eg. $L = \{ a^i b^j c^k \mid i*j = k \text{ and } i, j, k \geq 1 \}$

$a^2 b^3 c^6$ ✓ (aabbb cccccc)

aabbb cccccc ✗

a rough iteration

f a b b b c c c c c c

d e

f a b b b c c c c c c

d d e

f a b b b c c c c c c

d d e e

f a b b b c c c c c c

d d e e

f d d d e e e

f a b b b c c c c c c

Now all b's are over so we will give back b its own identity. (striking d's and putting b) [I haven't shown the method, but logic is given]

f a b b b e e e c c c

encounters f, then run again same iteration as above for second a

so final string ffdddeeeeee

For a 'a', we are striking c's equal to b's count.

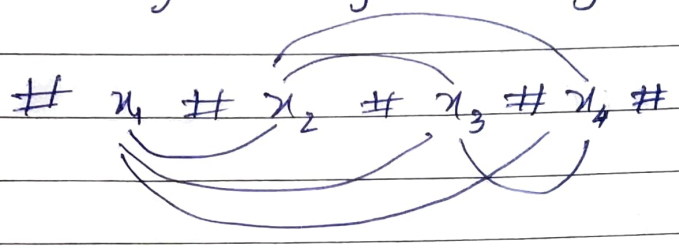
$2 * 3 \Rightarrow (3+3)$

eg. $L = \{ \# x_1 \# x_2 \# \ldots \# x_k \# \mid x_i \in \{0,1\}^* \text{ and } x_i \neq x_j \text{ for } i \neq j \}$

(exclude boundary cases)

$\# 1100 \# 0011 \# 0100$ ✓

we need to make ${}^nC_2$ comparisions for checking $x_i \neq x_j$ for $i \neq j$



$\$ x_1 \# x_2 \# x_3 \$ x_4 \#$ → this means comparing x_4 to x_4 (string after $)

$\$ x_1 \$ x_2 \# x_3 \# x_4 \#$ comparing strings x_1, x_2

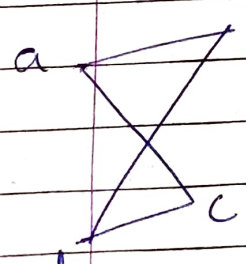
by moving back and forth compare each alphabet in string.

after this iteration (if x_1 and x_2 are unequal, else they are rejected)

$\$ x_1 \# x_2 \$ x_3 \# x_4 \#$

and further such iterations.

eg. $L = \{ G \mid$ G is a connected graph. ~~such that each vertex lies in a connected component~~ $\}$



(a,b,c,d) {(a,b) (b,d) (c,d) (a,c)

We start with A for a vertex and in an iteration if a vertex belongs to same connected comp. then we will replace it with capital alphabet.

(A, b, c, d) ((a, b), (b, d), (c, d), (a, c))

→ b encountered
come back, capitalise b.

(A, B, c, d) ((a, b) (b, d) (c, d) (a, c))

→ go forward ↓ got c

capitalise c

(A, B, C, d) (_____)

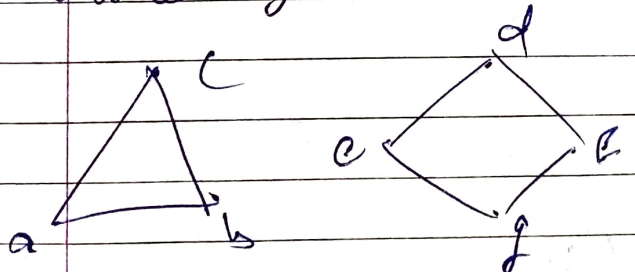
all vertices over.

Now we will see edges in connection with b. (as it is capital).

(b, d) is present so d is made B.

All a, b, c, d are capital, so they are in a same connected component.

another eg. on same L



(a, b, c, d, e, f, g) ((a, b) (b, c) (c, a) (d, e) (d, f) (g, e) (g, f))

Page No.: YOUVA
Date:

Now starting from A.

(A, b, c, d, ...) (a, b) (b, c), ...)

first edge includes a, opposite end is b, so we capitalise B.

→ indicates edge counted

(A, B, c, d, ...) (a, b) (b, c) (c, a) (d, e) (d, f) (g, e) (g, f)

then we get ~~a~~ edge ~~a, e~~ (c, a)

(A, B, C, d, e, f, g) ((a, b) (b, c) (c, a) (d, e) ...)

all ~~edge~~ edges ~~of~~ connected with ~~f~~ a are over, we will pick next capital element. (remember it was connected to a).

We look similarly for B now.

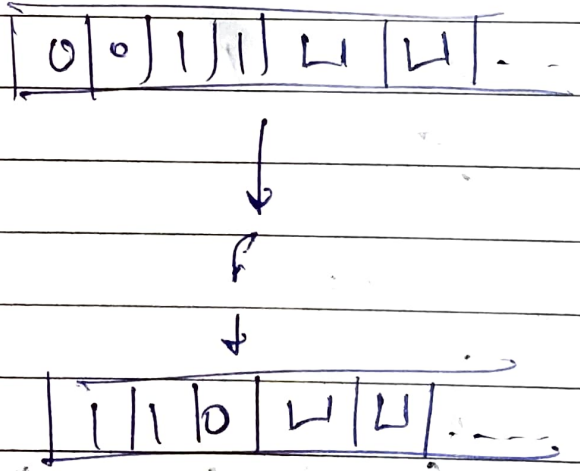
(A, B, C, d, e, f, g) ((a, b) (b, c) (c, a) (d, e) (d, f) (g, e), (g, f))

After all capital letters are done with similar process, we check if ~~the vertex~~ all the vertices are capital or not.
Here they aren't, so the graph is not a connected.

If in middle step we get all vertices capitalised then no need to repeat process, we can say graph is connected.

Apart from accepting languages, turing machines can also compute functions.

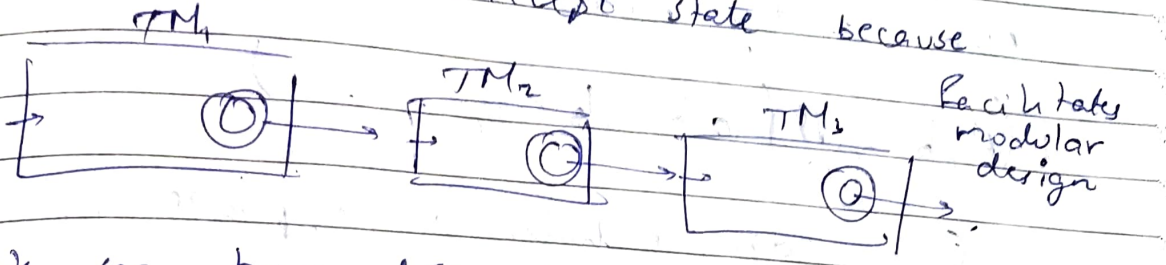
eg.



$f(0011) = 110$

There is only one accept state.

We have only one accept state because



Facilitates modular design

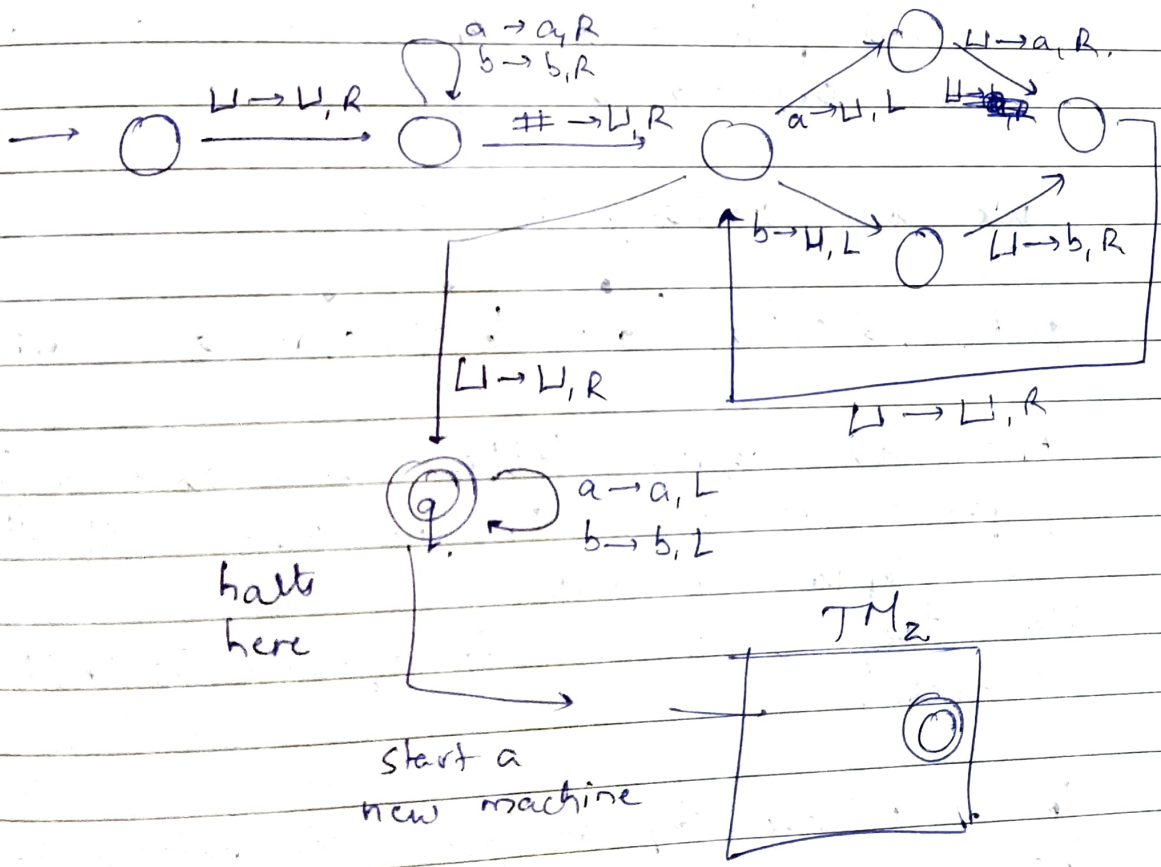
We can have different Turing machines in a chain.

eg. String concatenation.

␣aab#baab␣␣␣...

↓

␣aab␣baab␣␣␣



$f(w) = w'$ w is a +ve int, w is in unary representation

$w' = 3w$ w' is also unary representation

$w = |||$

unary systems is like using tally marks according to convention used in this question

unary → | => 0 ← decimal

|| = 1

||| = 2

|||| = 3

⋮ and so on

If $w = |||$ $w' = |||||||$

Initial tape |1|1|2|␣|␣|␣|...

place # in place of last 1

copy first bunch ||#||

place # ||#||#

copy the bunch again ||#||#||

remove all #s ||#|||| → |||| |||

place a 1 at last → |||| |||| |||

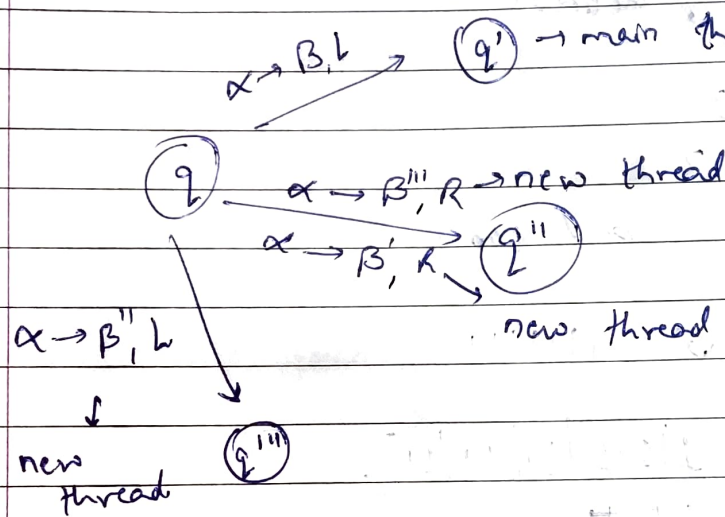
We must be able to make TMs, for each of these tasks.

NTM → Non-deterministic Turing Machine

$\delta: Q \times T \to \mathcal{P}(Q \times T \times \{L, R\})$

$(q, \alpha) \to \{ (q', \beta, L), (q'', \gamma, R) \ldots \}$

non-deterministic = explore all possible options, via spawning new threads



While we are making new threads, we need to remember / copy

- tape contents
- tape head location
- pointer to a state q'''', ... current state in this thread
- non-deterministic guess to take in that thread.

Rest thread related rules remain same as in NFA, NPDA

one thread accepted - rest all killed, ...

eg $L = \{$ positive integer r in unary $|$ r is composite$\}$

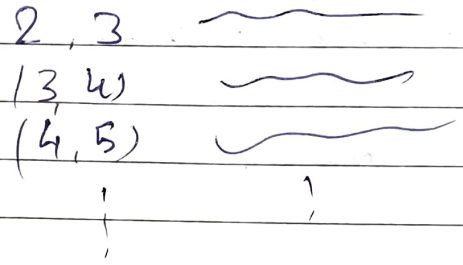
input is 1^{r+1} $\underbrace{1111\ldots1}_{(r+1) \text{ no. of } 1's}$

1) non-deterministically guess p and q.

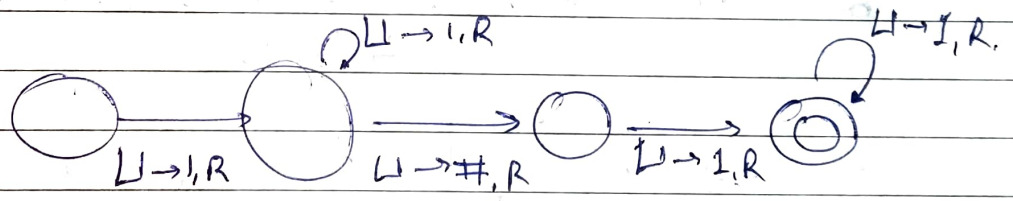
2) $p \neq r$, $q \neq r$

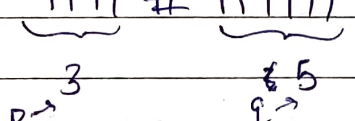
3) if $r = p \cdot q$ then accept, otherwise reject.

It like computing using each combination of p, q in each thread



2) 3) can be checked deterministically. but guessing p, q ~~requir~~ requires a non deterministic method





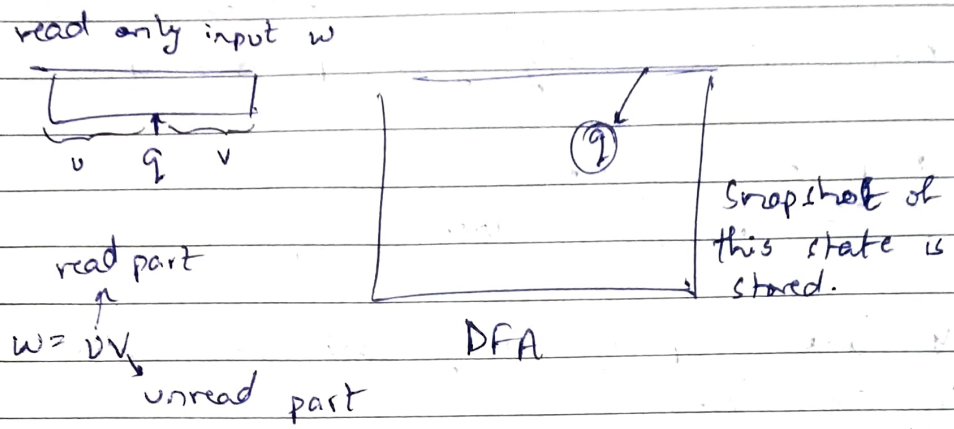
So this guesses p, q

further we can join DTMs to check condition 2), 3).

We can also add upper bound to p, q to limit thread formation

- ID of a machine a.k.a Configuration
  ↓
  instantaneous description

DFA
NFA
PDA
DTM
NTM



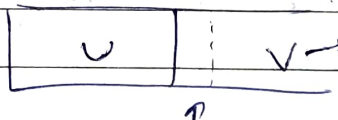
Suppose you go out, closing machine, and then come again and resume it, then it starts from where it left earlier.
For this you require few properties called instantaneous description

- Here for DFA, you need to remember q, v
- For PDA, q, v, γ → stack contents from top to bottom

If stack is  then γ = 1001

- For DTM
  entire tape contents, tape head, current state

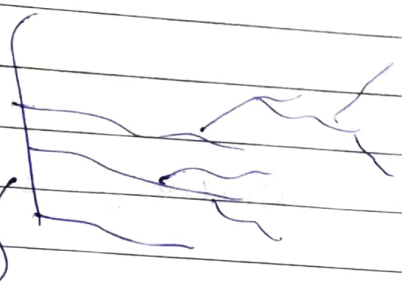
u, q, v, state
(similar to DFAs)



tape head.

- for NFA

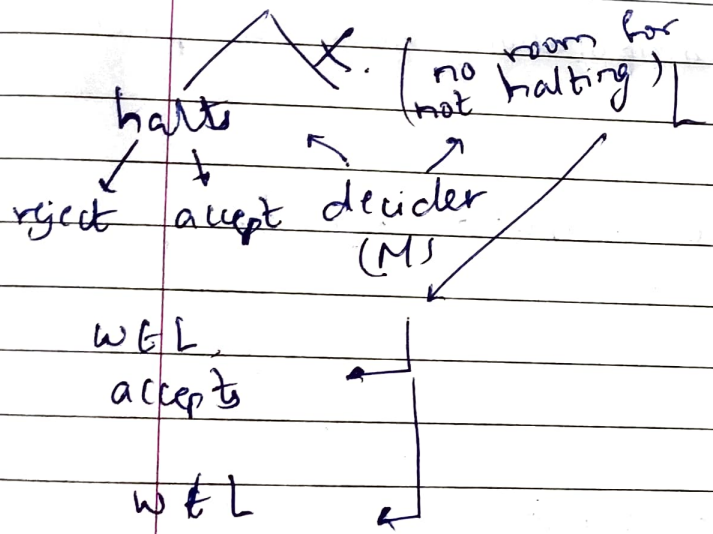
we have different threads.

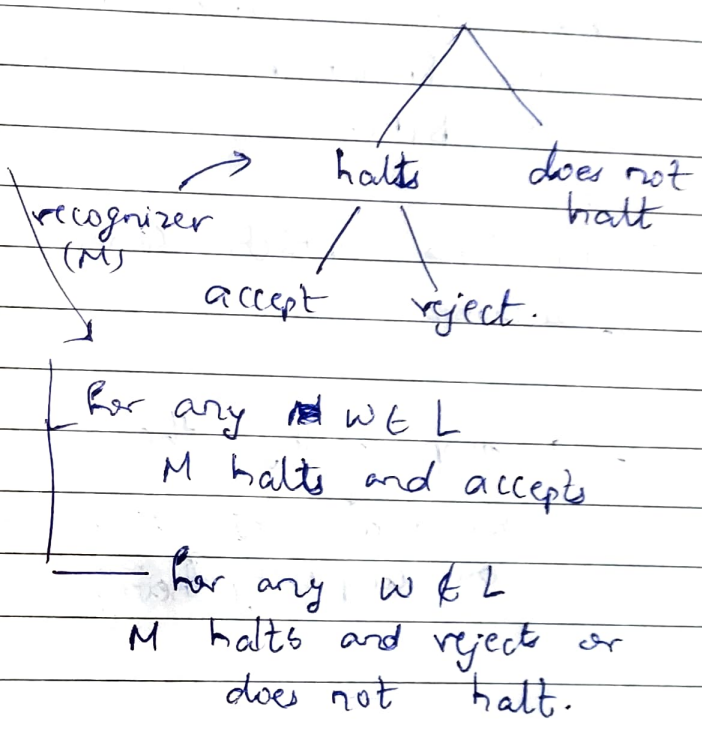


We will have a set of diff configuration of each thread.

$\{ v_1 q_1 v_1 , v_2 q_2 v_2 , \ldots \}$

, current states.





So according to this we can ~~can~~ compare ~~do~~ powers of different machines

has more power as, it also has cases of no halt.

DFA ≤ NFA ≤ PDA ≤ NTM

PDA has an extra Stack

want distinguish between DPDA, NPDA in this course.

for NTM, DTM
tapes can be thought as stacks.
we could push and pop elements in it.

So DPDA $\leq$ DTM $\leq$ NTM

What will come in next few lectures
These inequalities are intuitively correct but now we will show

DFA = NFA $<$ PDA $<$ DTM = NTM
This is called chomsky hierarchy

- Also give church turing thesis
  All known models of computation have a power of DTM.

- Functions which cannot be computed by turing machines

  → computability theory
  ↓
  based on this, define various notions on complexity theory