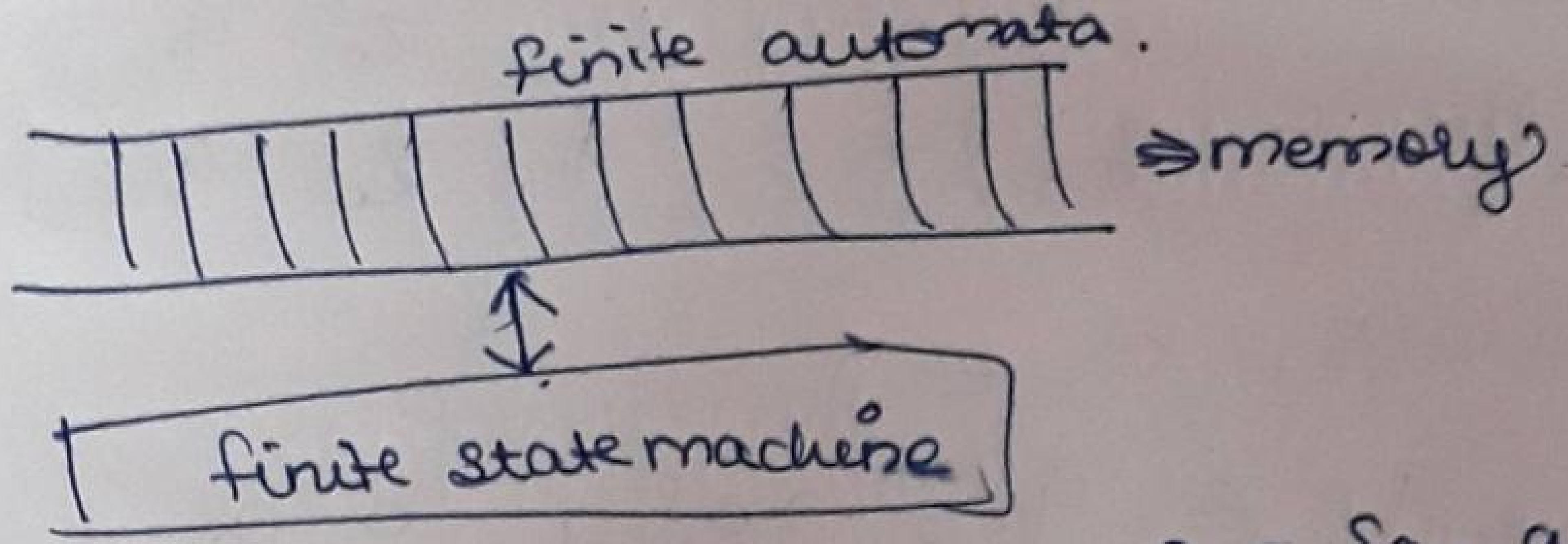
$a^n b^n c^n \mid n \geq 0$ [cant ~~not~~ be done using PDA. So, turing machine]

TURING MACHINE



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

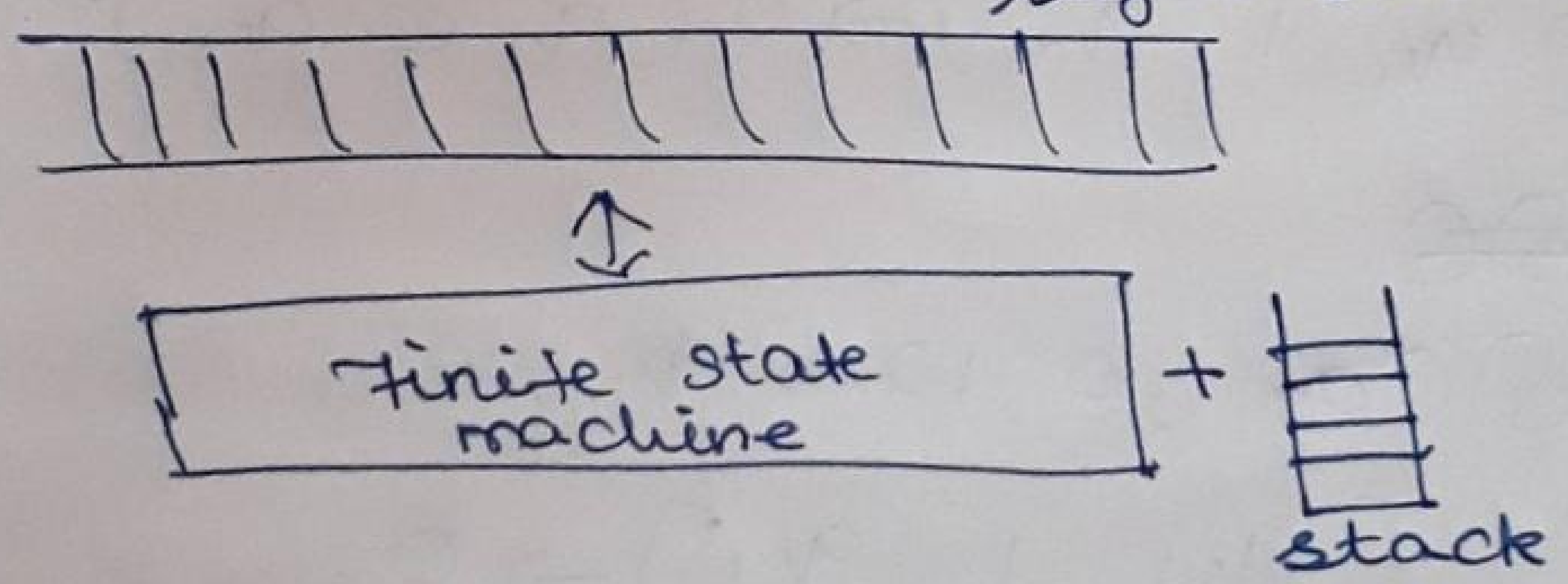
$\delta(q, \epsilon) = Q.$

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

Disadvantage: Moves 1 direction (left to right).
less memory.

PDA

* This also left to right but xtra memory



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

$\delta = (q, \Sigma, \Gamma = Q + \Gamma^*$

Turing machine:

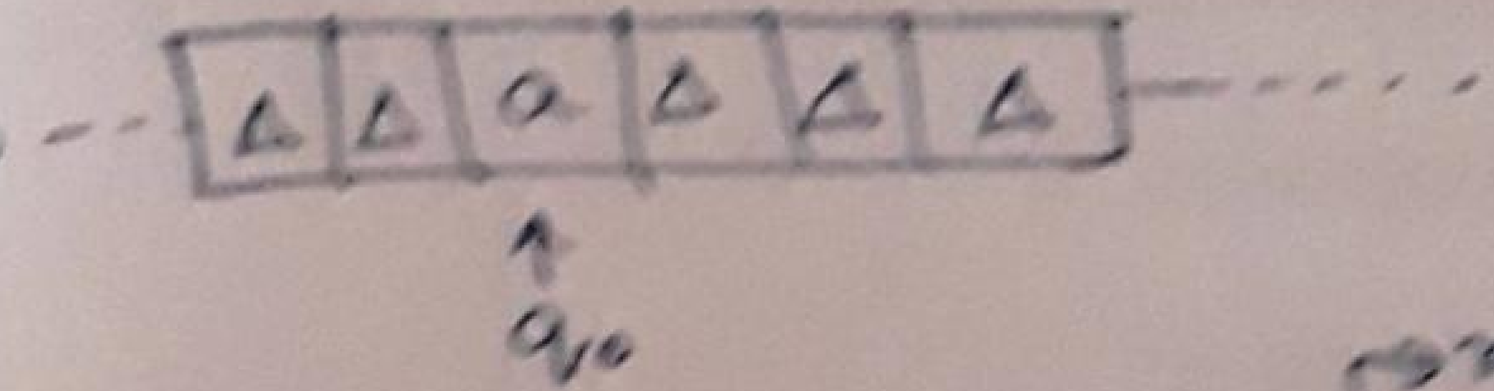
⊛ unrestricted memory

⊛ bidirectional reading ($L \to R$ (or) $R \to L$)

$\delta(q, a) = (Q, x, \underbrace{\{L R\}}_{\text{either L or R}} \}$

$\delta(Q * \Gamma) \rightarrow Q * \Gamma * \{L, R\}$

Δ → blank

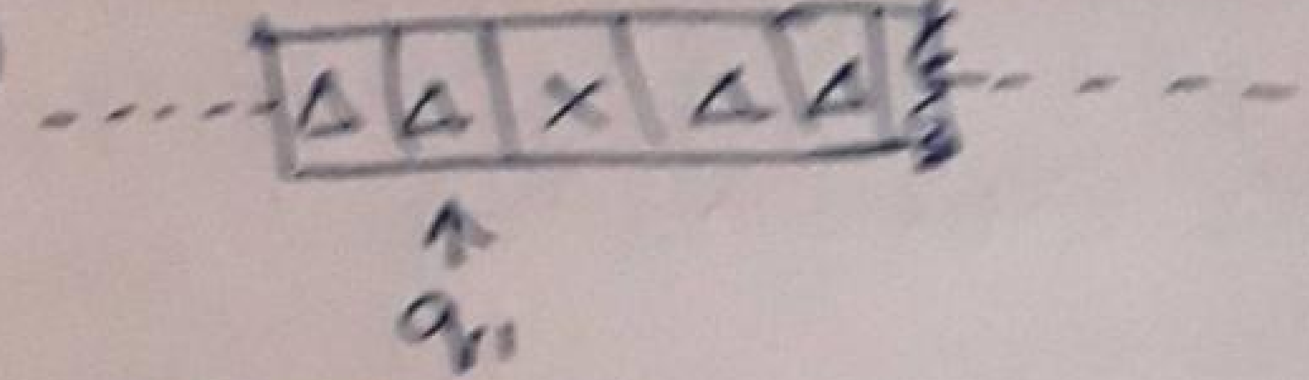
① 

→ replaces a to x

$\delta(q_0, a) = (q_1, x, L)$

↳ moves cursor to left

⇓

② 

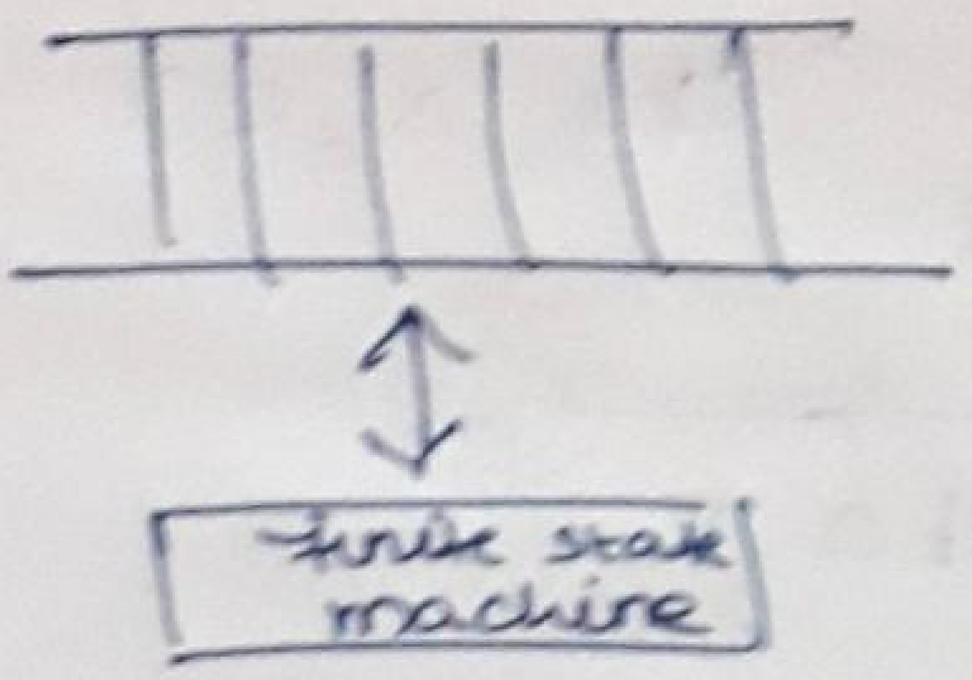
~~δ(q₁~~

⊛ from final state we cannot go anywhere on previous scenario (cant go beyond)

⊛ Turing machine used for unrestricted grammar.

⊛ Turing machine → Alen Turing, 1936.

⊛ No stack.

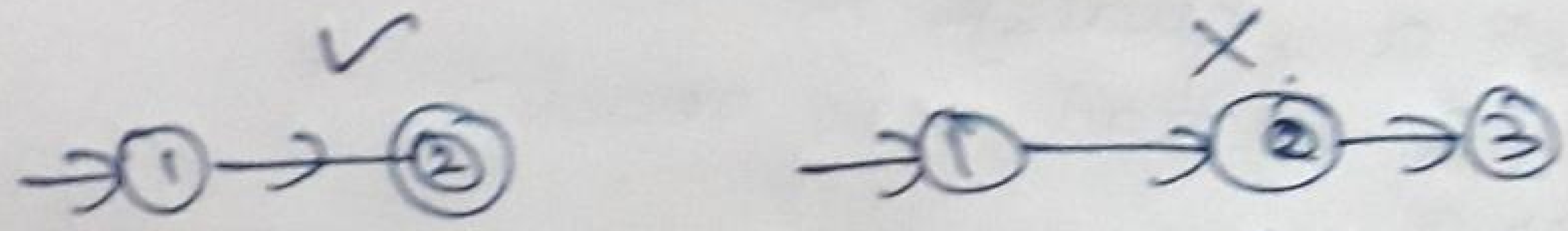


→ read / write tape

⊛ $T = (Q, \Sigma, F, q_0, \delta, \Gamma, \Delta)$

→ tape symbol (set of all stack symbols)

↳ blank / special symbol



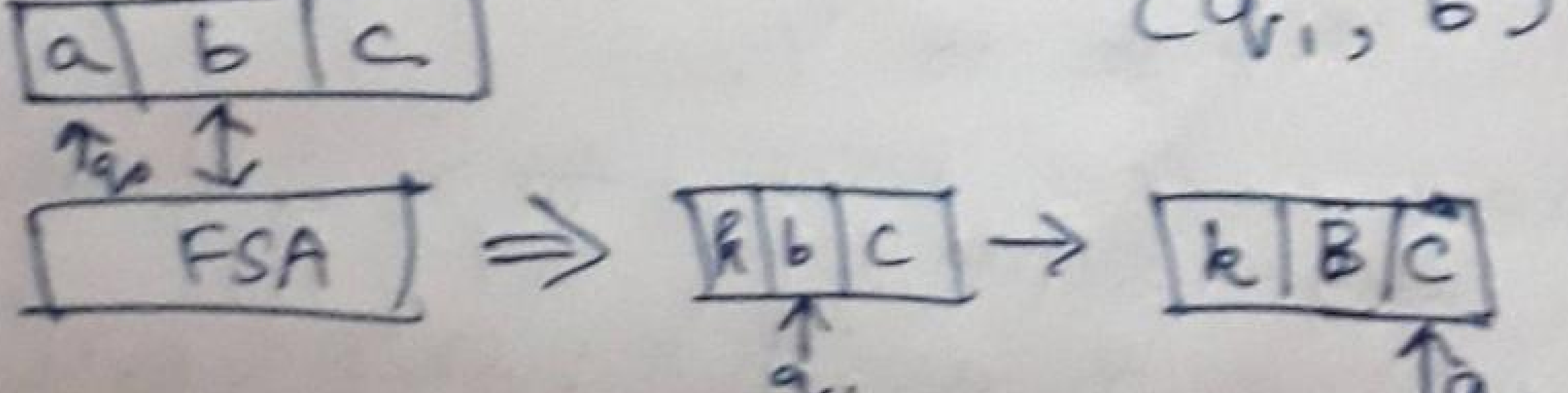
Here:

$\Gamma = K, B$

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

$(q_0, a) = (q_1, k, R)$

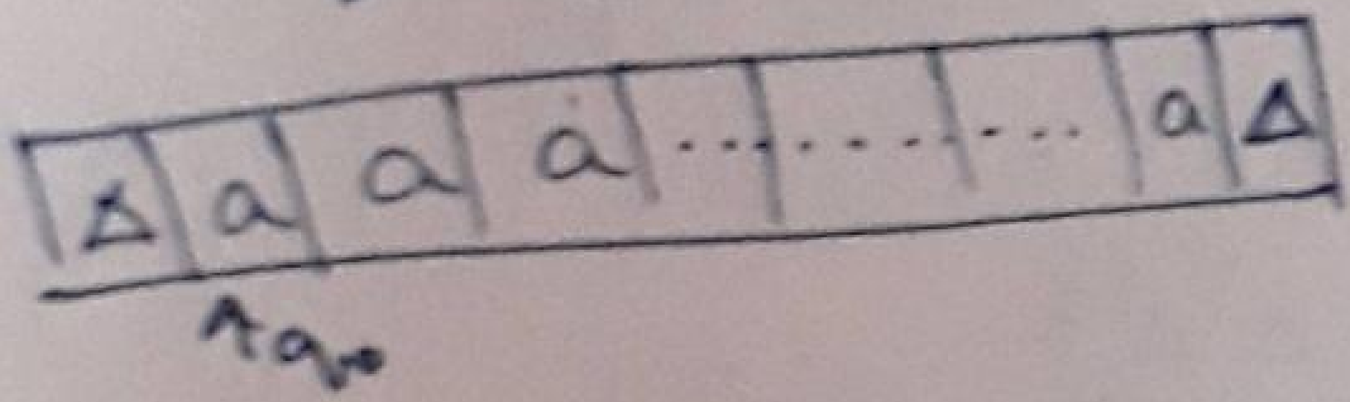
$(q_1, b) = (q_1, B, R)$



Scanned by TapScanner

1) Design a turing machine for the following language.

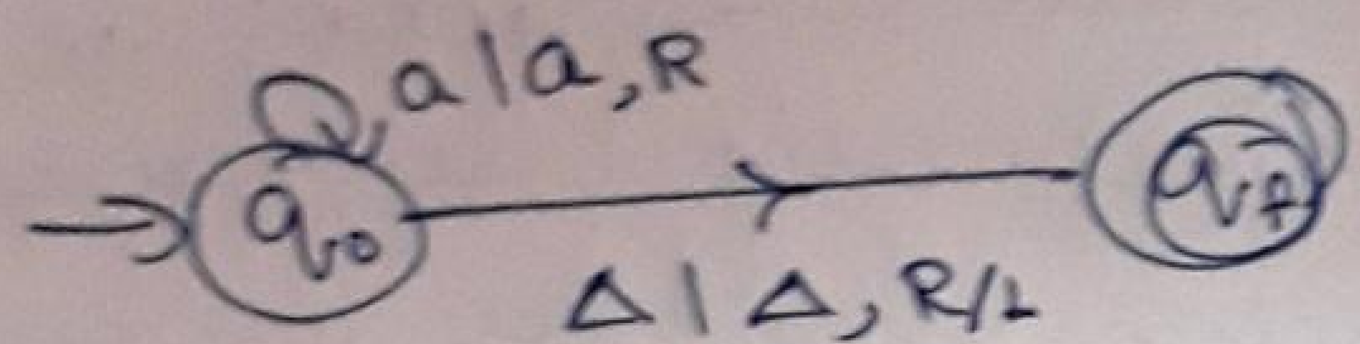
$L = \{a^*\}$



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

$\vdots$

$\delta(q_0, \Delta) = (q_f, \Delta, R)$

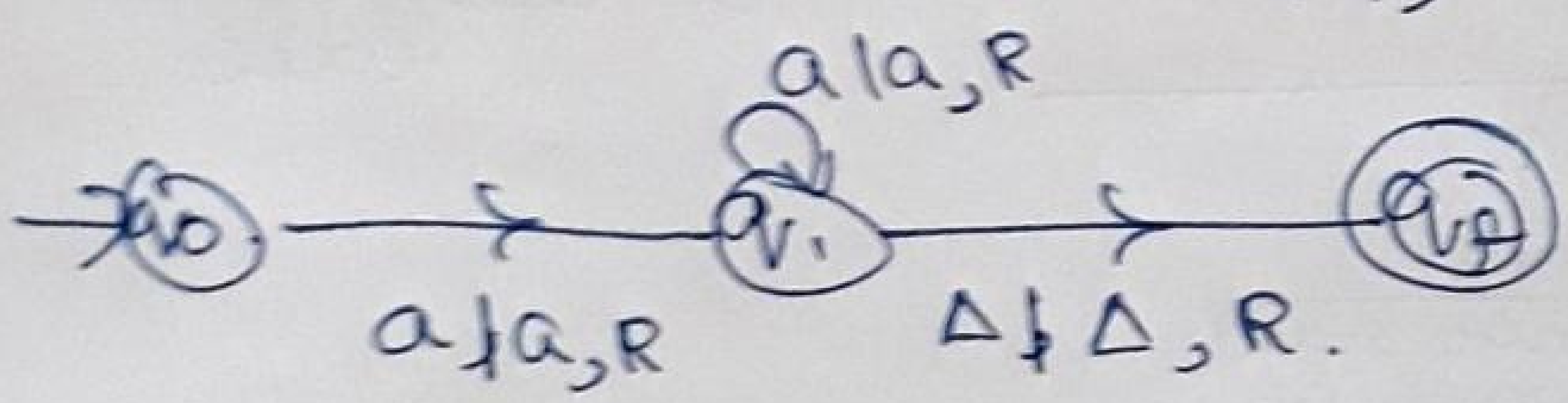


2) $L = \{a^+\}$

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

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

$\delta(q_1, \Delta) = (q_f, \Delta, R)$



3) $L = \{a^n b^n \mid n \geq 1\}$

eg: $a^3 b^3$ = aaabbb

→ first a, change to x.
(a is unread x is read)

→ Skip a.

→ then when we find b, change to Y.

→ then come back to x & change to next state.

→ Replace a with x; Skip a & y.

→ Replace b with Y. repeat same

→ Find blank → accepted.

q_0 Δ a a a b b b Δ

Δ X a a b b b Δ

q_1

Δ X a a Y b b Δ

q_1

q_1 Δ X X a Y b b Δ

q_2 Δ X X a Y Y b Δ

q_2 Δ X X X Y Y b Δ

q_2 Δ X X X Y Y Y Δ

accepted cuz blank

$\delta(q_0, a) = (q_1, X, R).$

$\delta(q_1, a) = (q_1, a, R).$

$\delta(q_1, b) = (q_2, Y, L)$

$\delta(q_2, a) = (q_2, a, L$

~~$\delta(q_2, a) = (q_2,$~~

$\delta(q_2, X) = (q_0, X, R)$

$\delta(q_1, Y) = (q_1, Y, R)$

$\delta(q_1, b) = (q_2, Y, L).$

$\delta(q_2, Y) = (q_2, Y, L)$

$\delta(q_0, Y) = (q_3, Y, R)$

$\delta(q_3, \Delta) = (q_f, \Delta, R).$

$\delta(q_3, y) = (q_3, y, R).$

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

$\delta(q_0, a) = (q_1, X, R)$ Xaabbbccc

$\delta(q_1, a) = (q_1, a, R)$ Xaabbbccc

$\delta(q_1, b) = (q_2, Y, R)$ XaaYbbccc

$\delta(q_2, b) = (q_2, b, R)$ XaaYbbccc

$\delta(q_2, c) = (q_3, Z, L)$ XaaYbbZcc

$\delta(q_3, b) = (q_3, b, L)$ XaaYbbZcc

$\delta(q_3, Y) = (q_3, Y, L)$ XaaYbbZcc

$\delta(q_3, a) = (q_3, a, L)$ XaaYbbZcc

$\delta(q_3, X) = (q_0, X, R)$ XXaYbbZcc

$\delta(q_1, Y) = (q_1, Y, R)$

$\delta(q_2, Z) = (q_2, Z, R)$

$\delta(q_3, Z) = (q_3, Z, L)$

$\delta(q_0, Y) = (q_4, Y, R)$

$\delta(q_0, X) = (q_4, X, R)$

$\delta(q_4, Y) = (q_4, Y, R)$

$\delta(q_4, Z) = (q_4, Z, R)$

$\delta(q_4, \Delta) = (q_f, \Delta, R)$

$L = \{w \in \{a,b\}^* \mid n(a) = n(b)\}$

eg: ababbbaa / babbabaa

b a b b a b a a.

X Y X skip Y

X Y X X Y skip Y

X Y X X Y X Y Y.

$\delta(q_0, \Delta) \rightarrow (q_1, \Delta, R)$

$\delta(q_0, a) \rightarrow (q_2, X, R)$.

$\delta(q_0, b) \rightarrow (q_3, X, R)$

$\delta(q_2, a) \rightarrow (q_2, a, R)$

~~$\delta(q_3, b) \rightarrow (q_3, b, R)$~~

$\delta(q_2, y) \rightarrow (q_2, y, R)$

$\delta(q_3, b) \rightarrow (q_3, b, R)$

$\delta(q_3, y) \rightarrow (q_3, y, R)$.

$\delta(q_2, b) \rightarrow (q_4, Y, L)$.

$\delta(q_3, a) \rightarrow (q_4, Y, L)$.

$\delta(q_4, X) \rightarrow (q_1, X, R)$

$\delta(q_4, y) \rightarrow (q_1, Y, R)$.

$\delta(q_1, \Delta) \rightarrow (q_f, \Delta, R)$.

$\delta(q_1, a) \rightarrow (q_2, X, R)$.

$\delta(q_1, b) \rightarrow (q_2, X, R)$

$\delta(q_4, a) \rightarrow (q_4, a, R)$

$\delta(q_4, b) \rightarrow (q_4, b, R)$.

$\delta(q_4, y) \rightarrow (q_4, y, R)$

Instaneous description

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

solve, aaabbb: $\vdash \underline{q_0} aaabbb$.

$\vdash X\underline{q_1 a}abbb$

$\vdash Xa\underline{q_1 a}bbb$

$\vdash Xaa\underline{q_1 b}bb$

$\vdash Xa\underline{q_2 a}\cancel{b}Ybb$

$\vdash X\underline{q_2 a}aYbb$

$\vdash q_2 XaaYbb$

$\vdash Xq_0aaYbb$

$\vdash XXq_1aYbb$

$\vdash XXaq_1Ybb$

$\vdash XXaYq_1bb$

$\vdash XXaq_2YYb$

$\vdash XXq_2aYYb$

$\vdash Xq_2XaYYb$

$\vdash XXq_0aYYb$

$\vdash XXXq_1YYb$

$\vdash XXXYq_1Yb$

$\vdash XXXYYq_1b$

$\vdash XXXYq_2YY$

$\vdash XXXq_2YYY$

$\vdash XXq_2XYYY$

$\vdash XXXq_0YYY$

$\vdash XXXYq_0YY$

$\vdash XXXYYq_0Y$

$\vdash XXXYYYq_0\Delta$

$\vdash q_f \Delta R$

$\cancel{\vdash XXXXX}$

$\cancel{\vdash XXXXYY}$

Try for $w \in \{a,b\}^*$, $n(a) = n(b)$

abbbaa

$L = \{a^n b^n c^n, n \geq 1\} \Rightarrow aabbcc$

$\underline{q_0} aabbcc \vdash x\underline{q_1}abbcc \vdash xa\underline{q_1}bbcc$

$\vdash xaY\underline{q_2}bcc \vdash xaYb\underline{q_2}cc \vdash xaY\underline{q_3}bzc$

$\vdash xa\underline{q_3Y}bzc \vdash x\underline{q_3}aYbzc \vdash \underline{q_3X}aYbzc$

$\vdash X\underline{q_0}aYbzc \vdash XX\underline{q_1}Ybzc \vdash XXY\underline{q_1}bzc$

$\vdash XXYYq_2Zc \vdash XXYYZ\underline{q_2c} \vdash XXYY\underline{q_3}ZZ$

$\vdash XXYq_3YZZ \vdash XXq_3YYZZ \vdash Xq_3XYYZZ$

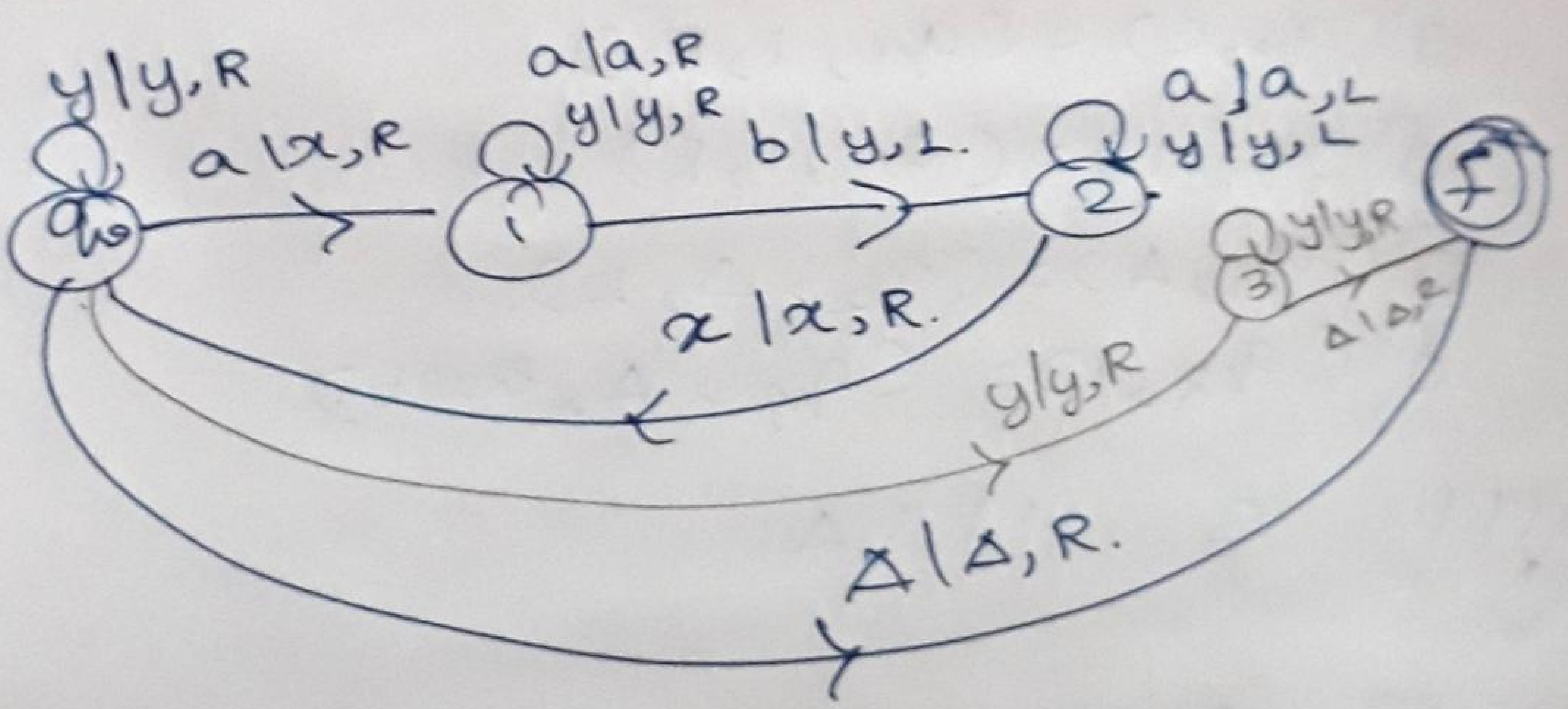
$\vdash q_3XXYYZZ \vdash X\underline{q_0}XYYZZ \vdash XX\underline{q_4}YYZZ$

$\vdash XXYq_4YZZ \vdash XXYYq_4ZZ \vdash XXYYZq_4Z$

$\vdash XXYYZZq_4\Delta \vdash q_f\,\Delta, e$

$L = \{a^n b^n \mid n \geq 0\}$. diagram:

For $n \geq 1$



Try: $L = \{a^i b^j c^k \mid i = j+k\}$.

aaabbc.

Turing machine

accepting device

eg: $L = \{a^n b^n, n \geq 0\}$

$L = \{a^n b^n c^n, n \geq 0\}$

computing device

eg: 1's complement,
2's complement,
+, -, *, ÷, concatenation of binary / unary numbers

Computing Device

1) Design turing machine for concatenation of two unary strings (either 0 (or) 1).

$w_1 . w_2 = w$.

$w_1 = 1111 \quad w_2 = 11111$

Δ	1	1	1	1	c	1	1	1	1	1	Δ

↑ q_0 $\quad w_1 \quad$ ↓ seperator $\quad w_2$.

$\delta(q_0, 1) = (q_0, 1, R)$.

$\delta(q_0, c) = (q_1, 1, R)$

$\delta(q_1, 1) = (q_1, 1, R)$

$\delta(q_1, \Delta) = (q_2, \Delta, L)$.

$\delta(q_2, 1) = (q_f, \Delta, \text{halt})$

eg: Δ111 c 1111Δ
↑ q_0

111 c 1111Δ
↑ q_0

111 1 1111Δ
↑ q_1

111 1 11111Δ
↑ q_1

111 1 11111Δ
↑ q_2

111 1 11111Δ
↑ q_f

2) Design for $x+1$ (unary)

eg: if $x = 5$.

Δ	1	1	1	1	1	Δ

Δ	1	1	1	1	1	1

$\delta(q_0, 1) = (q_0, 1, R)$

$\delta(q_0, \Delta) = (q_0, 1, \text{halt})$.

3). Design for $x+3$.

$\delta(q_0, 1) = (q_0, 1, R)$

$\delta(q_0, \Delta) = (q_1, 1, R)$.

$\delta(q_1, \Delta) = (q_2, 1, R)$.

$\delta(q_2, \Delta) = (q_3, 1, \text{halt})$.

4) Design for 1's complement.

eg: 00000 ⇒ 11111.
11111 ⇒ 00000.
10101 ⇒ 01010.

$\delta(q_0, 0) = (q_1, 1, R)$

$\delta(q_0, 1) = (q_1, 0, R)$.

$\delta(q_1, 1) = (q_1, 0, R)$

$\delta(q_1, 0) = (q_1, 1, R)$

$\delta(q_1, \Delta) = (q_f, \Delta, \text{halt})$.

5) Design for 2's complement. eg: $10101 = \begin{array}{r} 01010 \\ 1 \\ \hline 01011 \end{array}$

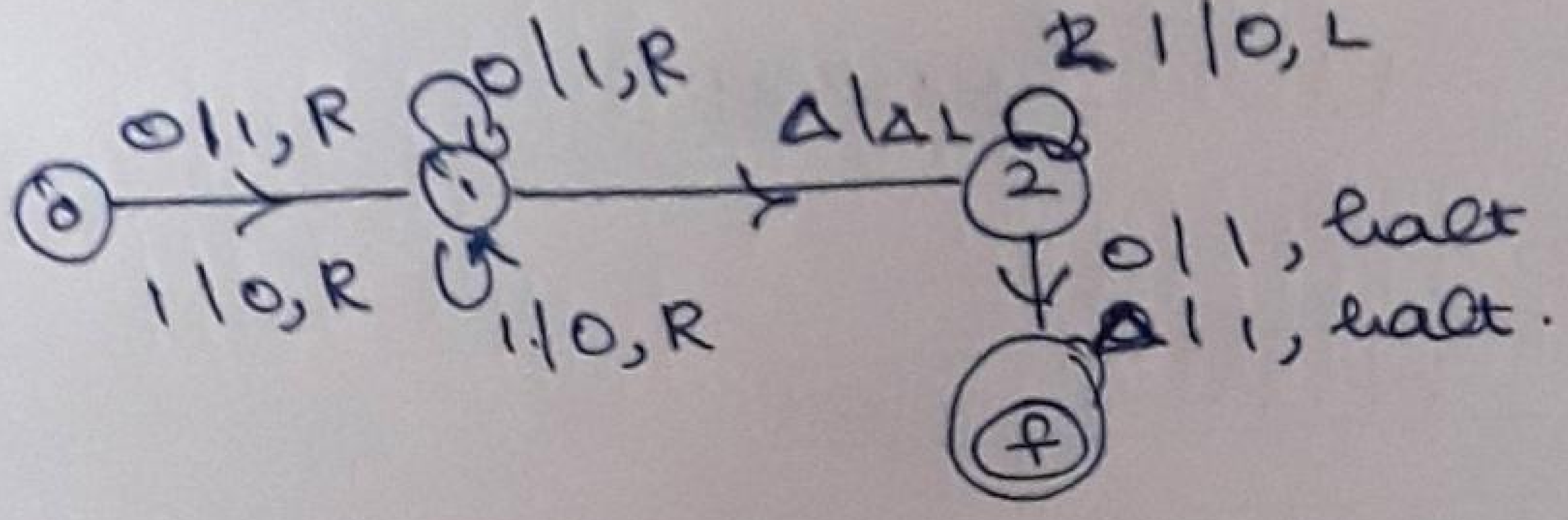
$\delta(q_1, \Delta) = (q_2, \Delta, L)$

$\delta(q_2, 1) = (q_2, 0, L)$

$\delta(q_2, 0) = (q_f, 1, \text{halt})$

$\delta(q_2, \Delta) = \delta(q_f, 1, \text{halt})$

Scanned by TapScanner



6) compute $f(x) = 2(x)$

eg: $2(3) = 111111$; $2(2) = 1111$

Take, $x = 2$

$\delta(q_0, 1) = q_1, X, R$	1 1 Δ Δ ↑q_0
$\delta(q_1, 1) = q_1, 1, R$	X 1 Δ Δ. ↑q_1
$\delta(q_1, \Delta) = q_2, Y, L$	X 1 Δ Δ ↑q_1
$\delta(q_2, 1) = q_2, 1, L$	X 1 Y Δ ↑q_2
$\delta(q_2, X) = q_0, X, R$	X 1 Y Δ ↑q_2 X 1 Y Δ. ↑q_0
$\delta(q_1, Y) = q_1, Y, R$	X X Y Δ Δ ↑q_1
$\delta(q_2, Y) = q_2, Y, L$	X X Y Δ Δ ↑q_1
$\delta(q_0, Y) = q_3, Y, R$	X X Y Y Δ ↑q_2 ↑q_2 ↑q_1
$\delta(q_3, Y) = q_3, Y, R$	X X Y Y Δ ↑q_0 ↑q_3 ↑q_3
$\delta(q_3, \Delta) = q_4, \Delta, L$	X X Y Y Δ ↑q_4
$\delta(q_4, Y) = q_4, 1, L$	X X Y 1 ↑q_4
$\delta(q_4, X) = q_4, 1, L$	X 1 1 1 ↑q_4
$\delta(q_4, \Delta) = q_f, \Delta$, halt	Δ 1 1 1 1 ↑q_4

Scanned by TapScanner

7) construct for proper subtraction :

$x \geq y$ then $x - y$

eg : 5 - 3. = 2

C1 | Δ | 1 | 1 | 1 | 1 | 1 | C | 1 | 1 | 1 | Δ | Δ | Δ |

C2 : 3 - 5 = 0

| Δ | 1 | 1 | 1 | C | 1 | 1 | 1 | 1 | 1 | Δ | Δ | Δ |

C3 : 3 - 0 . = 3

| Δ | 1 | 1 | 1 | C | Δ | Δ | Δ | Δ | Δ |

C4 : 0 - 3 = 0.

| Δ | C | 1 | 1 | 1 | Δ | Δ | Δ | Δ |

$L(G) = \{ x \geq y \quad x - y$

eg : 11111 - 11

$\delta(q_0, 1) = q_1, \Delta, R.$

$\delta(q_1, 1) = q_1, 1, R$

$\delta(q_1, C) = q_2, C, R.$

$\delta(q_2, 1) = q_3, C, L$

$\delta(q_3, C) = q_3, C, L.$

$\delta(q_3, 1) = q_3, 1, L$

$\delta(q_3, \Delta) = q_0, \Delta, R$

$\delta(q_2, C) = q_2, C, R$

$\delta(q_2, \Delta) = q_4, \Delta, L$

$\delta(q_4, C) = q_4, C, L$

$\delta(q_4, 1) = q_4, 1, L$

$\delta(q_4, \Delta) = q_f, 1, \text{halt}.$

8) construct where L(m) gives balanced branches

eg: ( ), ((( ))), (( )). etc...
( )( )(( ))

Variation of turing machine.

Till now we were using infinite tap (infinite blanks in both L & R).

7 types of variation:

(1) turing machine with stay option
(2) multiple track TM.
(3) semi infinite tape TM.
(4) offline TM
(5) multitape TM
(6) multi dimensional TM.
(7) non deterministic TM.

Note: Each classes of TM have the same power with standard T.M.

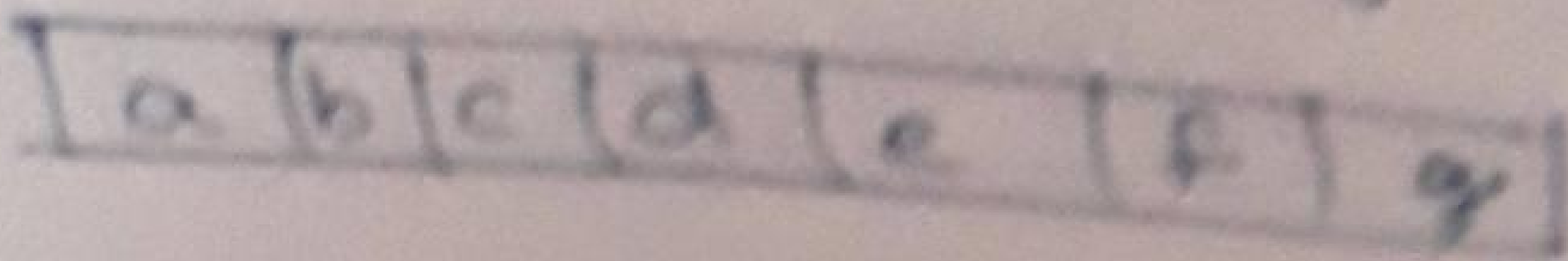
$L(M_1) = L(M_2)$ & vice versa.

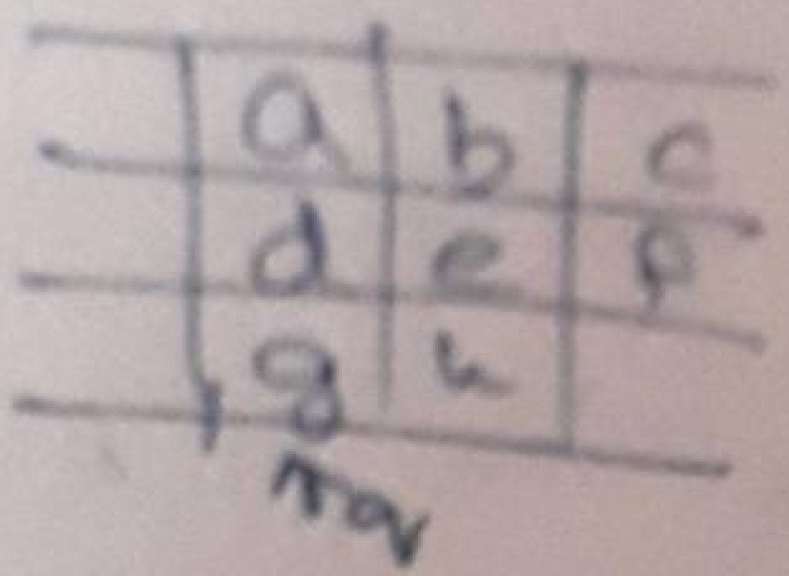
① Turing machine with stay option:

$\delta(Q \times \Gamma \rightarrow Q \times \Gamma^* * \{L, R, S\}$ stay.

*left right (or) stay.

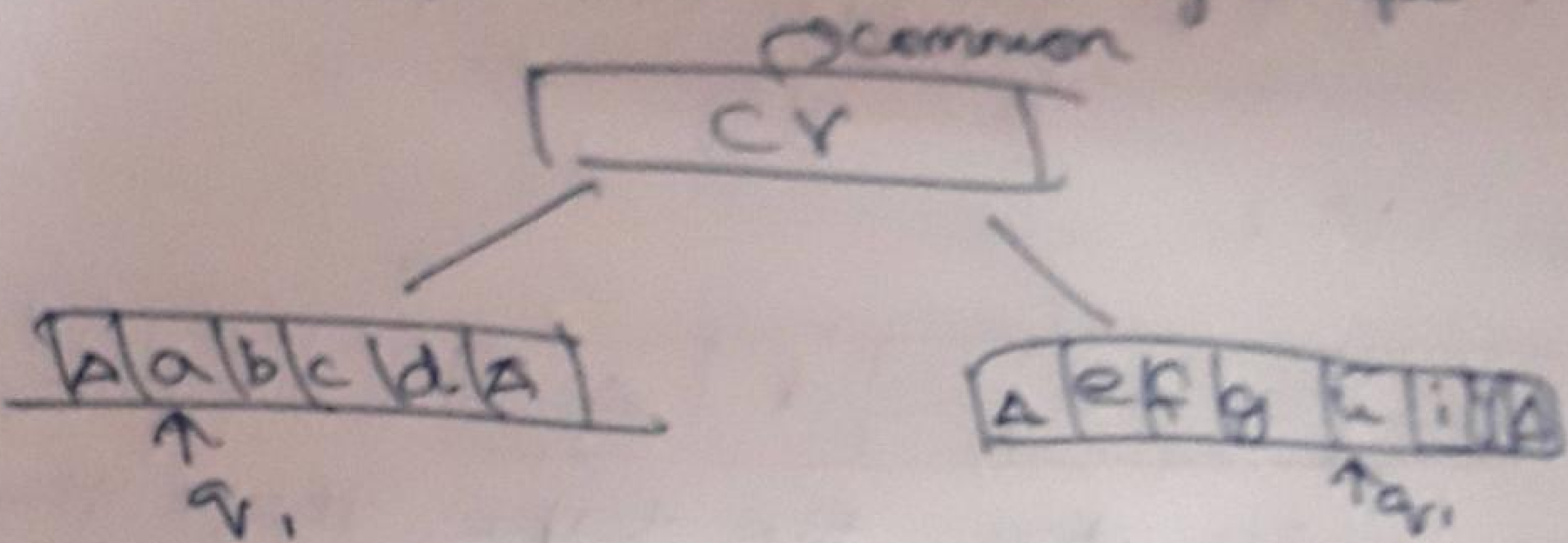
② Multiple track T.M: single tape



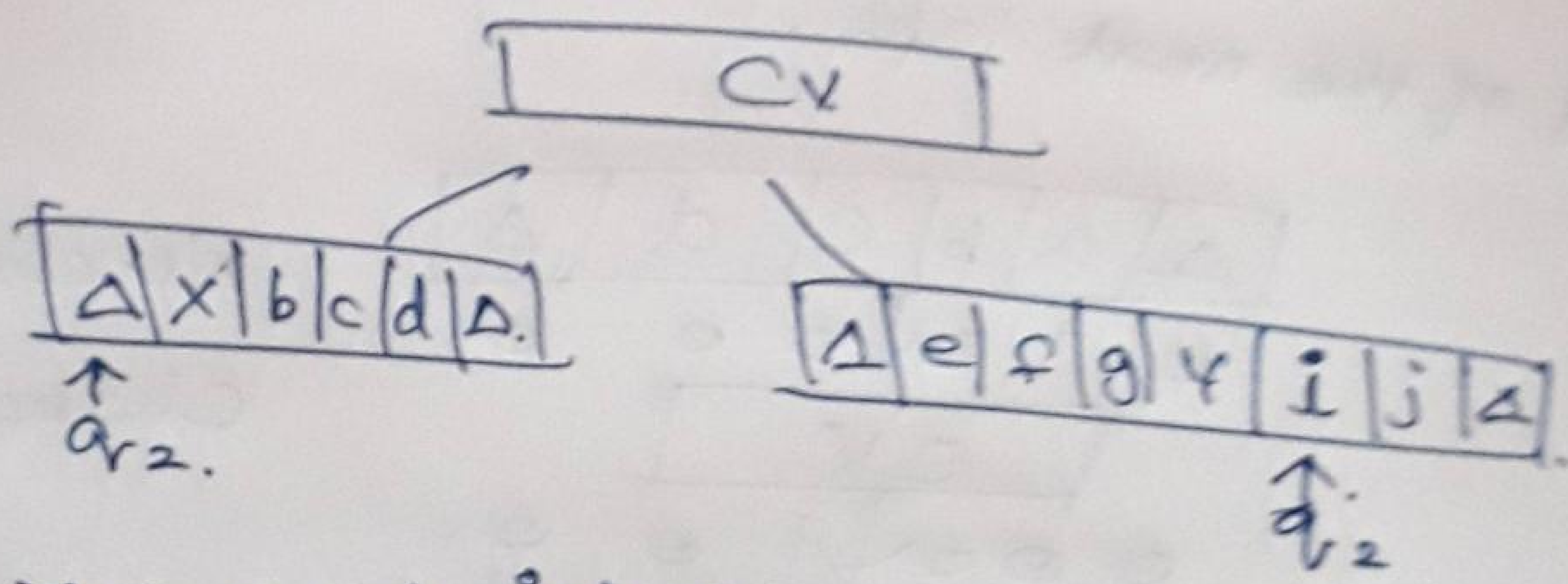


$\delta(q_1, (a, d, g)) = q_2, (x, y, g), R)$.

③ multitape TM: many tape

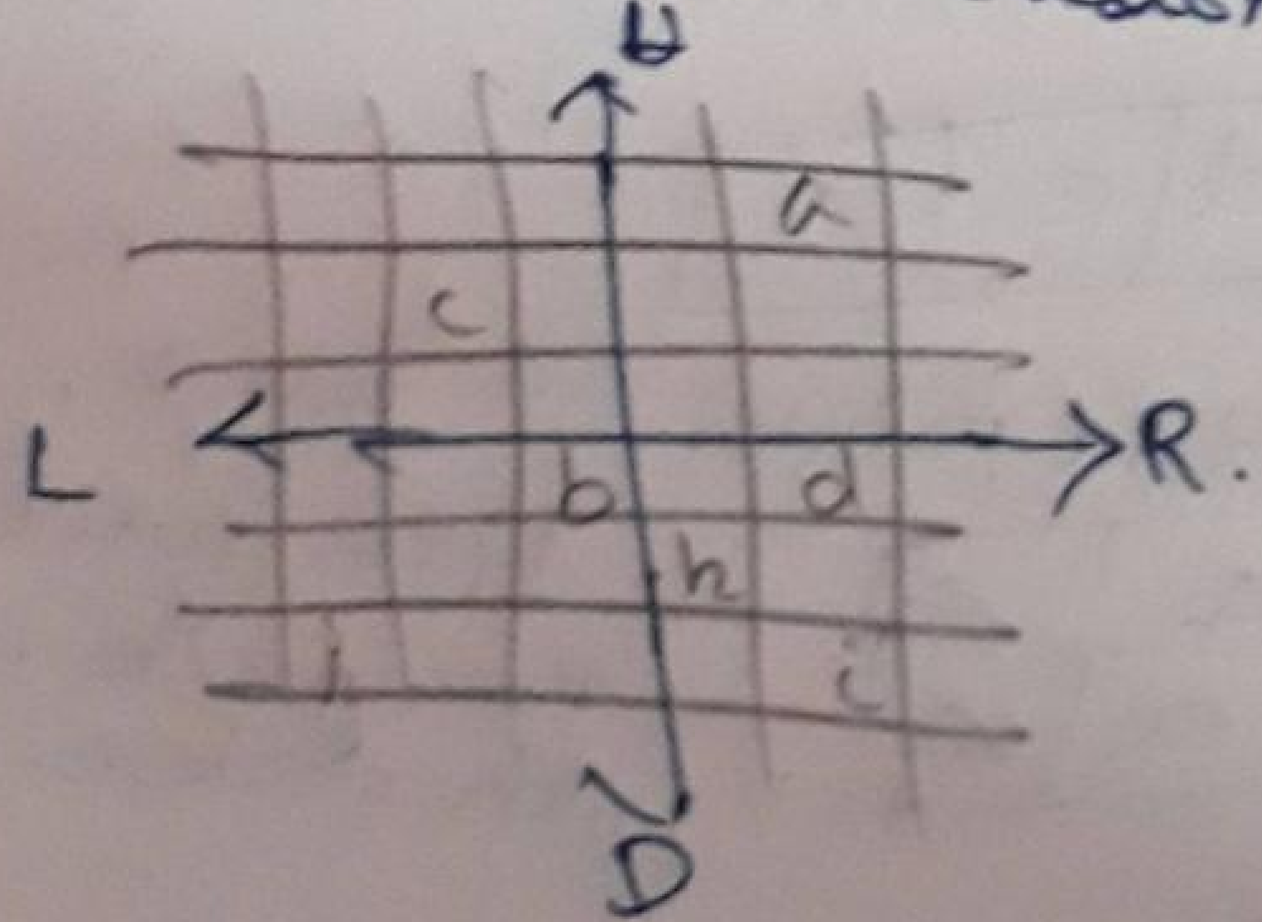


$\delta(q_1, (a, h)) = \delta(q_2, (x, y), (L, R))$



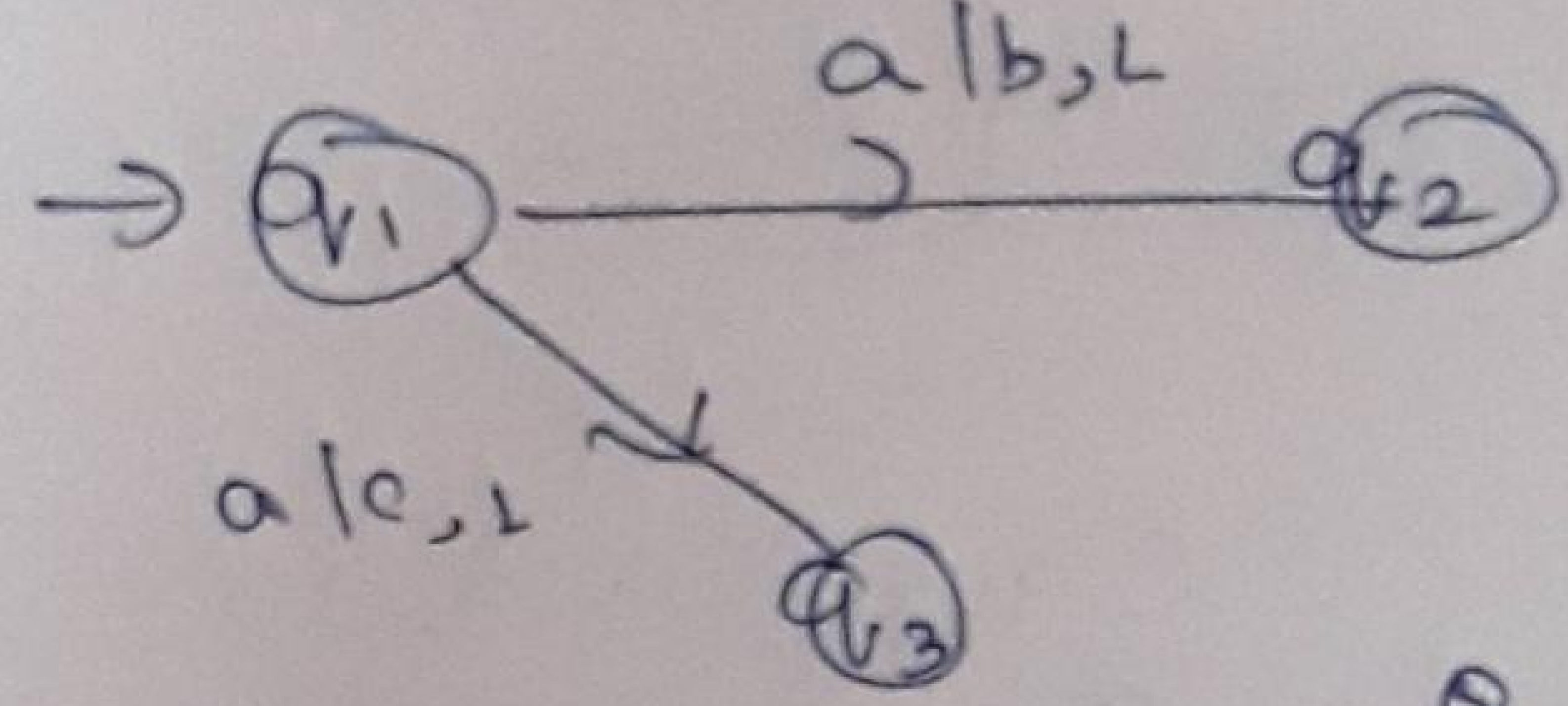
For each input, L & R can be changed.

④ multi dimensional T.M:



$\delta(Q \times \Gamma) \Rightarrow Q \times \Gamma^* \times \{L, R, U, D\}$

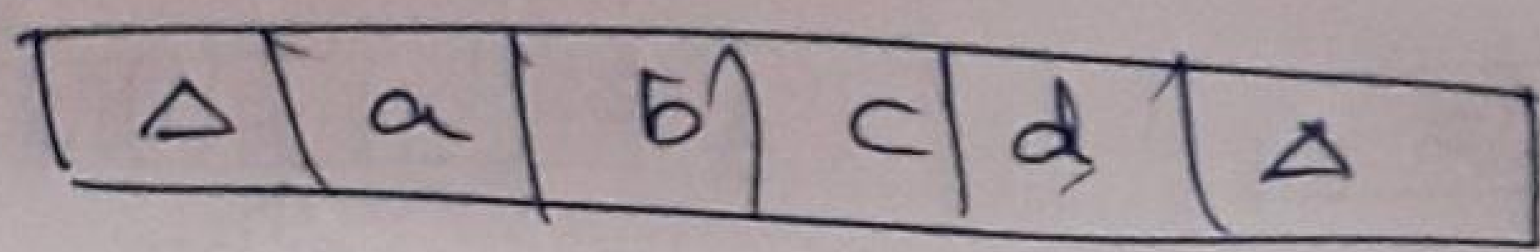
⑤ Non deterministic TM:



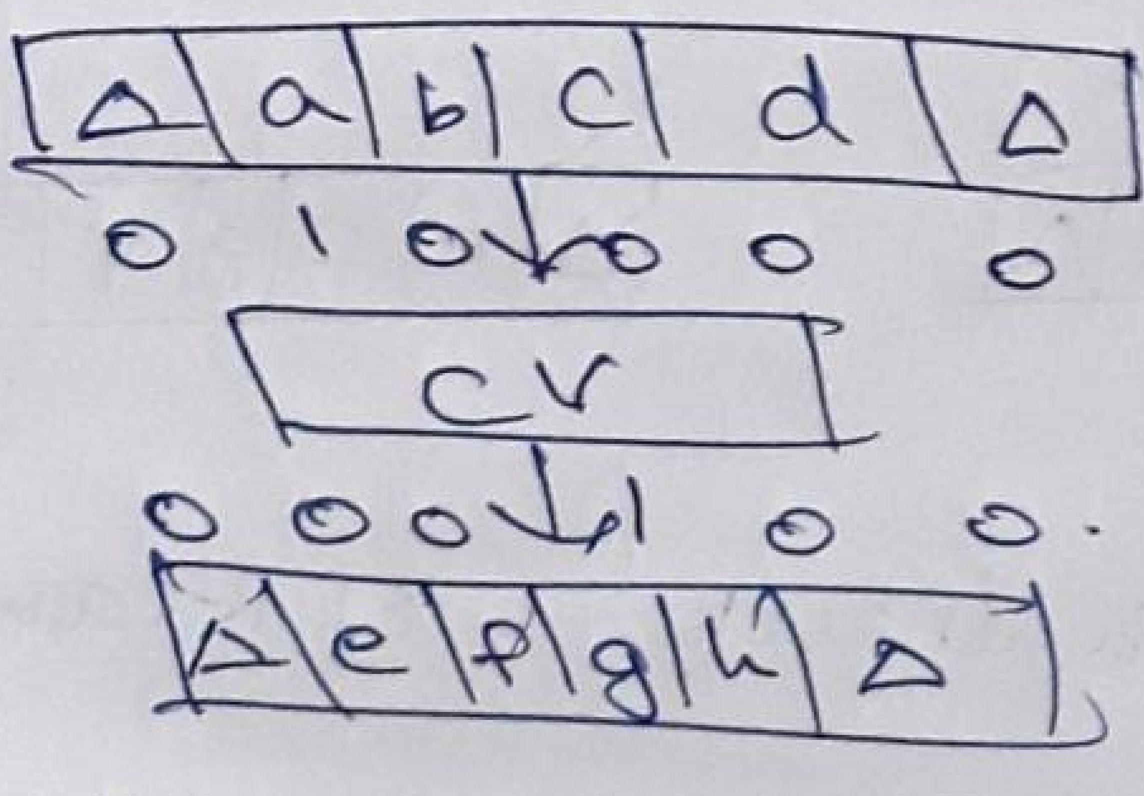
$$\delta(Q * \Gamma) \rightarrow 2^{Q} * \Gamma^{*} \times \{L, R\}$$

$$\boxed{DTM = NDTM.}$$

⑥ Offline TM:

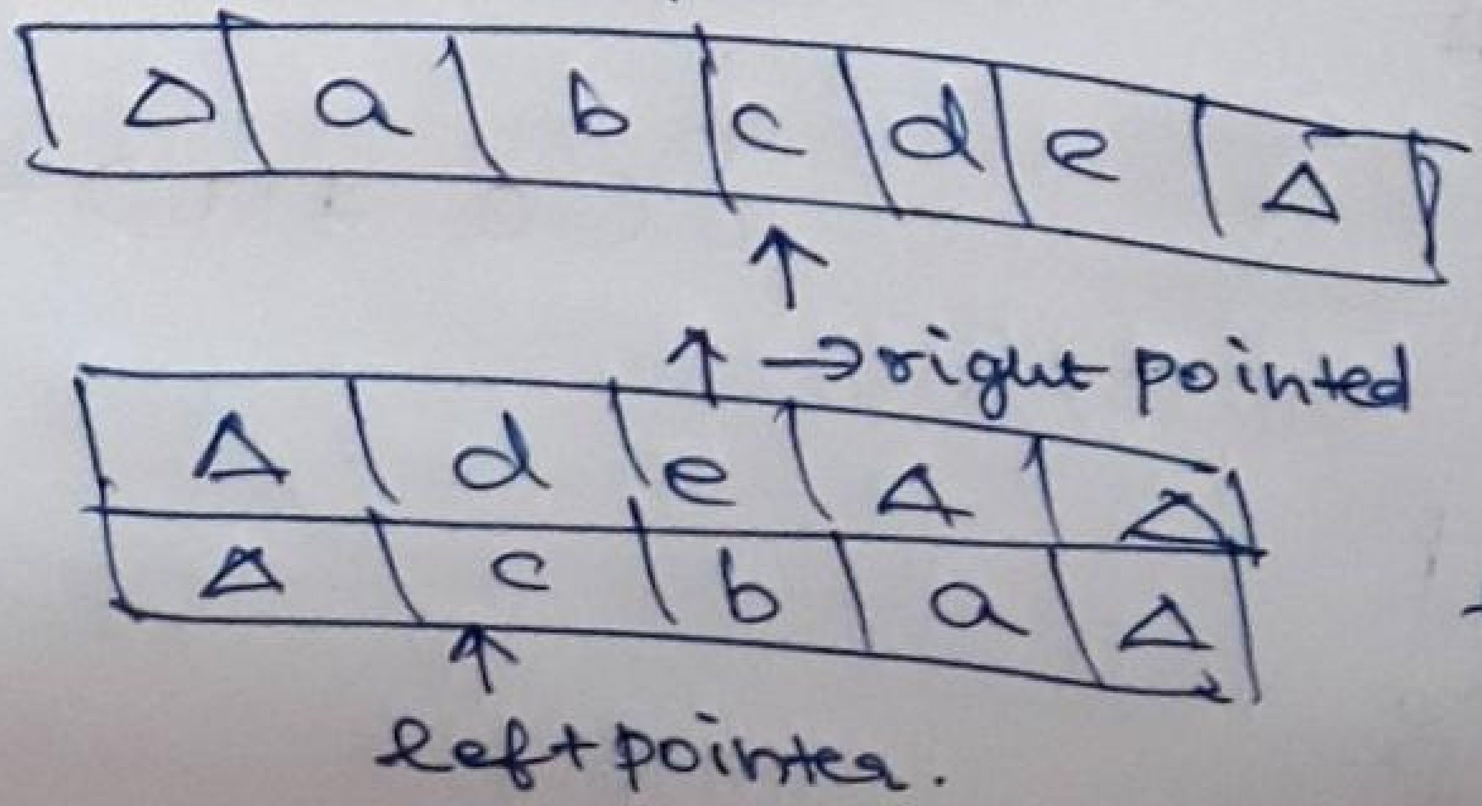


* It will pass the tape thru the next tape, this tape will take a copy of the next tape.



No change ⇒ 0
Change ⇒ 1

⑦. Semi infinite tape:



÷ by 2.
first write 2nd half
then write 1st half
in order

right pointer → starts from middle
left pointer → starts from start

Recursive language:

1) Recursive ennumerable laugle.
2) Recursive language.

1) The language is recursively enumerable if some turing machine accepts it.

if $w \in L$, T.M M halts @ a final state.
if $w \notin L$, ... is not @ final State / is in loop.

2) A language is recursive if some T.M M accepts it & halts on any input string (or) a languarge is recursive if there is membership algoritam for it (eg: CYK algo).

Scanned by TapScanner