S R M UNIVERSITY

Programming Techniques for TM Construction

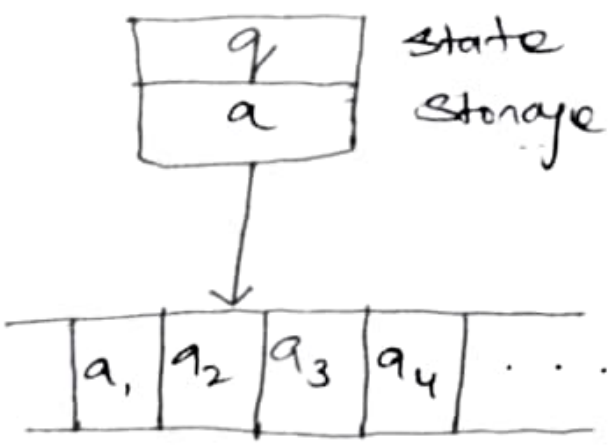
The following are the different techniques of constructing a TM to meet high-level needs

(a) Storage in the finite control (or) State
(b) Multiple Tracks
(c) Subroutines
(d) Checking-off Symbols.

(a) Storage in the State (or) Finite Control

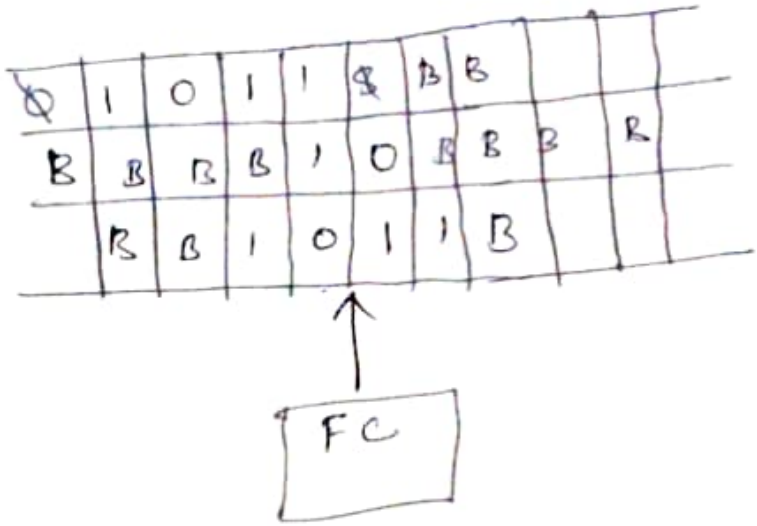
It can also be used to hold a finite amount of information along with task of representing a Position in the Program.

The state is written as a Pair of elements, one for control another storing a Symbol

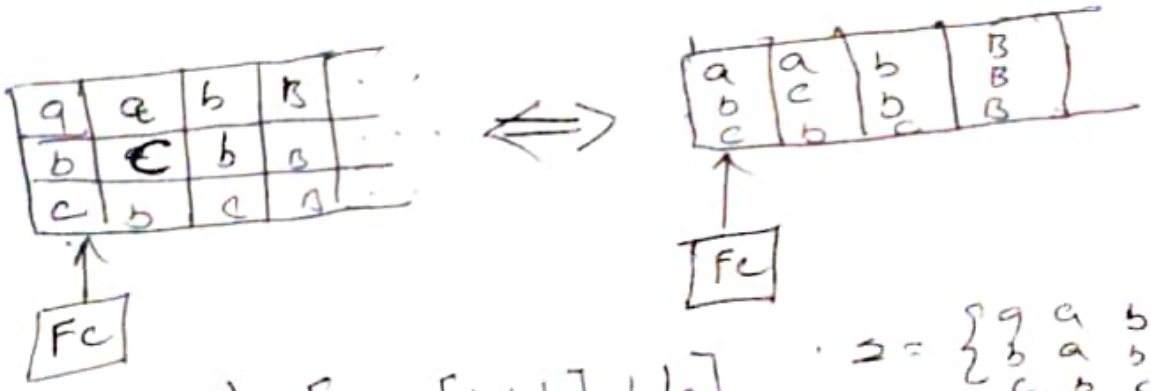


(b) Multiple Tracks

It is also possible that a TM input tape can be divided into several tracks. Each track can hold one symbol, and the tape alphabet of the TM consist of tuple with one component for each track.



Ex:



$\delta(q_0, [a,b,c]) = [q_1, [b,b,b], L/R]$

$\Sigma = \left\{ \begin{matrix} a & a & b \\ b & a & b \\ c & b & c \end{matrix} \right\}$

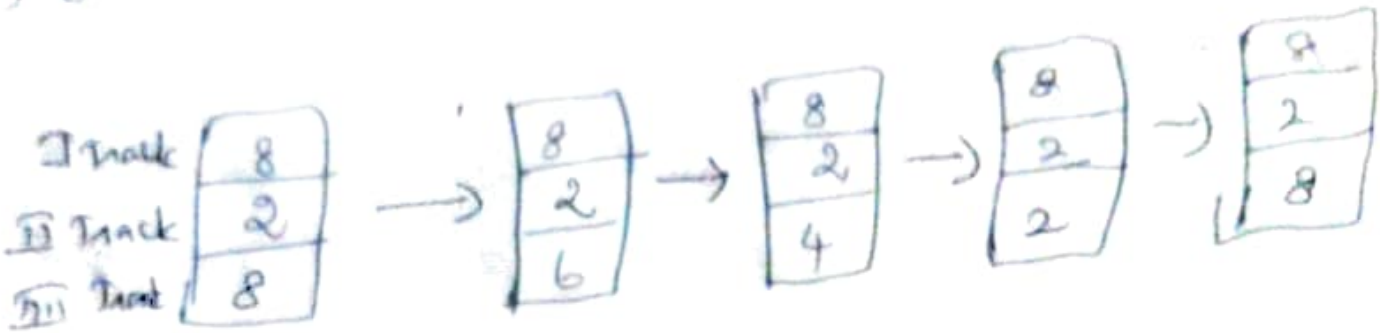
$\left\{ q_0, \begin{matrix} a \\ b \\ c \end{matrix} \right\} = \left(q_1, \begin{matrix} b \\ b \\ c \end{matrix} \right)$

eg: whether the given i/p is Prime (n) not

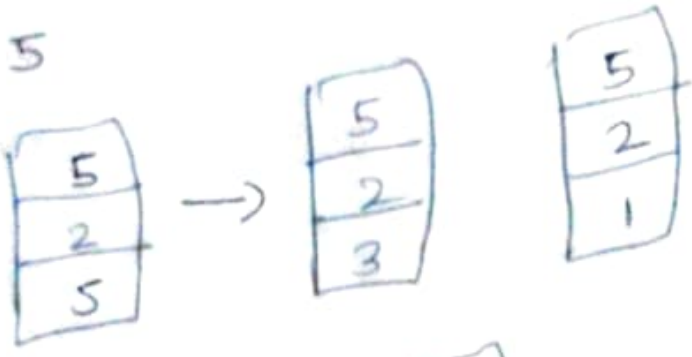
Logic

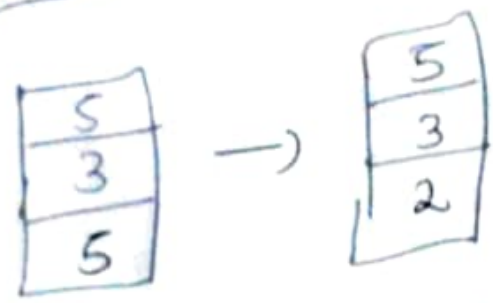
The number n the second track is subtracted from n the third track as many times possible till getting the reminder. If the reminder is zero, then the No n the 1st track is not Prime

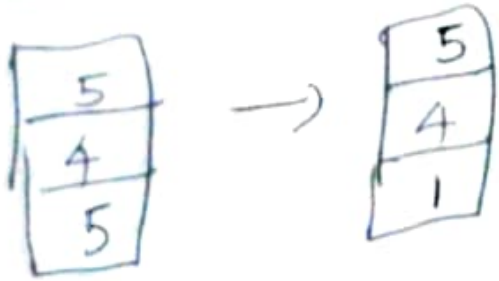
(i) 8



(ii) 5

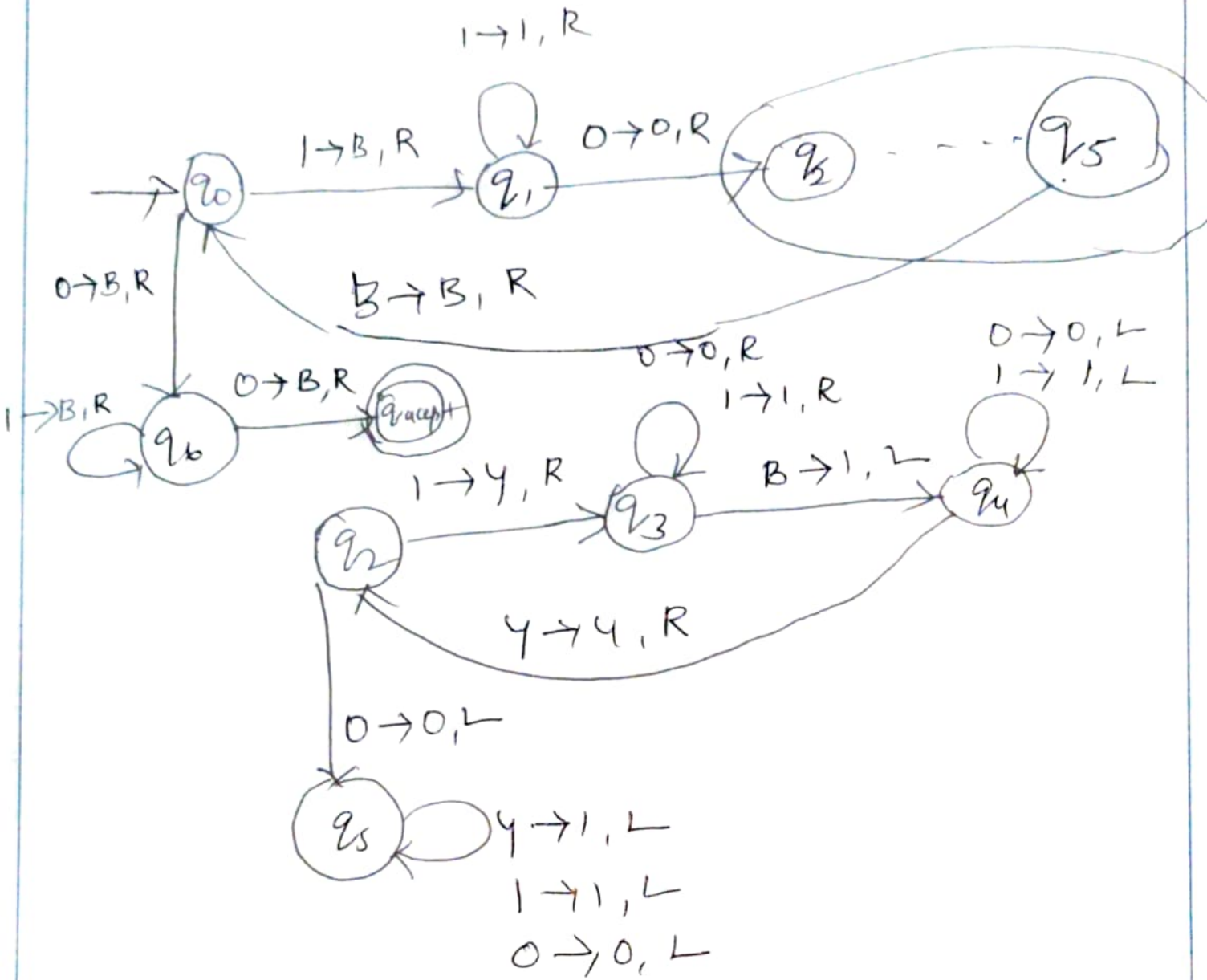






Here Number in I st track = II nd track

The given Number is Prime


1→1, R
1→B, R
0→0, R
q0
q1
q2
q5
0→B, R
B→B, R
0→0, R
1→B, R
0→B, R
q6
q accept
1→1, R
0→0, L
1→1, L
1→Y, R
B→1, L
q3
q4
q2
Y→Y, R
0→0, L
q5
Y→1, L
1→1, L
0→0, L

(3) Checking off symbols

Here we track of the tape can be used to Mark the symbols already read.

(4) Subroutines

A problem with some tasks to be repeated for many times, can be Programmed using subroutine. A TM with subroutine is a set of states that Perform some useful process.

The idea here is to write Part of TM Programs to serve as a subroutine which has its own initial state & return state for returning to the calling routine. It improves the modular (or) top down Programming design.

Multiplication in Turing Machine

Eg:-

$2 \times 3 \rightarrow 2+2+2$
$3+3$

11011110