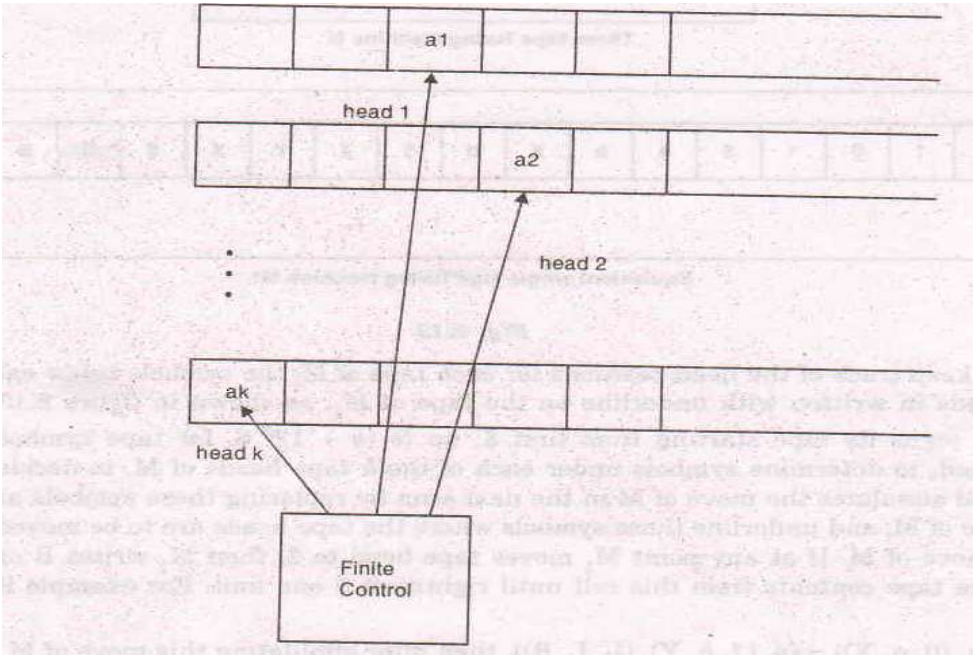
Multi Tape Turing Machine

- **It is a Turing Machine with several tapes , each has its own tape heads**
- **It consists of a finite control with k tape heads and k tapes (k>1)**
- **The move of this Turing machine depends on the present state of the finite control and the symbol scanned by each of the tape heads.**



Multi Tape Turing Machine

- In each move, depending on the present state and the symbols scanned by each of the tape heads , the machine can
  - change state of the finite control
  - print a new symbol on each of cells scanned by each of the tape heads
  - move each tape one cell left or one cell right independently



Multi Tape Turing Machine

- **The transition function of Multitape turing machine is :**
- **δ:Q X {r1 X r2 X ……. rk} ->Q X {r1 X r2 X ……. rk} X { {L,R,N} X {L,R,N} ……{L,R,N} k times}**

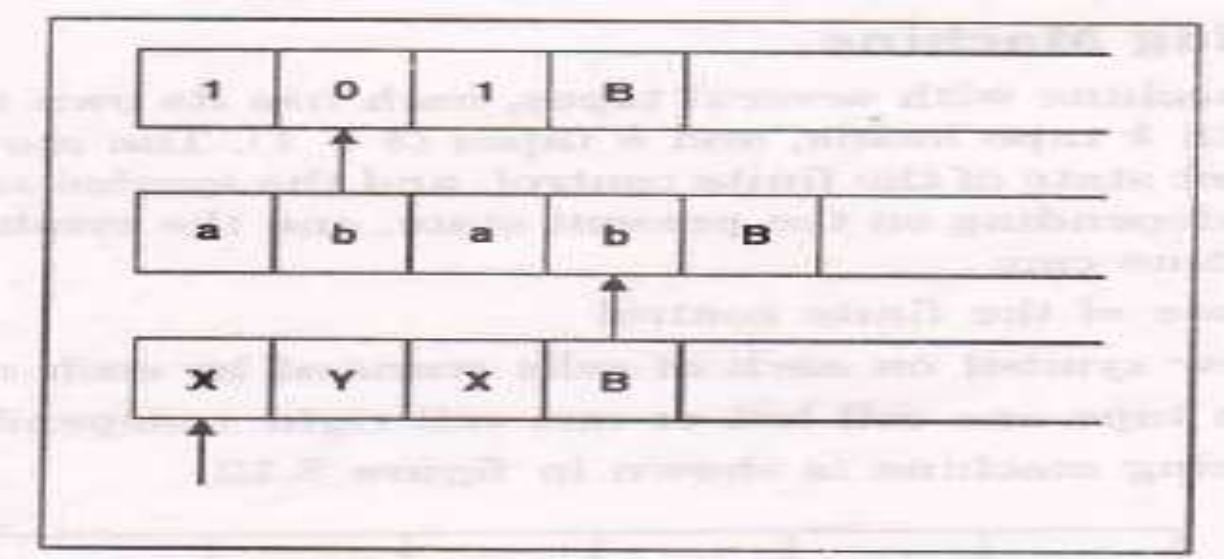
- **δ(q1, (a1,a2,……ak))=(p,(b1,b2,…….bk),(L,R,L,L……k times))**

Multi Tape Turing Machine

- **All Variants have same power, every multitape turing machine has an equivalent single tape Turing machine**

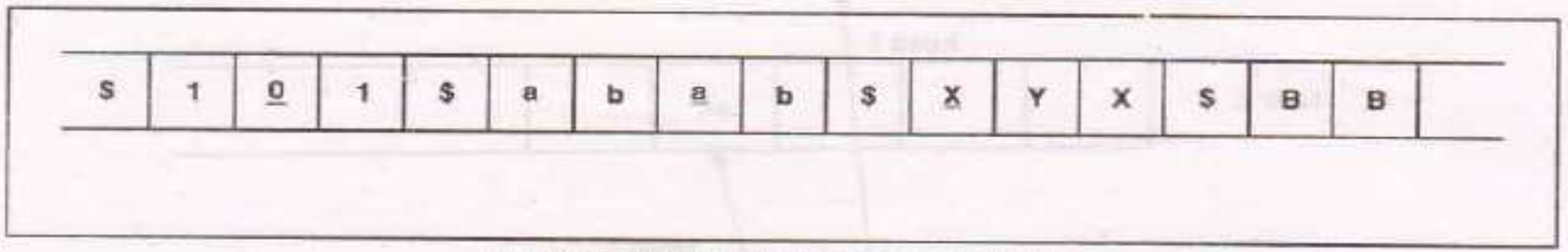
Multi Tape Turing Machine

- **Let M be a k tape Turing Machine(k>1)**



Three tape Turing machine M

- **Let us construct a single tape turing machine M1, which simulates M as follows:-**

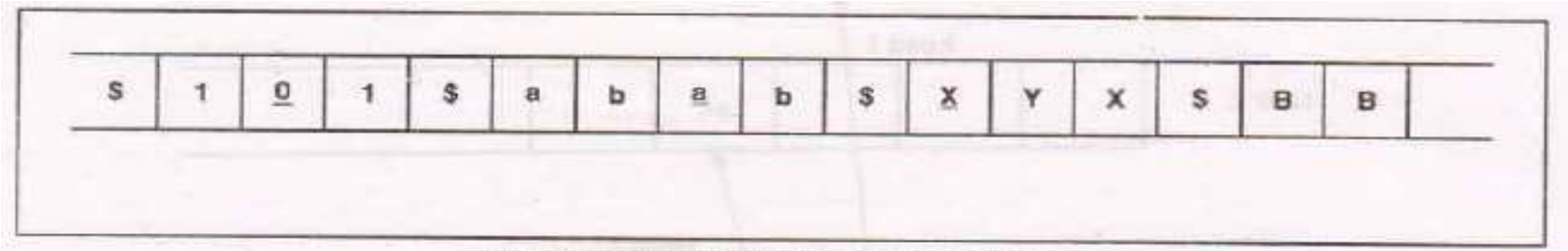


Equivalent single tape Turing machine M1

Multi Tape Turing Machine

- **M1 stores the information available on the k tapes of M on its tape, using a symbol $ as delimiter**

- **To keep a track of the head positions for each tape of M, the symbols below each of these tape heads are written with underline on the tape of M1**

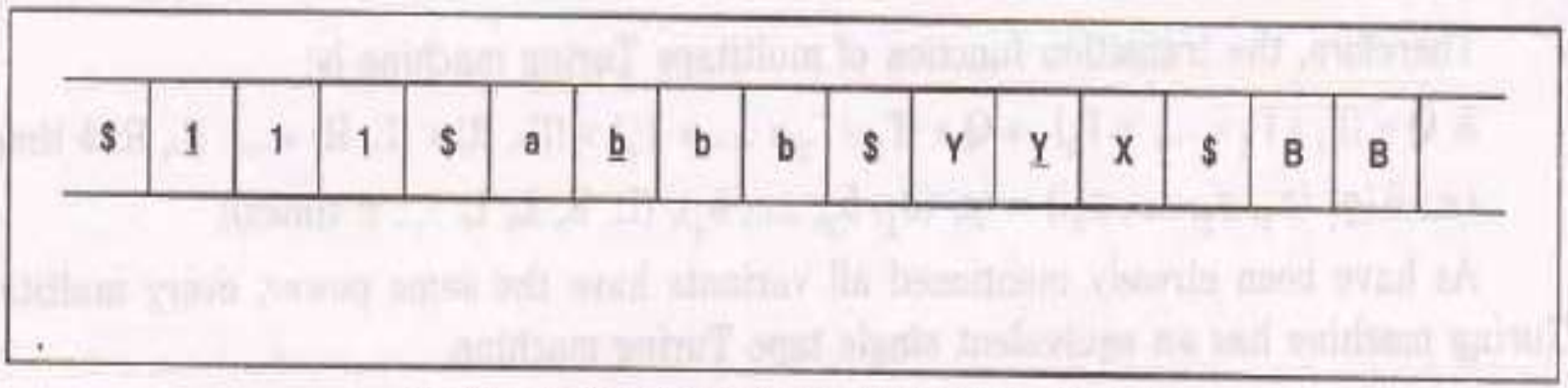


Equivalent single tape Turing machine M1

Multi Tape Turing Machine

- **M1 scans its tape starting from first $ up to (k+1)st $ for tape symbols that are underlined to simulate the move by replacing these symbols according to the move of M**

- **δ(q, (0,a,X))=(p,(1,b,Y),(L,L,R))**
- **After this move the contents of the tape are-**



Contents of tape of M1 after simulating the move of M

Non-Deterministic Turing Machine

- Turing Machine which at any point while carrying out computation , may proceed according to several possibilities
- For a tape symbol scanned, The machine has a finite number of choices of next move
- The transition function defines mapping from
- Q X Γ to a finite subset of Q X Γ X {L,R,N}
- It accepts input, if some sequence of choices of moves lead to an accepting state

Non-Deterministic Turing Machine

- **All variants have the same power, thus Every Non-Deterministic Turing Machine has an equivalent deterministic Turing machine**

Non-Deterministic Turing Machine

- For a Non-Deterministic Turing machine M,
- It is always possible to construct a turing machine M1, which tries all possible sequences of computations or branches of computations of M
- If M enters into an accept state on any of these branches, M1 accepts otherwise continues to simulate Mwithout termination
- Thus, M1 is accepting the same language as M

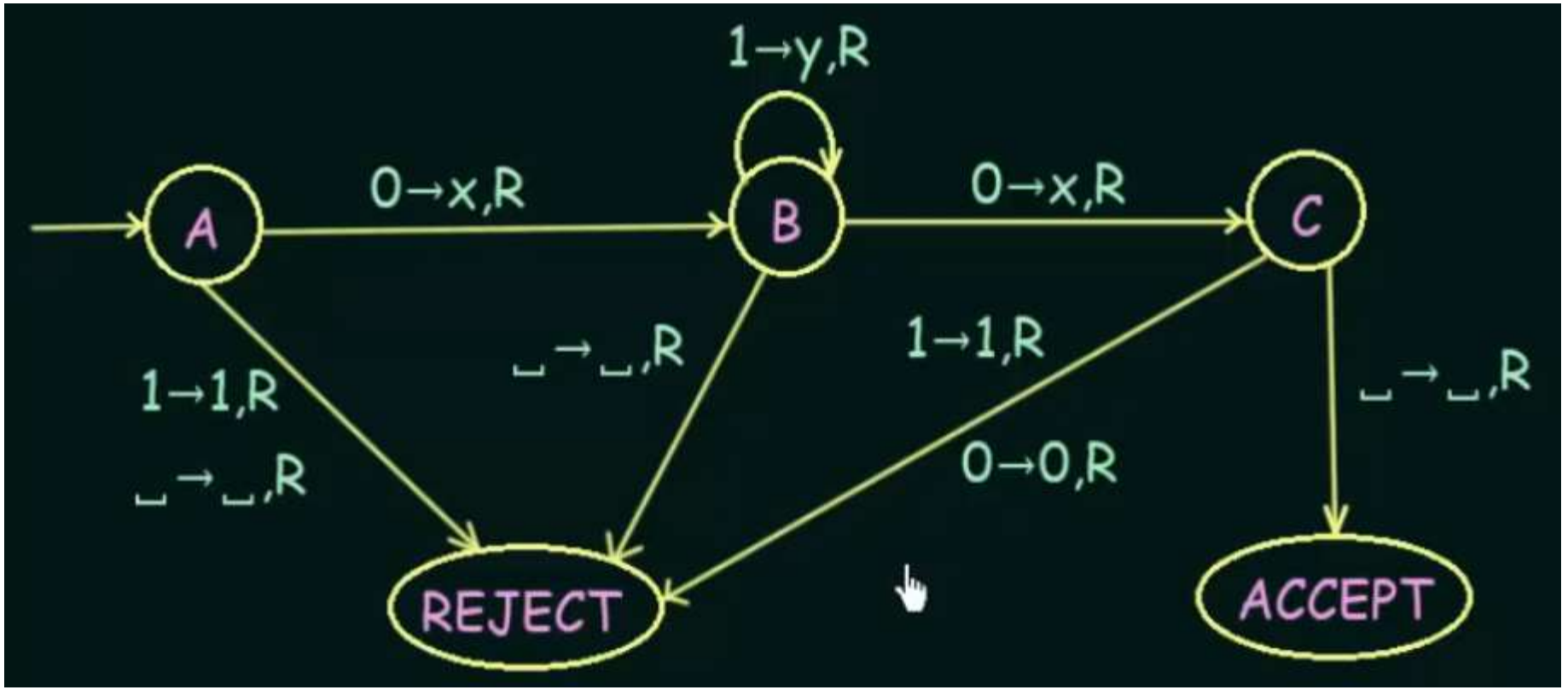
TM Example 5

Design a TM which recognizes the language L=01*0 for ∑={0,1}

Lets take string 0110

TM Example 5

Design a TM which recognizes the language L=01*0 for ∑={0,1}



0	1	1	0	B
X	Y	Y	X	B

TM Example 5

Design a TM which recognizes the language $L=0^N1^N$ for $\sum=\{0,1\}$

TM Example 5

Design a TM which recognizes the language $L=0^N1^N$ for $\sum=\{0,1\}$

Algorithm-

- **Change 0 to X**
- **Move Right to first “1”**
- **If None:Reject**
- **Change “1” to “y”**
- **Move LEFT to Leftmost “0”**
- **Repeat the above steps until no more “0”s**
- **Make sure no more “1”s remain**

TM Example 5

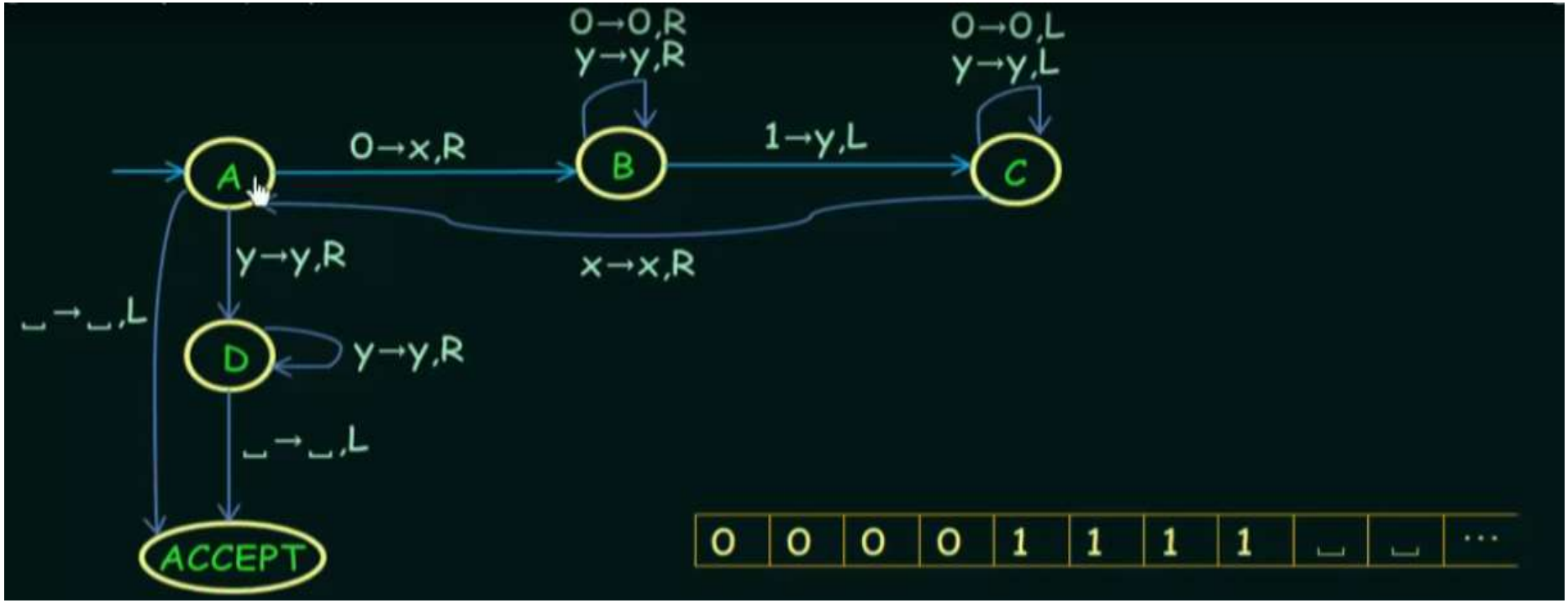
Simulation for $L=0^N1^N$, String 00001111

Algorithm-

- **Change 0 to X**
- **Move Right to first “1”**
- **If None:Reject**
- **Change “1” to “y”**
- **Move LEFT to Leftmost “0”**
- **Repeat the above steps until no more “0”s**
- **Make sure no more “1”s remain**

TM Example 5

Design a TM which recognizes the language $L=0^N1^N$ for $\sum=\{0,1\}$



Algorithm-

Change 0 to X

Move Right to first "1"

If None:Reject

Change "1" to "y"

Move LEFT to Leftmost "0"

Repeat the above steps until no more "0"s

Make sure no more "1"s remain