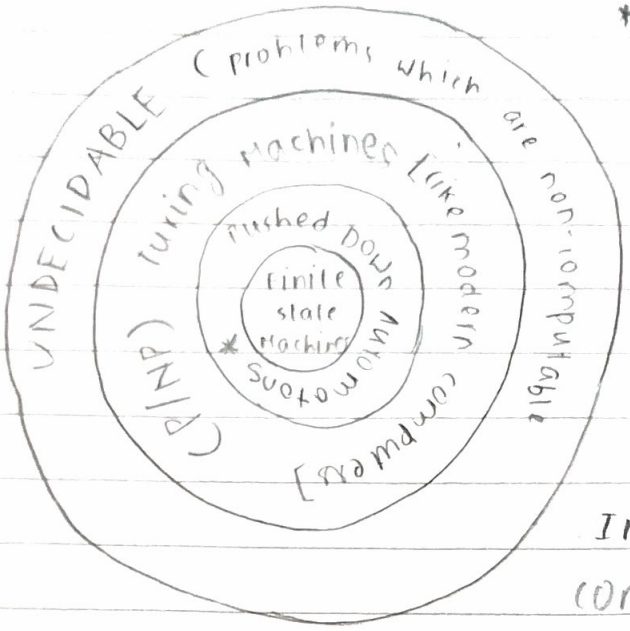
store
67



* context free languages [compilers]

P: deterministic polynomial

NP: Nor deterministic polynomial

Induction, contradiction, construction

Q1 If K is a positive integer, prove that K^2+K is even

P(1): $(1)^2+1 = 2$ — even

so we assume that P(K) is true. : $K^2+K = 2m$

Now we have to prove that P(K+1) is true

$(K+1)^2+(K+1) = K^2+1+2K+K+1 = (K^2+K)+2K+2$

$= 2m+2K+2$

which is also even. Hence by induction K^2+K is even

Q2 P.T K^3-K is divisible by 6 for $K \geq 1$

P(1) = $1^3-1 = 0$ divisible by 6

so we assume that P(K) : $K3-K = 6m$ for all $k \leq K$

To prove that P(K+1) is true —

$(K+1)^3-K-1 = K^3+1+3K^2+3K-K = (K^3-K)+3K^2+3K+1-1$

$= 6m+3K^2+3K+1-1$

$P(n): 3n^2 + 3n$

$P(1): 3+3=6$ — divisible by 6

Now assume that $P(n)$ is true. So $3n^2 + 3n = 6l$

To prove that $P(n+1)$ is also true —

$$P(n+1) = 3(n+1)^2 + (3)(n+1)$$
$$= 3n^2 + 3 + 6n + 3n + 3$$
$$= (3n^2 + 3n) + 6n + 6$$
$$= 6l + 6n + 6$$

$P(K+1): 6m + 3K^2 + 3K = 6m + 6l + 6n + 6$

$= 6(m + l + n + 1)$

Hence proved.

Q3. If K is a square, then P.T $K = 0 \pmod 4$ or $1 \pmod 4$

Q4. If a number has odd number of factors, what can you say about the number (square)

A3. P(1) (when $K=1$) $1 \neq 0$

Prove for even and odd

$K^2 = 4q + 0$ $q \in N$

$K^2 = 4q + 1$

Assume $q \in N$ and $K^2 = 4q$ is true (remainder = 0)

Now to prove $q+1$ is true

$K^2 = 4(q+1)$

store 67

12. $K^3-K = K(K^2-1) = K(K-1)(K+1)$

Assuming K is a multiple of 3 $\rightarrow$ $(K-1)\,3m\,(K+1)$ (3m: odd)

Then either K-1 or K+1 will be even

$3m \times 2n\,(K-1)$ or $3m \times 2n\,(K+1)$ $= 6mn\,(K-1)$ / $6mn(K+1)$

Base case : $P(1) : (1)^3 - 1 = 0$

Induction hypothesis : true for all $K \leq m \rightarrow 6 | (m^3-m)$

Extension : $6 | ((m+1)^3 - (m+1))$

$m^3 + 3m^2 + 3m + 1 - m - 1 = m^3 + 3m^2 + 2m$

$= m^3 - m + 3(m^2+m) = 6l + 6n = 6(l+n)$

Examples used to disprove - counter examples

Q. Prove that the set of all prime numbers is infinite

$P = \{2, 3, 5, 7 \ldots\} = \{p_1, p_2, p_3 \ldots p_n\}$

Suppose P is finite, then p_n is the largest prime

Let $K = p_1 p_2 p_3 \cdots p_n + 1$ and $K \geq p_n$

$\frac{K}{p_i} = 1 \pmod{p_i}$ where $1 \leq i \leq n$

Two consequtive numbers are coprimes

since $K \geq p_n$, p_n is not the largest prime.

So the set P is infinite

$\neg$ - not $\Rightarrow$ implies

$\wedge$ - and

$\vee$ - or

P	Q	P ⇒ Q
0	0	1
0	1	1
1	0	0
1	1	1

$!(P \wedge Q) = !P \vee !Q$

$!(P \Rightarrow Q) = !(!P \vee Q) = P \wedge !Q$

$!(\exists) = \forall$

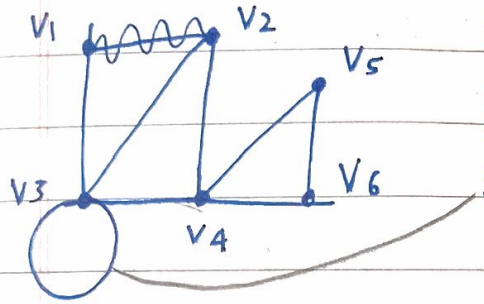
$!(\forall) = \exists$

GRAPHS - To define binary relationships

$G = (V, E)$ - set of vertices & edges

$E \subseteq V \times V$

$E_1 \subseteq V_1 \times V_2$



undirected

Directed

self loop

To go from V_1 to V_4

$V_1V_2V_4$, $V_1V_3V_4$

$V_1V_2V_3V_4$, $V_1V_3V_2V_4$

walks

$V_1V_2V_3V_4V_1$ closed path with first and last vertex as same - cycle

Closed walk: $V_1V_3V_2V_1V_3V_4V_2V_1$ - circuit

A graph with no self loops / multi edges — simple

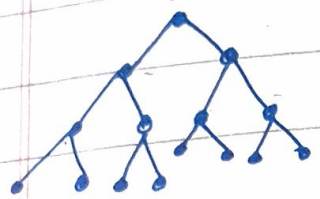
Maximum no. of edges = ${}^nC_2$

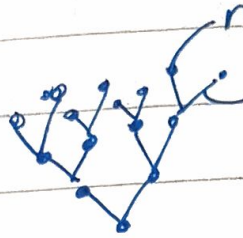
Q In a tennis tournament there are 128 players. How many matches occur in it?

Ans: 127 (∵ every player plays a match after which one is eliminated in a round

$64 + 32 + 16 + 8 + 4 + 2 + 1 = 127$

TREES



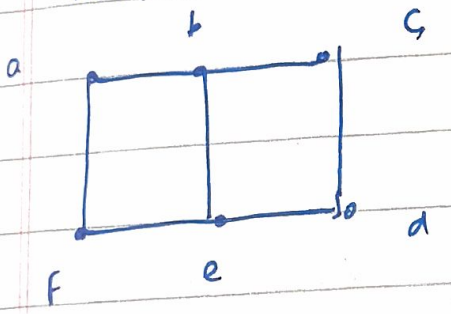


nodes

leaves

Binary tree - only 2 nodes are produc from each node

Degree of a vertex : No. of edges incident on a vertex



degree (a) = 2 deg(a)

degree (b) = 3 deg(b)

$\max \deg(x) \quad x \in V = \delta$ (3)

$\min \deg(x) \quad x \in V = \Delta$ (2)

$\sum_{x \in V} \deg(x) = \text{Even}$

Every edge contributes to 2 vertices

$$\sum_{x \in V} \deg(x) = 2|E|$$

$|E|$ — no. of edges

HANDSHAKE LEMMA

Q. Degree of every vertex = 3
n vertices & m edges
Is n odd or even?

$$\sum_{x \in V} deg(x) = 3n = 2m \Rightarrow n = \frac{2m}{3} \Rightarrow \frac{3n}{2} = m$$

since m represents no. of edges it has to be a natural number. For this n should be divisible by 2
∴ n is even

Q. You have a graph G with n vertices & m edges. can the degrees of the n vertices be distinct

max (deg) <= n-1
min (deg) >=0
You have n vertices - if they have different degrees it should be between 0 to (n-1)
But if the degree of a vertex is (n-1), degree of no vertex can be 0 (i.e - it connects all other vertices)
min (deg) >=1
NOW degrees should be b/w 1 - n-1
= n-1 ✗ +1 = n-1 degrees for n vertices.
so they are not distict
FALSE

Assume a pigeon hole can have atmost m/n pigeons

$\left(\frac{m}{n}-1\right)m > \frac{m}{n}\cdot n$

m_{almost} $\quad$ n_{actual}

contradiction

a pigeon hole can have atleast $\frac{m}{n}$ pigeons (store)

since $m>n \Rightarrow m/n > 1$ and m/n is an int : $m/n \geq 2$

PIGEON HOLE PRINCIPLE

If there are m pigeons and n pigeon holes & $m>n$ then $\exists$ a pigeon hole containing at least 2 pigeons

If $\min(\deg x) = 0 \Rightarrow \max(\deg x) \leq n-2$ (n-1 values)

Q n vertices — n1 vertices have even degrees (even)
n2 vertices have odd degrees (odd)

∴ no. of vertices of odd degrees = even

$\sum_{x \in n} \deg(x) = \text{even}$

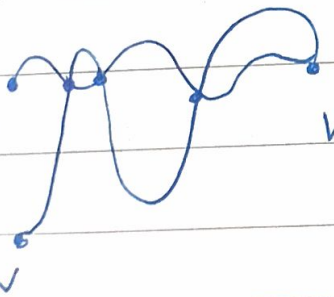
Q u & v are connected if the is a path b/w them

$(u,v) \sim$ is an equivalence relation. P.T

u is connected to u $u \sim u$ — REFLEXIVE

$u \sim v \Rightarrow v \sim u$ - SYMMETRIC

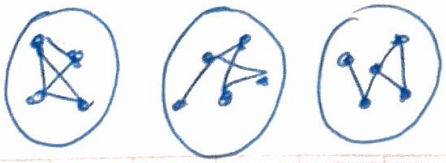
$u \sim v$ & $v \sim w \Rightarrow u \sim w$ - TRANSITIVE



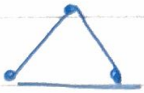
∴ Connectedness is an equivalence relation

Equivalence classes — components in the graph that are not connected to each other (partition of the original set)

Ex: Odd nos. & even nos. in $\mathbb{N}$



- All vertices in the same component are connected
- Cycle : closed path

 4 cycle  Triangle / 3 cycle

TREE : "A tree is a connected acyclic graph"

Q. You have a tree with n vertices. Find the no. of edges possible.

(Unproved statements : conjecture)

-  n vertices ⟶ (n-1) edges

Q. P.T a tree on 'n' vertices has '(n-1)' edges

Every node except the 1st node has a corresponding edge so n nodes give (n-1) edges

When n=1 En = 0

Consider a tree on k+1 vertices has k edges

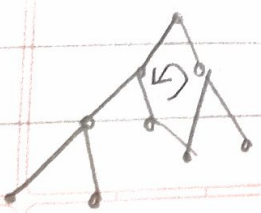
$T' = T \cup v$

$|V(T')| = k$

$|E(T')| = k-1$

adding doesn't work for graphs only removing works

If you add more than 1 branch - it is no longer acyclic



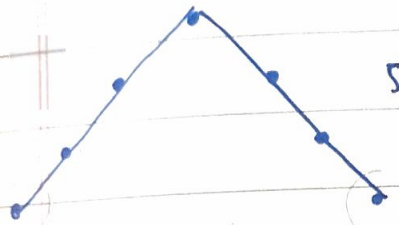
If you have multiple trees in a graph — forest

Q1) A graph has K forests find the no. of edges

Q2) A binary tree has L leaves and I internal nodes
Can you bound (upper) I as a function of L?

$|I| \leq |L| - 1$ each ~~node~~ tree has at most 2 children

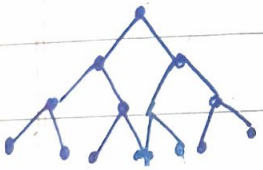
counter example



5 int. nodes & 2 leaves

change the definition: A tree has exactly 0/2 child

$I = L - 1$ or $I \leq L - 1$?



#1 Basic case: 1 nodes, 0 leaves

consider a tree on K leaves ~~edges~~ - assume $I = K - 1$

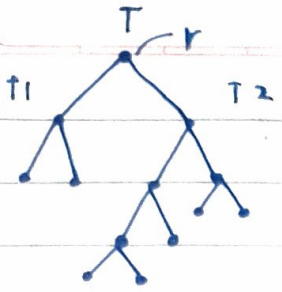
NOW consider a tree on K+1 leaves

to prove it works, remove one node

∴ Internal nodes $- = 1$

leaves ~~edges~~ $= -2 + 1$ $(- = 1)$

#2



$|I_1| = |L_1| - 1$

$|I_2| = |L_2| - 1$

$I = I_1 + I_2 + 1$ ($I = I_1 \cup I_2 \cup \{r\}$) } disjoint sets

$L = L_1 \cup L_2$

so $I_1 + I_2 + 1 = I$ —— ①

$L = L_1 + L_2$ —— ②

① $I = L_1 - 1 + L_2 - 1 + 1$

$I = L_1 + L_2 - 1$

$I = L - 1$ from ②

$|I| = |L| - 1$

#3 $\sum_{v \in V(T)} deg(u) = 2(n-1) = 2n - 2$ — Double counting method

$I_n = n - L$ — ① (L = no. of leaves)

root has degree 2

leaves have degree 1

all other int. nodes have degree 3

$2 + 1.L + 3(I-1) = 2n - 2$

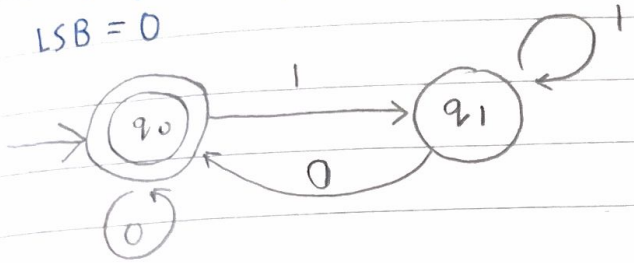
$2 + L + 3(I-1) = 2(I+L) - 2$ from ①

$2 + L + 3I - 3 = 2I + 2L - 2$

$I = L - 1$

Q. $\Sigma = \{0, 1\}$. construct a DFA which determines whether the string is divisible by 2. Assume an empty string is an even no.

LSB = 0



q_0 = even
q_1 = odd

State	0	1
→*q_0	q_0	q_1
q_1	q_0	q_1

$D = \{\{q_0, q_1\}, \{0, 1\}, \delta, q_0, \{q_0\}\}$

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

$\delta(q_1, 1) = q_1$

---

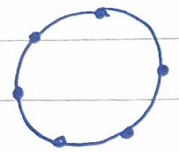
for the n^{th} component no. of vertices =

no. of edges $= v_n - 1$

$$\sum_{n=1}^{k} (v_n - 1) = V - k$$

For every $n >= 3$, ∃ a graph on n vertices such that Degree of every vertex is equal → graph is regular

$n >= 3$ & degree = 2



for $n >= 4$, degree = 3 (where n is even)

$n_1, n_2, n_{>3}$ $\quad n_{>3} \leq f(n_1, n_2)$ or $n_{>3} \leq f(n_1)$

vertices with degree = 1, 2 & 3

0 1 2 3

TUTORIALS

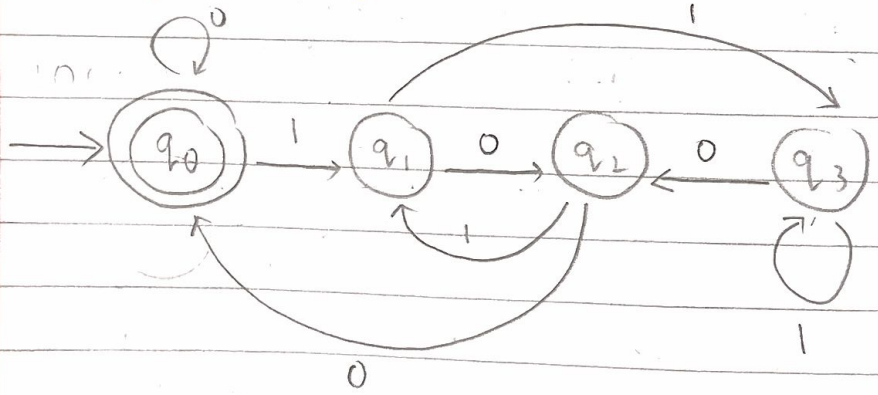
$\Sigma = \{0, 1\}$

#1. Number is divisible by 4 (00) should be at end

#2. Number is divisible by 3 (01)

#3. Strings having '101011' as substring

#1.



#2

#3

Q4.

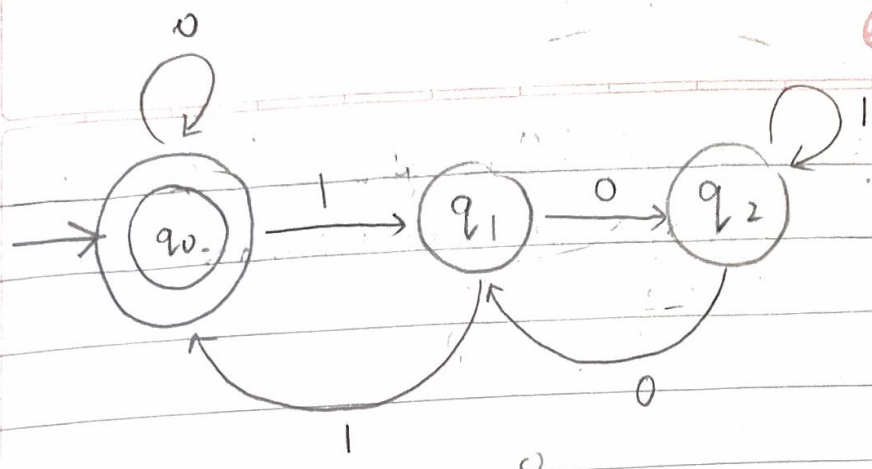
(ii)

(i)

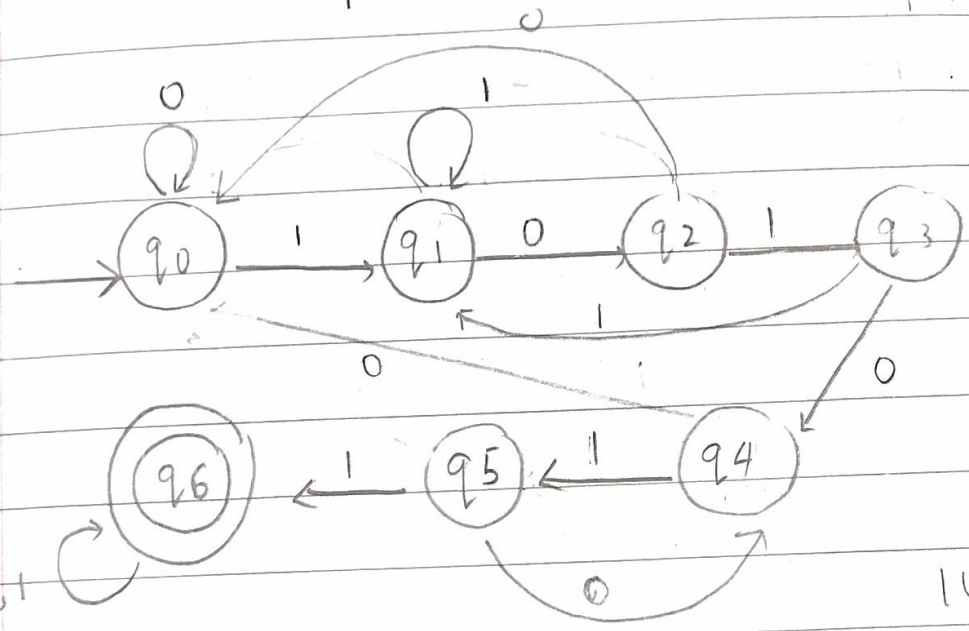
11
100
101
110

store
67

#2



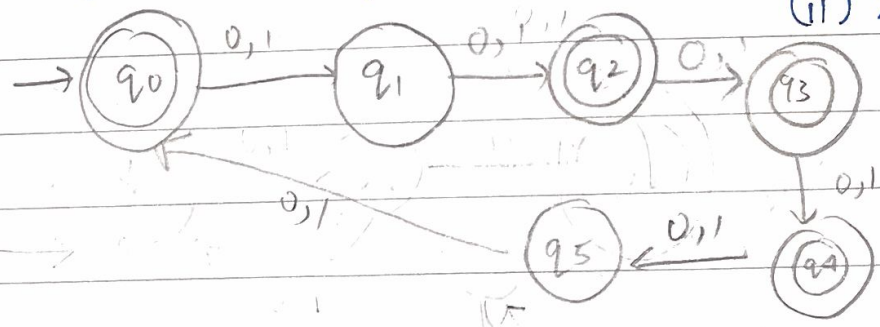
#3



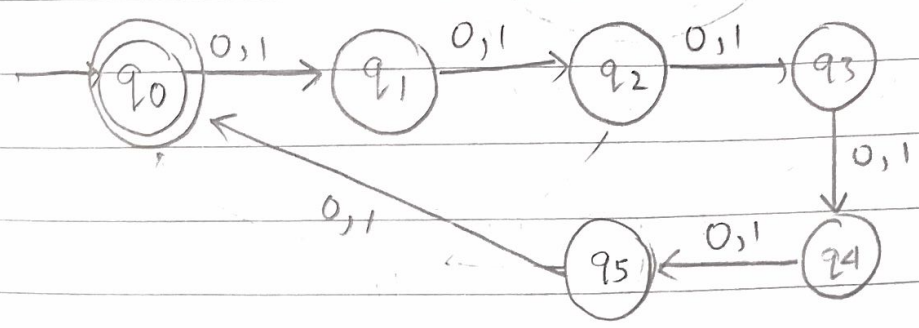
0101011
1101011
1110101
11001010
10111010
10100101011

Q4. Length of string is divisible by (i) 2 & 3 —
(ii) 2 Or 3

(ii)



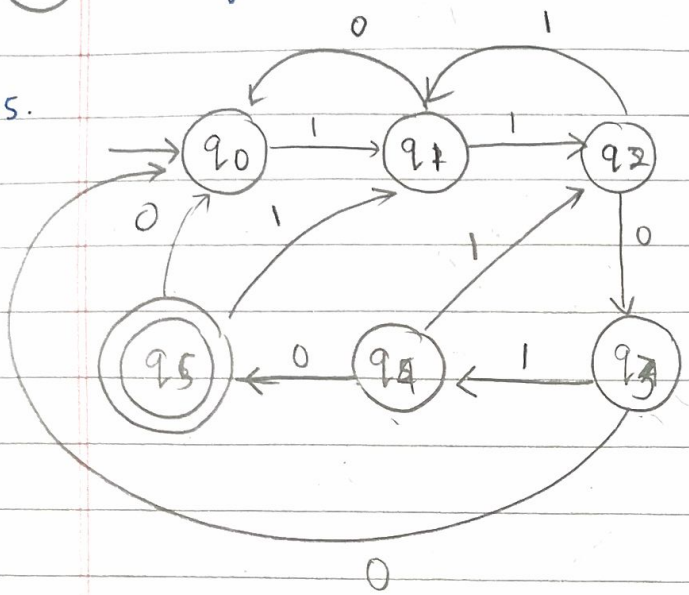
(i)



Q5. strings ending with '11010'

Q6. strings such that the 3rd bit from the end is a '1'

A5.



111

1100

11011

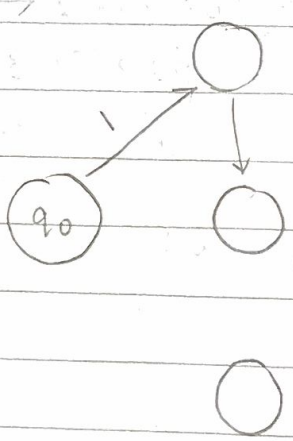
Valid

10100

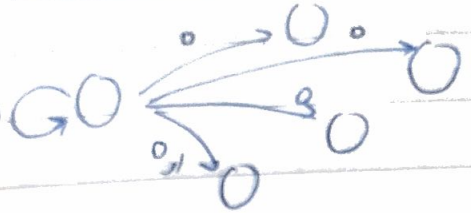
~~1001~~

~~1111~~

A6.



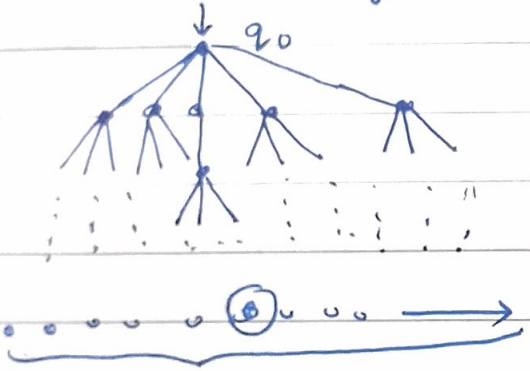
Non-deterministic finite automata



can be in any of the states or in many different states at a given time \ if next state is non-deterministic

If there is a path to reach a common final state then the string will be accepted

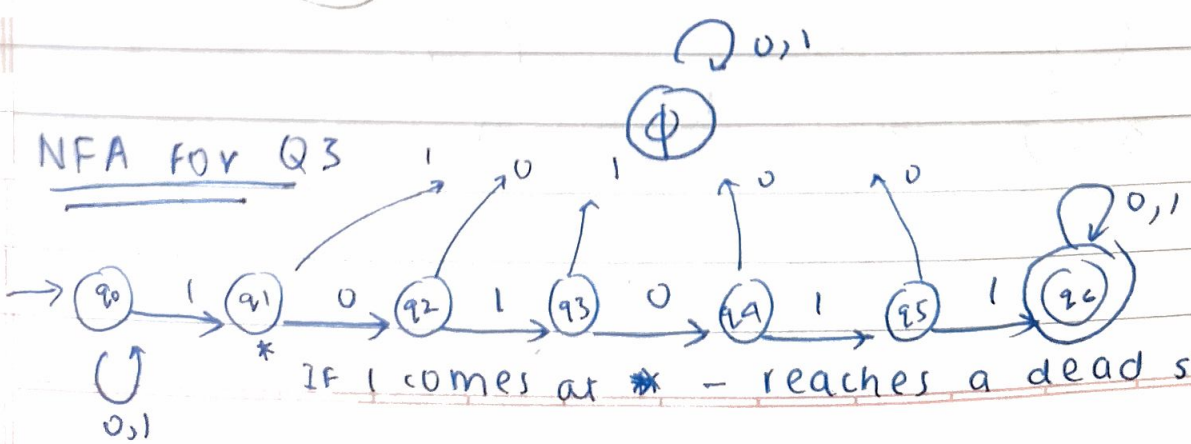
Accepts a string if there is a way of going from start state to one of the final states containing the input string



If any of them (K states) is a final state, string is accepted
Similar to a // computer (makes choices/guess)

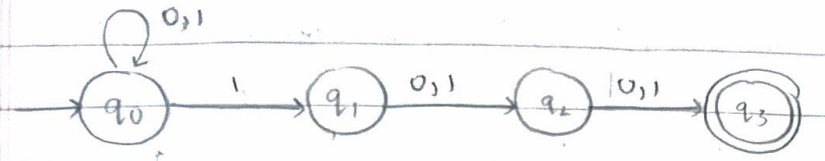
DFA ⊂ NFA (subset) — easier to construct

NFA for Q3



not x or y
not (x and y)

NFA for Q6.



Q1. L = set of all binary numbers divisible by 3 ✓

Q2. All strings where 3rd bit from the end is 1 ✓

Q3. strings having even no. of 0s or having exactly two 1s

Q4. {w | w has even length and odd number of 0s}

Q5. Set of all binary numbers not divisible by 3 ✓

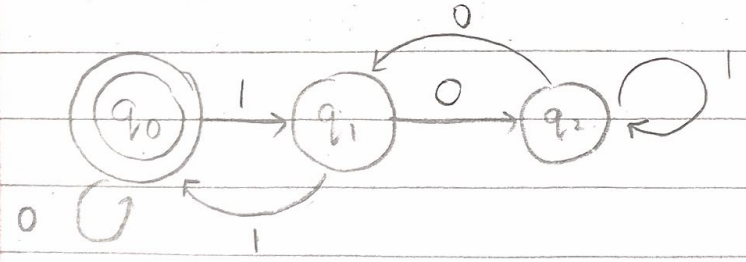
Q6. All strings where the 3rd bit from the end is not 1

Q7. {w | w does not contain '01' as substring} ✓

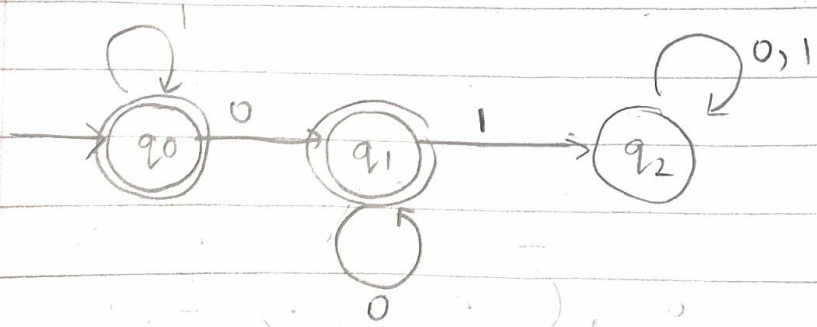
Q8. {w | w contains atleast 3 1s} ✓

10010100

A1

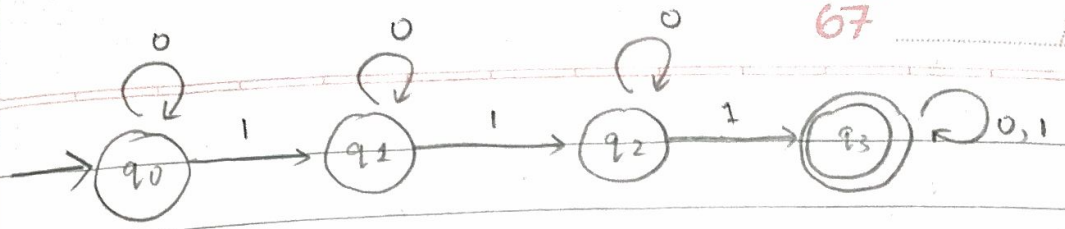


A7.

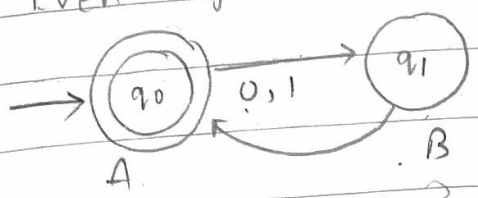


store
67

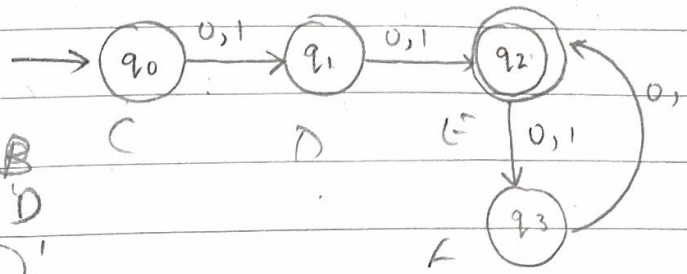
A8



A4. Even length

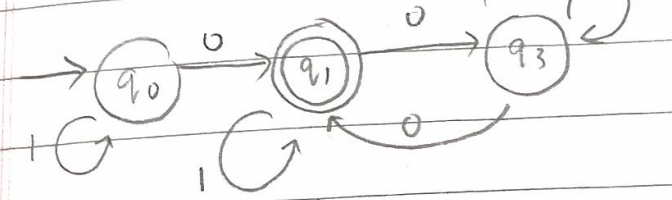


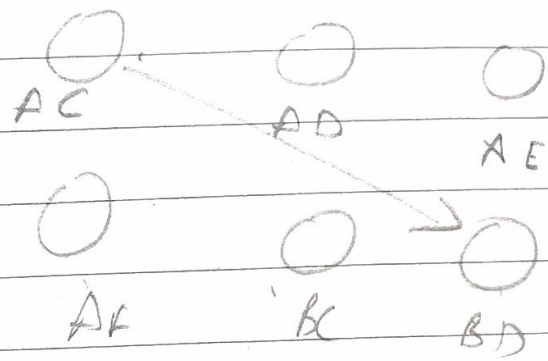
Even length apart from 0 length

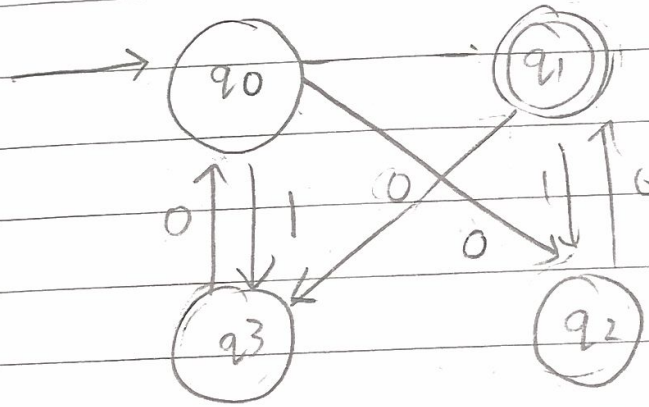


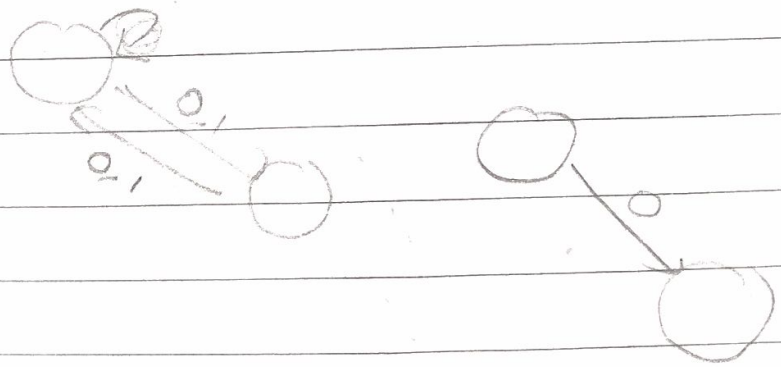
Odd no. of zeroes

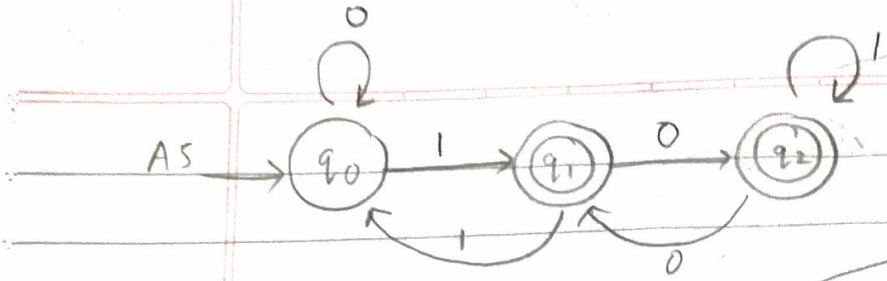
A → B
C → D







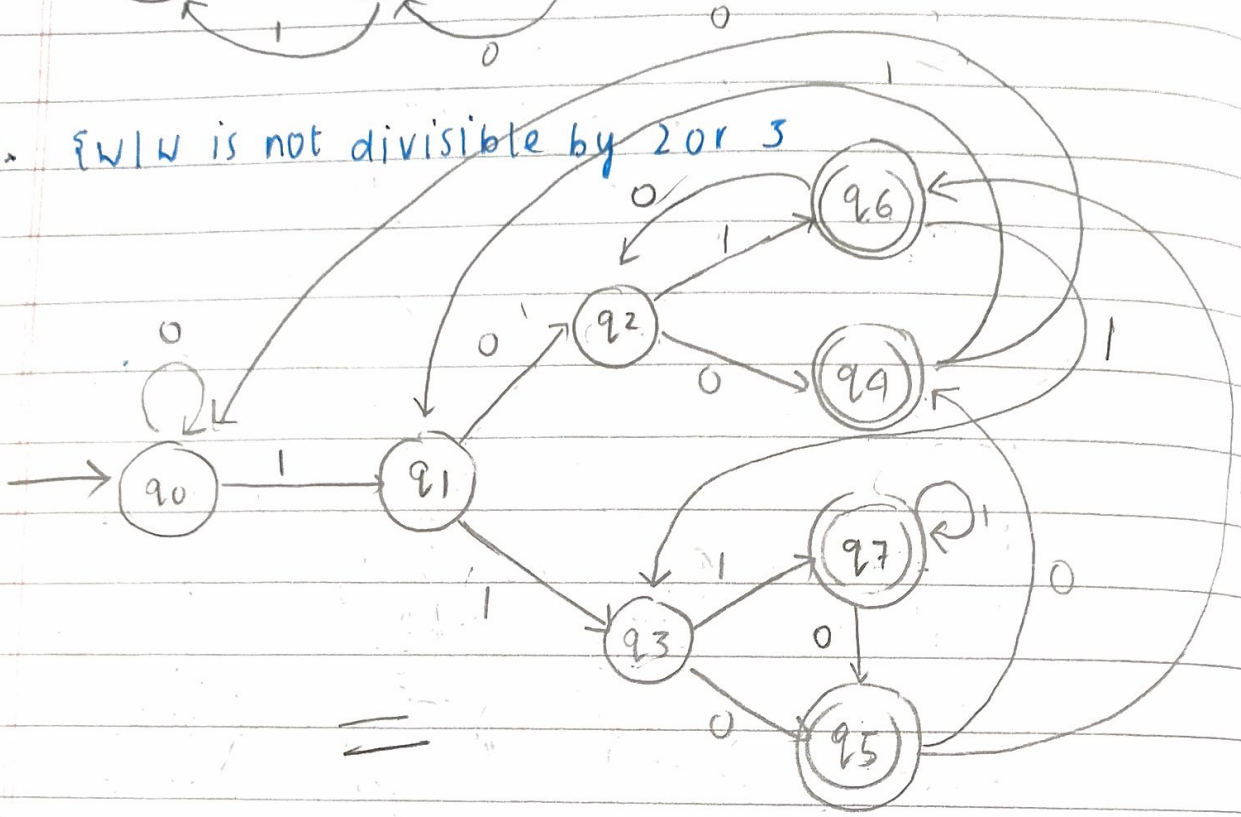




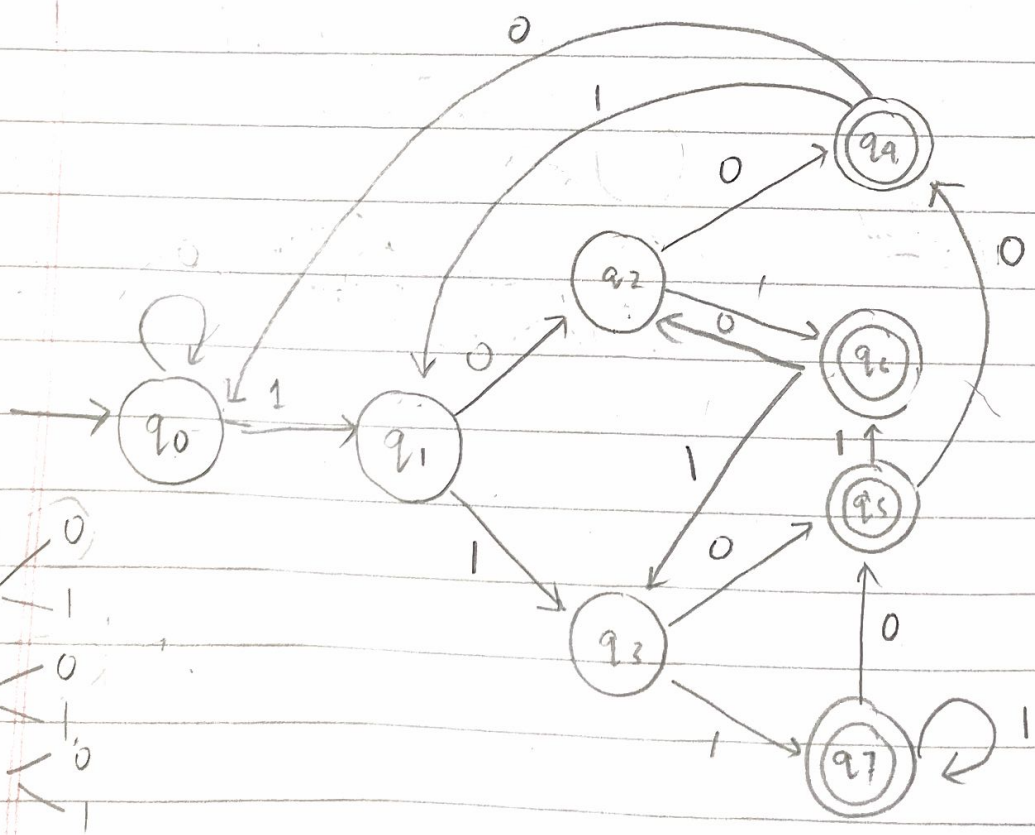
0101

Q9. $\{w \mid w$ is not divisible by 2 or 3

A2

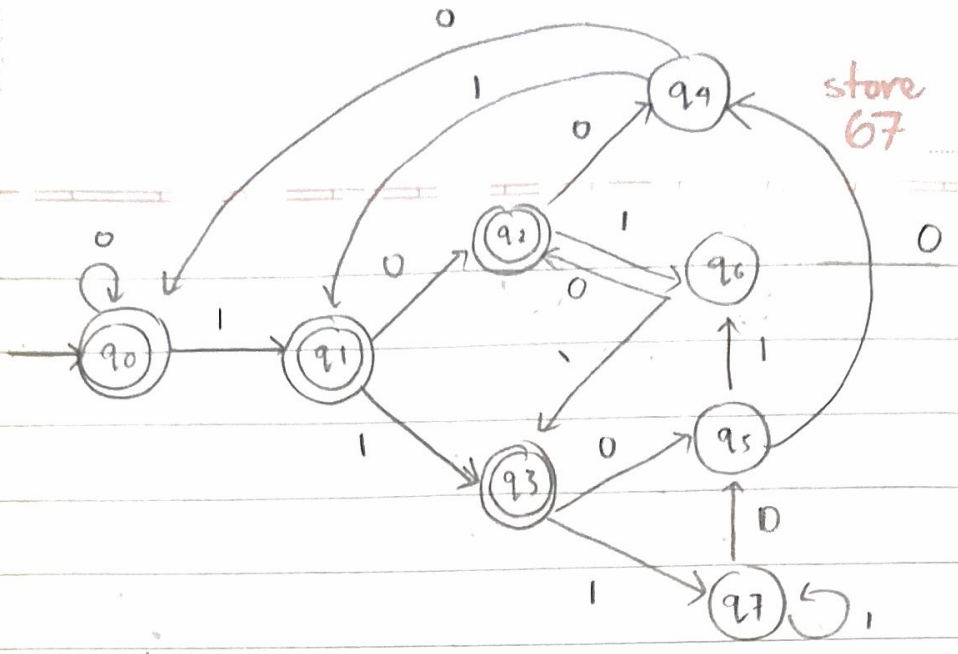


=

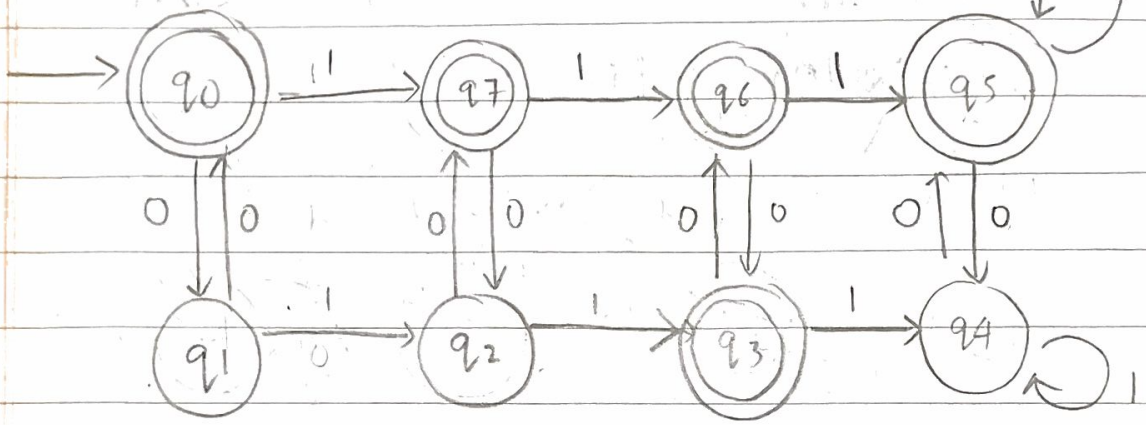


100
101
110
111

A6

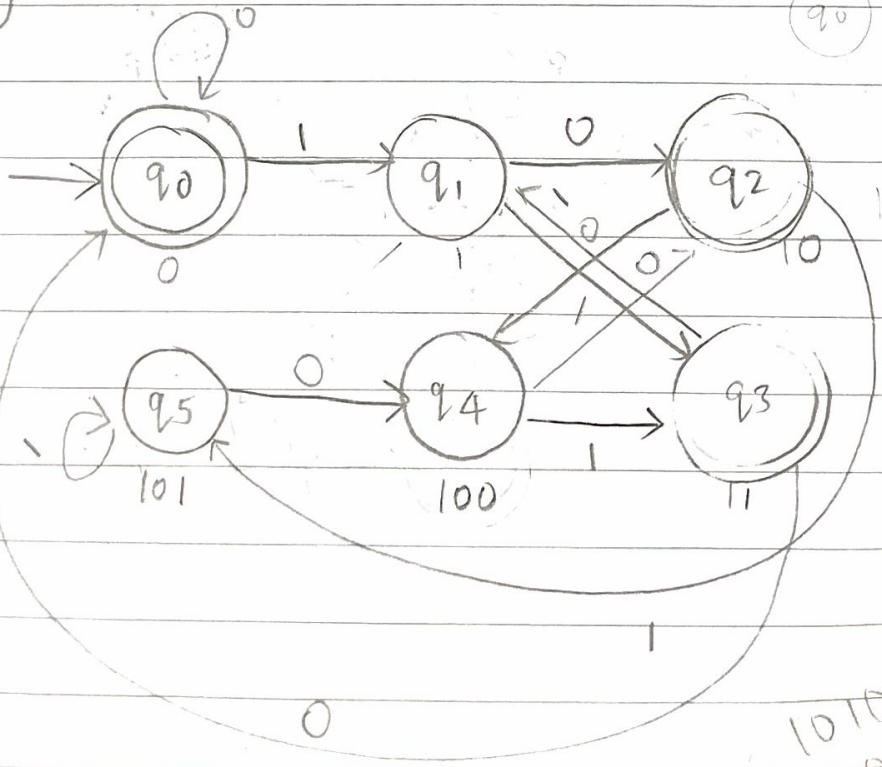


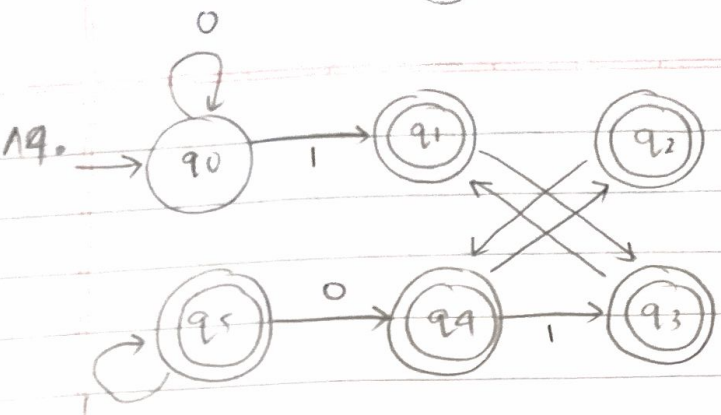
A3.



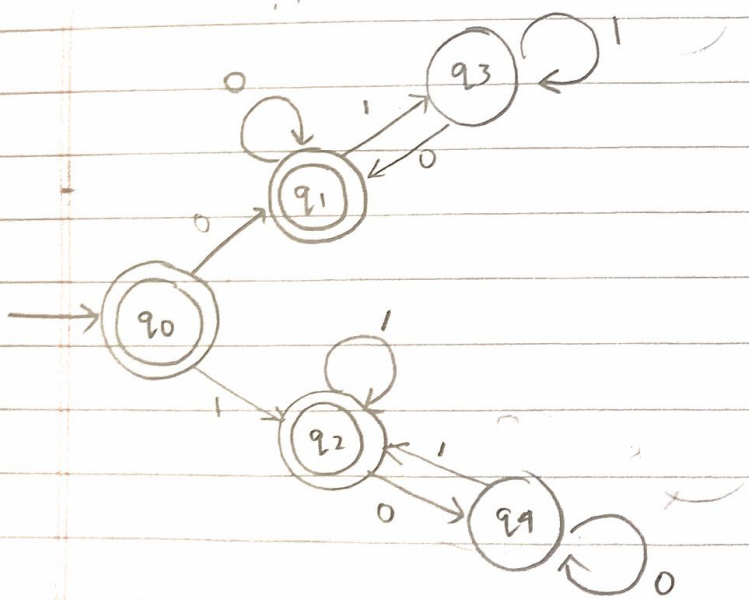
A9.

2 and 3 = 6





10 $\{w \mid w$ contains equal matches of occurences of substring 01 & 10

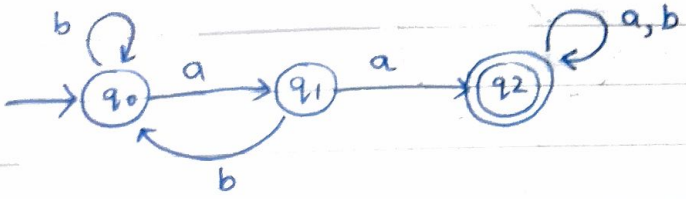


diff b/w the no. of 01s & 10s can never be > 1

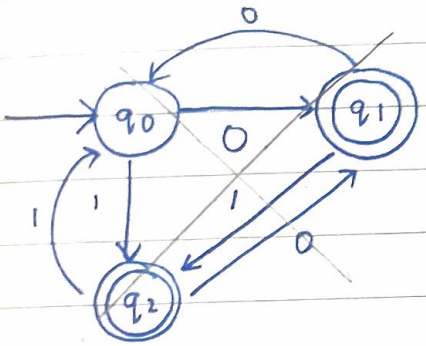
w - string
u/v - substrings

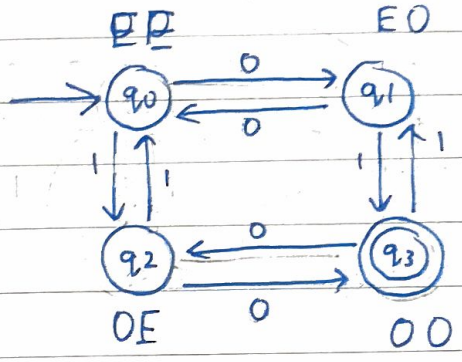
store
67

Q. Design a DFA over $\Sigma = \{a, b\}$ that accepts a string which contains 2 consequtive as

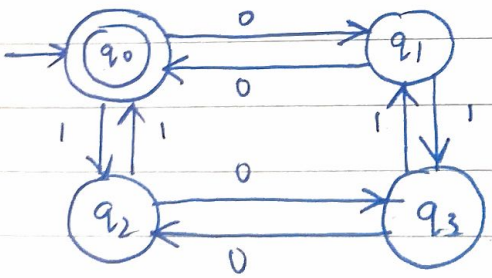


Q. Design a DFA over $\Sigma = \{0, 1\}$ that accepts a string w if there are odd number of 0's & 1's

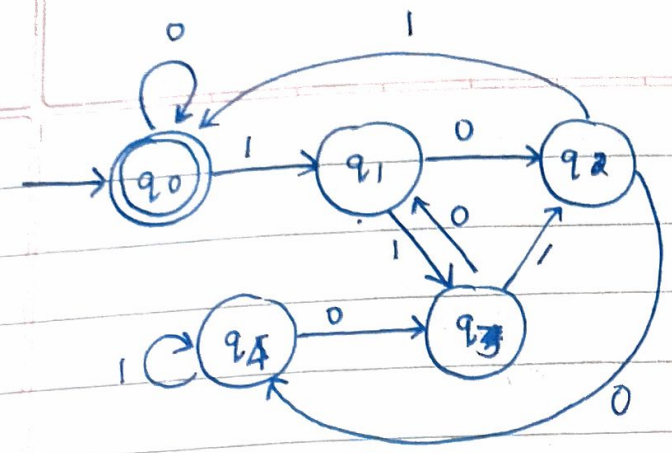




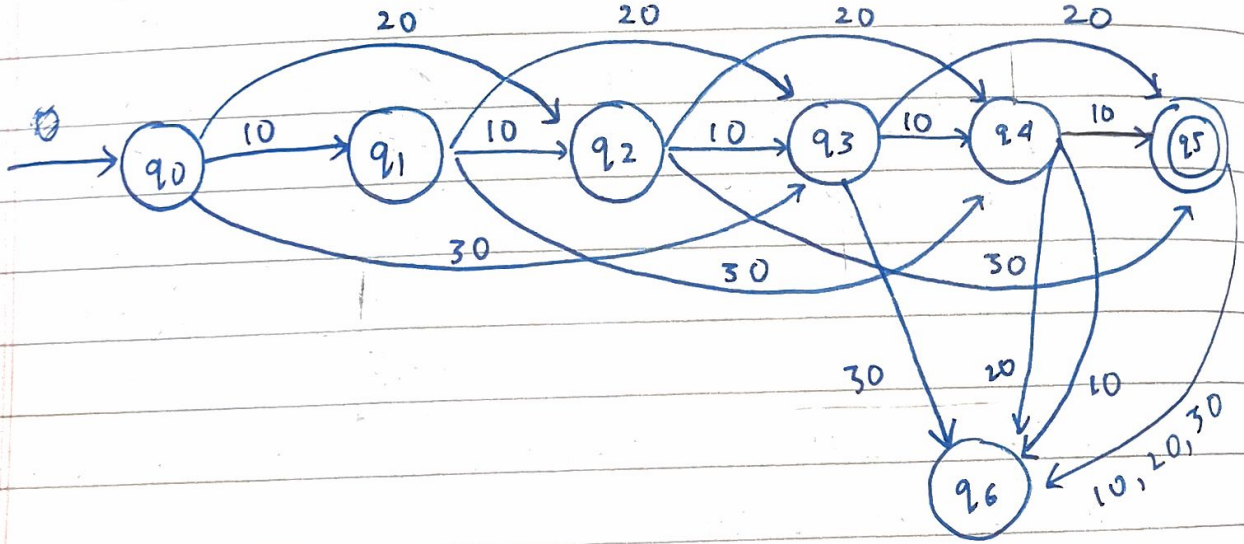
Q. Design a DFA over $\Sigma = \{0, 1\}$ accepting strings with even no. of 0's and 1's



Q. Design a DFA over $\Sigma = \{0, 1\}$ that accepts a string w if it divides by 5



Q. Design a DFA that displaces coca cola cans if we deposit 50 rupees. The denominations should be 10, 20, 30



1. All binary strings such that the no. is divisible by 3
(Assumption: Empty string is divisible by 3)
- For a no. to be divisible by 3 - sum of digits should be divisible by 3

for divisibility

✓ 78348321 sum = 36 rem = 0
× 78328322 sum = 37 rem = 1
× 78328323 sum = 38 rem = 2

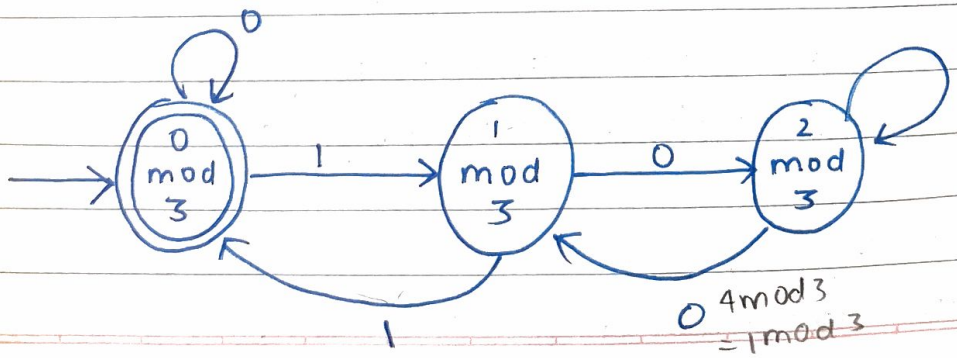
} equivalence classes / partitions

1. Identify the states
In DFA: $\Sigma = \{0, 1\}$ (reading strings from MSB)
In NFA: $\Sigma_\varepsilon = \{\varepsilon, 0, 1\}$

$\{Q, \Sigma, \delta, q_0, F\}$
$\delta : Q \times \Sigma \rightarrow Q$
} DFA

$\{Q, \Sigma_\varepsilon, \delta, q_0, F\}$
$\delta : Q * \Sigma \rightarrow P^Q$ (Power set of Q) — set of all subsets

M



$w = 101010011 \longrightarrow N$

$w0 = 1010100110 \longrightarrow 2N+0 \pmod 3$

$w1 = 1010100111 \longrightarrow 2N+1 \pmod 3$

$w10 = 10101001110 \longrightarrow 2(2N+1)+0 = 4N+2 \pmod 3$

$w11 : 2(2N+1)+1 = 4N+2+1 = 4N+3 \pmod 3$

only

THEOREM : M accepts strings such that the binary number is divisible by 3 (i.e)

M accepts a binary string $\Leftrightarrow$ the no. corrosponding to the binary string is divisible by 3

PROOF

LEMMA: Given a binary number w. After consuming w the machine M will be in —

(i) q_0 state if $w \pmod 3 = 0$ (if w = empty string then $w \pmod 3 = 0$)

(ii) q_1 state if $w \pmod 3 = 1$

(iii) q_2 state if $w \pmod 3 = 2$

We prove the statement by induction on the length of the input string $|w|$. (by removing)

- **Base case** : $|w| = 0$ Empty strings are accepted & the machine will correctly be in the state q_0

- $|w| = 1 \Rightarrow w : \{0, 1\}$

— $w = 0$ You will be in q_0 after consuming w. It should be in q_0 correctly as $0 \pmod 3 = 0$

- $w = 1$ $\quad 1 \pmod 3 = 1$ After consuming w, the machine will correctly be in q_1
- $|w| = 2 \Rightarrow w = \{00, 01, 10, 11\}$
  $\quad q_0 \quad q_1 \quad q_2 \quad q_0$

Hypothesis: Assume the statement to be true for all strings w of length $\leq K$

Extension: $|w| = K+1 \longrightarrow$ Delete LSB

Get a string x by deleting the last bit

$w = w_1 w_2 \cdots\cdots w_{K+1}$

$x = w_1 w_2 \cdots\cdots w_K$

- CASE 1: $w_{K+1} = 0$ $\quad x \rightarrow N \quad w \rightarrow 2N$

  $N = \ell \bmod (3)$

  $2N = 2\ell \bmod (3)$

  If x is in $q_{\ell (\bmod 3)}$ then w will be in $q_{2\ell (\bmod 3)}$

- CASE 2: $w_{K+1} = 1$ $\quad x \rightarrow N \quad w \rightarrow 2N+1$

  $N \rightarrow \ell (\bmod 3)$

  $2N+1 \rightarrow (2\ell+1)(\bmod 3)$

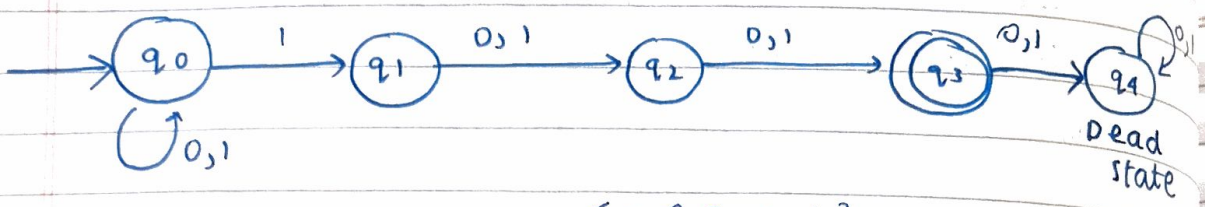
  If x is in $q_{\ell (\bmod 3)}$, then w will be in $q_{(2\ell+1)(\bmod 3)}$

Hence the lemma is proved.

Theorem is a corollary statement - proved (in both directions)

I've got this stew in my head
I'm not sure how it ends up like

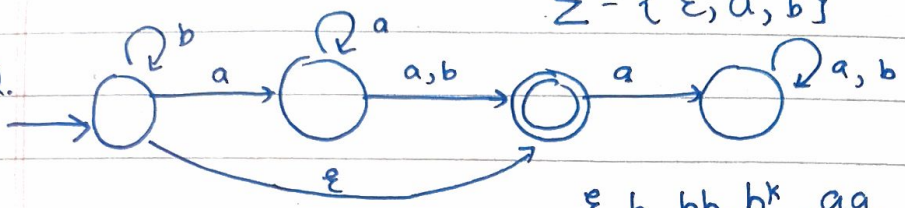
NFA : all strings where 3rd bit from the end is 1



Q.

Eg 1

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



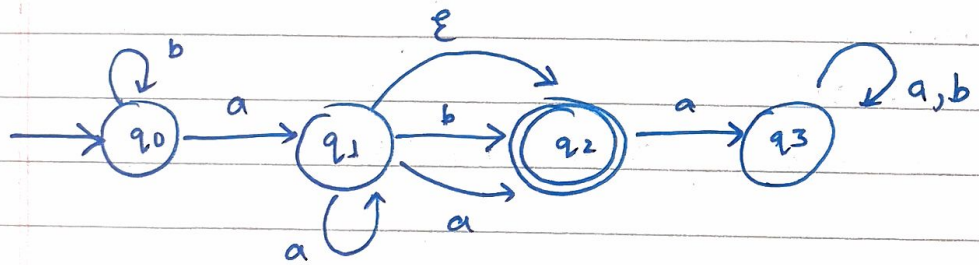
$\varepsilon, b, bb, b^k, aa, aaa, a^k$

$aaa^k b$

b - won't be valid but $b = b\varepsilon$ which is valid

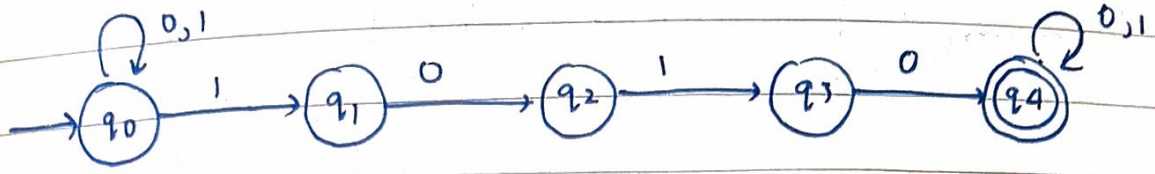
Invalid : $a, aba^k, abab^k, abb^k, ba$

Eg 2

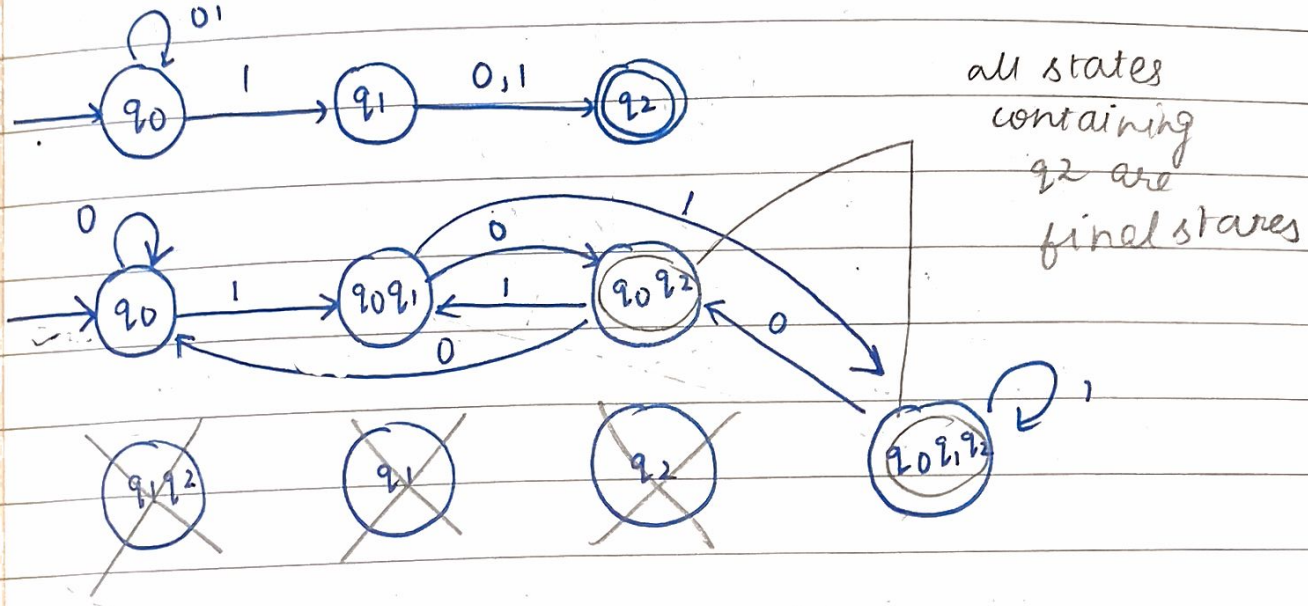


ε transition : If ∄ no input is provided also in an NFA, you can go from one state to another

Q. strings having 1010 as substring



Q. 2nd bit from the end is 1



$\delta(q_0, 0) = \{q_0\}$

$\delta(q_0, 1) = \{q_0, q_1\}$

$\delta(\{q_0, q_1\}, 0) \rightarrow \{q_0 q_2\}$

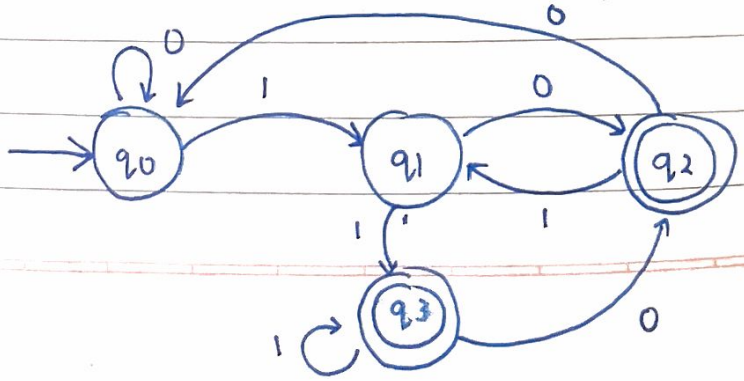
$\delta(\{q_0, q_1\}, 1) \rightarrow \{q_0 q_1 q_2\}$

$\delta(\{q_0 q_2\}, 0) \rightarrow \{q_0\}$

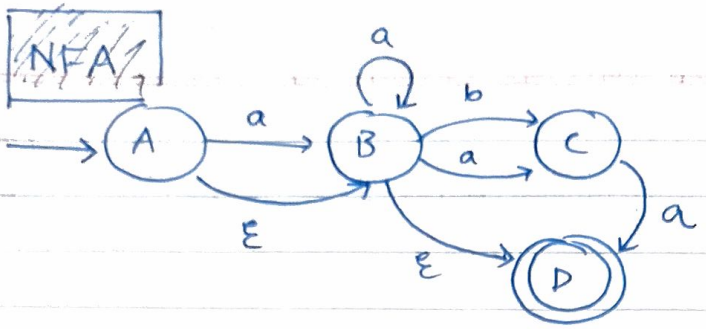
$\delta(\{q_0 q_2\}, 1) \rightarrow \{q_0, q_1\}$

$\delta(\{q_0 q_1 q_2\}, 0) \rightarrow \{q_0 q_2\}$

$\delta(\{q_0 q_1 q_2\}, 1) \rightarrow \{q_0 q_1 q_2\}$



Q.



$\delta(ABD, a) \rightarrow BCD$

$\delta(A, a) \rightarrow BCD$

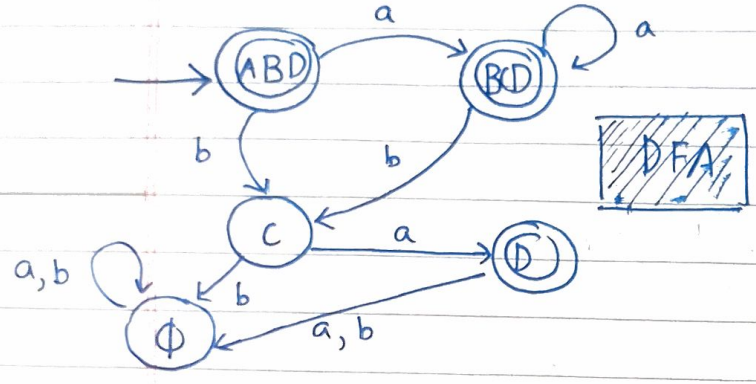
$\delta(B, a) \rightarrow BC$

$\delta(D, a) \rightarrow \phi$

$\delta(ABD, b) = C$

$\delta(BCD, a) = BCD$

$\delta(BCD, b$



Q2

DFA

Q3.

$q_0q_1, 0 \rightarrow q_0q_2$

$q_0q_1, 1 \rightarrow q_0q_1q_2$

$q_0q_2, 0 \rightarrow q_0q_3$

$q_0q_2, 1 \rightarrow q_0q_1q_3$

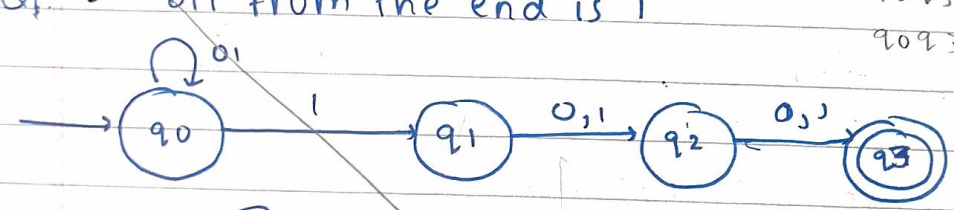
$q_0q_3, 0 \rightarrow q_0q_3$

$q_0q_3, 1 \rightarrow q_0q_1q_3$

DFA

Q4.

Q. 3rd bit from the end is 1



$q_0q_1q_30 \rightarrow q_0q_2q_3$

$q_0q_1q_3, 1$

$q_0q_1q_2q_3$

$q_0q_1q_20$

$q_0q_2q_3$

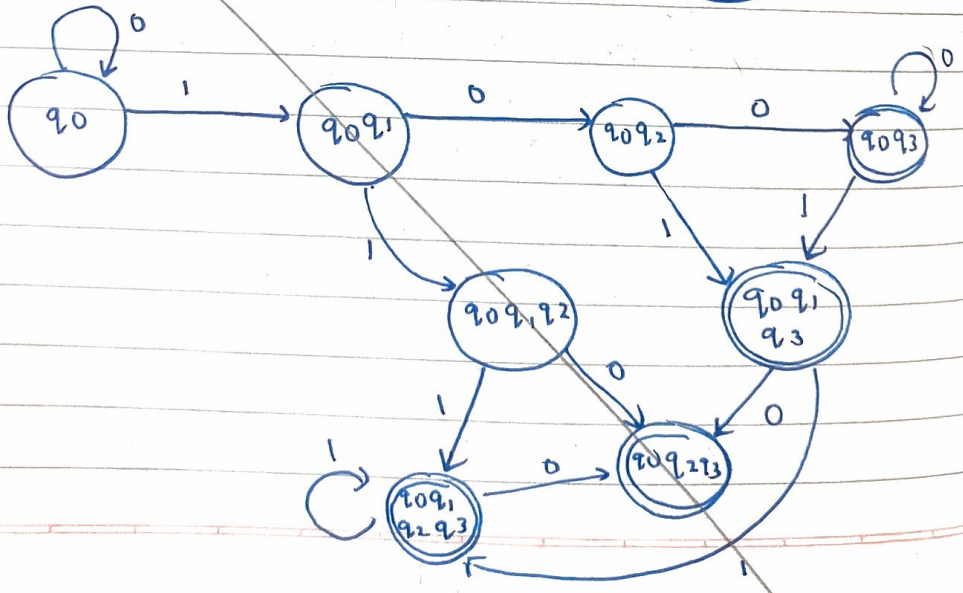
$q_0q_1q_31$

$q_0q_1q_2q_30$

$q_0q_2q_3q_3$

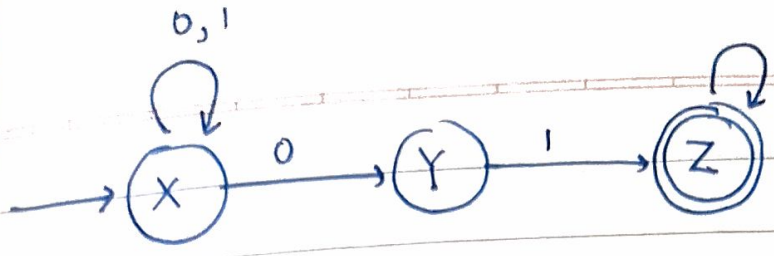
$q_0q_1q_2q_3$

$q_0q_1q_2q_3$



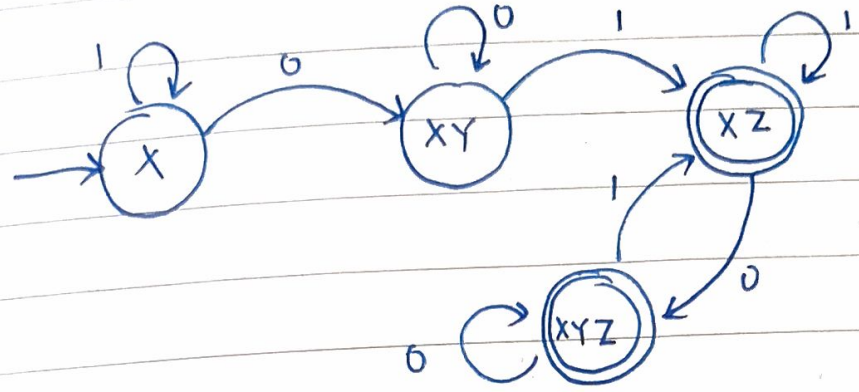
store
67

Q2.



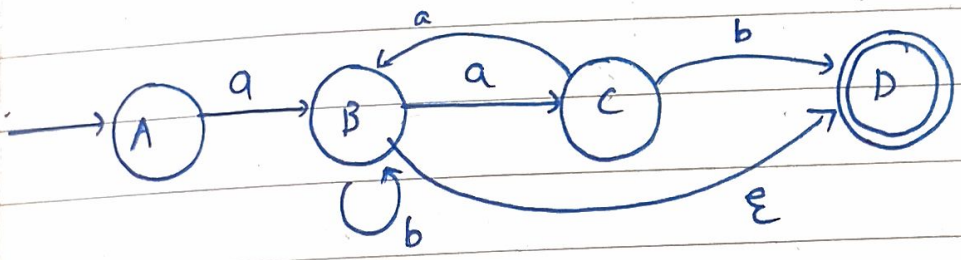
XY, 0 = XY
XY, 1 → XZ
XZ, 0 → XYZ
XZ, 1 → XZ
XYZ, 0 → XYZ
XYZ, 1 → XZ

DFA



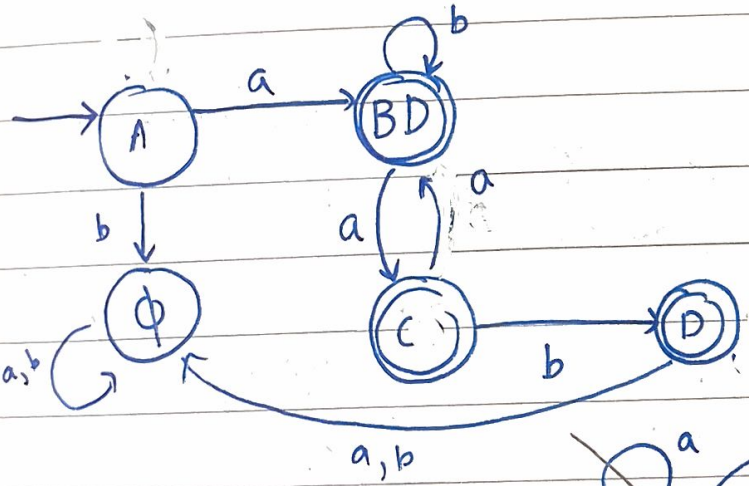
BD, b → BD

Q3.



a
a ε AP
A, a → ABD
A, b → ∧
BD, a → C
BD, b → BD
CD, a → BD
CD, b → DI

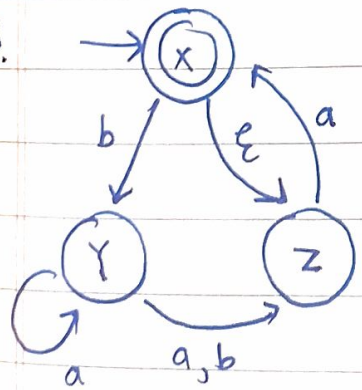
DFA



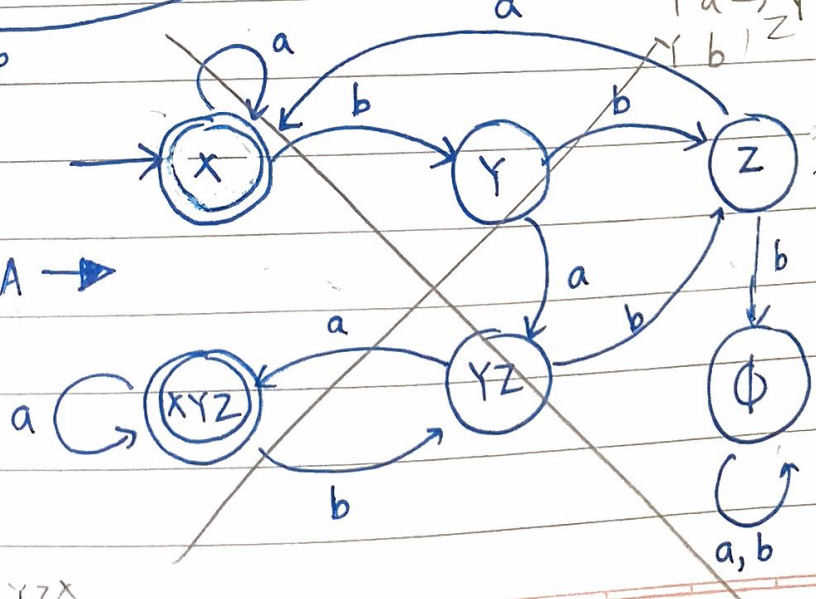
a = εa

X, a → X
X b → Y
Y a → YZ
Y b → Z

Q4.



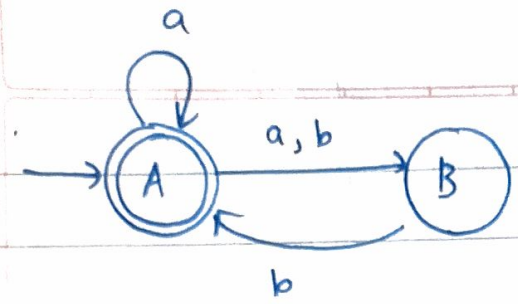
DFA →



Za, a
Zb, Z
YZ, a → YZX
YZ, b → Z

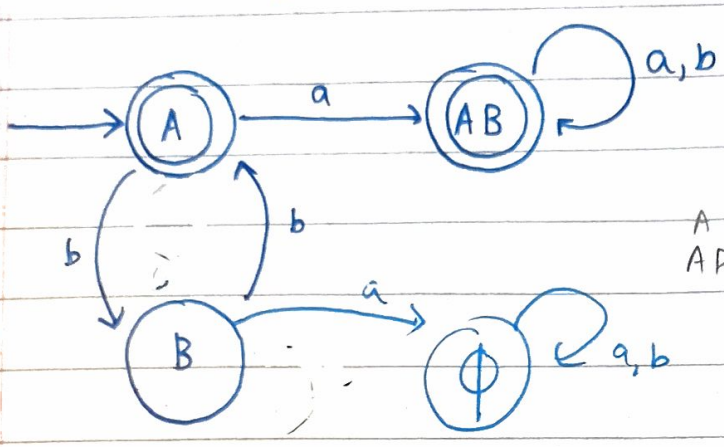
XYZ, a → XYZX
XYZ, b → YZ

Q5.



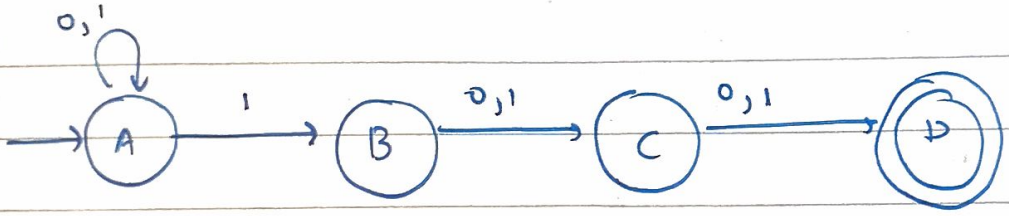
AB, b → BA

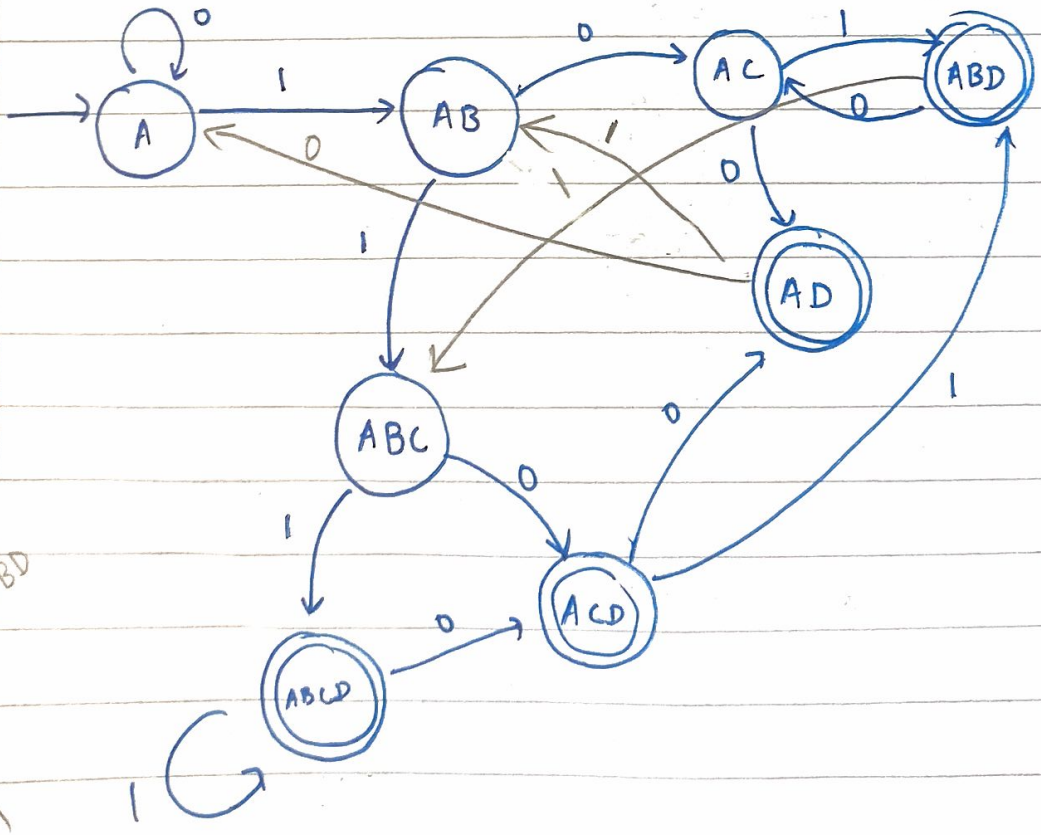
B, a



AB, 0 → AC
AB, 1 → ABC
AC, 0 → AD
AC, 1 → ABD
ABD, 1 → ABC
ABD, 0 → AC
AD, 0 → A
AD, 1 → AB

Q1.





ABC, 0
ABD

ABC, 1
ABCD

ACD, 0
AD

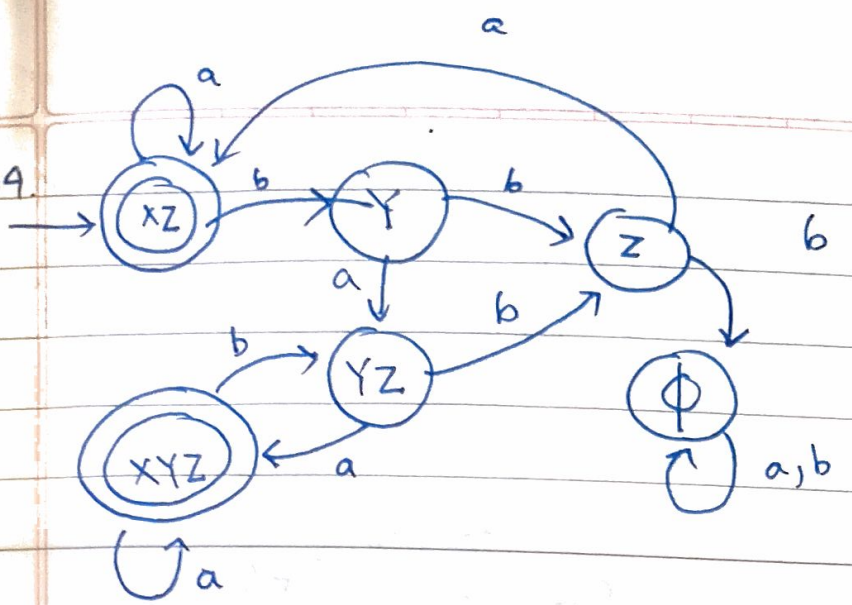
ACD, 1 → ABD

ABCD, 0
ACD

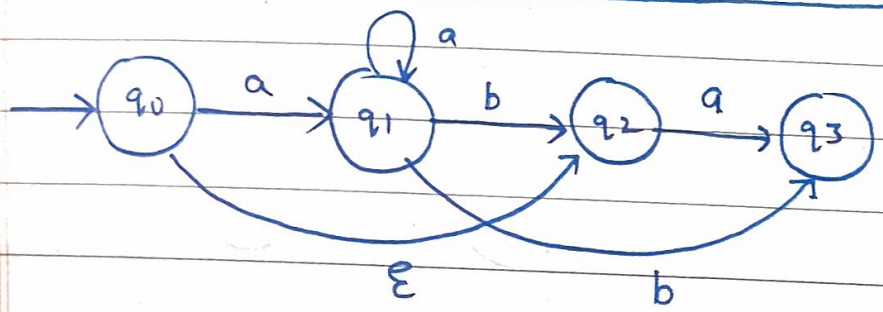
ABCD, 1
ABCD

store
67

A4



Generalized Transition Function



$\delta : \Sigma \times Q \rightarrow pQ$

$\delta(q_0, a) \rightarrow \{q_1, q_3$

$\delta(q_0, ab) \rightarrow \delta(\delta(q_0, a), b)$

$= \delta(\{q_1, q_3\}, b) = \{q_2, q_3\}$

$\boxed{\delta(q_i, a) \rightarrow S_j}$ — set of states

$\delta(q_i, w_1 w_2 \cdots w_k) = \delta(\delta(q_i, w_1), w_2 \cdots w_k)$

$= \delta(\delta(\delta(q_i, w_1), w_2), w_3 \cdots w_k)$

Definition

$\delta(q_0, w) \rightarrow s_j$ w is accepted by FSM M if $s_j \cap F \neq \phi$ and is rejected if $s_j \cap F = \phi$

P.T

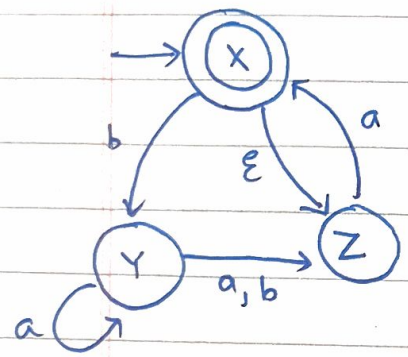
SUBSET CONSTRUCTION IS CORRECT

Let w be a string in a language such that an NFA accepts it then $\delta(q_0, w) \longrightarrow s_j$ such that $s_j \cap F \neq \emptyset$

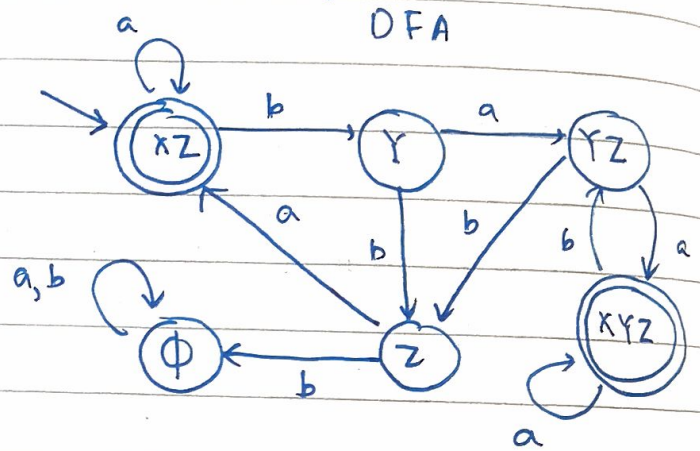
refers to a set of multiple states

In a DFA, $\delta(q_0, w) \longrightarrow s_j$ (one of the states in the set s_j)

NFA



DFA



$\delta(X, \varepsilon) = \{X, Z\}$

$\delta(XZ, baa) = \{X, Y, Z\}$

$\delta(X, bab) = \{Z\}$

$\delta(XZ, \varepsilon) \rightarrow XZ$

$\delta(XZ, baa) \rightarrow XYZ$

$\delta(XZ, bab) \rightarrow Z$

equivalent!

Any language accepted by a FSA is called REGULAR

CLOSURE PROPERTIES –

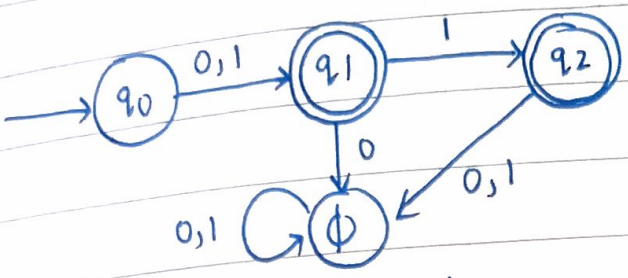
1. complementation
2. Union
3. set Intersection
4. set Difference
5. concatenation : L_1L_2

Final state of M_1 = Initial state of M_2

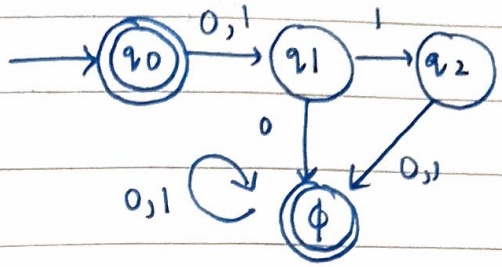
complementation

$L = \{0, 1, 01, 11\}$

$\bar{L} = \{\varepsilon, 10, 00, \text{any str of len} >= 3\}$



Accepts L

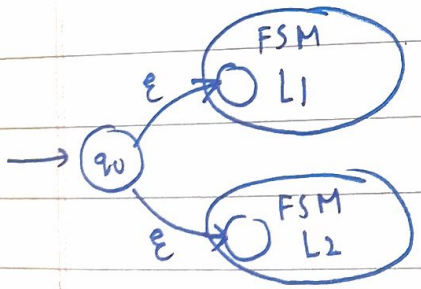


Accepts $\bar{L}$

(can only do this for DFAs not NFAs - toggling final and non-final states to get complement)

set of regular languages is closed under complement.

Union : L_1 & L_2 are regular languages $(L_1 \cup L_2)$



Intersection : L_1 & L_2 are reg. langs. $(L_1 \cap L_2)$

$\overline{(L_1 \cup L_2)} = \bar{L_1} \cap \bar{L_2}$

$\therefore \bar{L_1} \cup \bar{L_2} = \bar{\bar{L_1}} \cap \bar{\bar{L_2}} = L_1 \cap L_2$

union & complements are regular. so $L_1 \cap L_2$ is ~~reg~~ are regular

set difference : $L_1 - L_2 = L_1 \cap \overline{L_2}$

$\overline{L_1 \cup L_2} = \overline{L_1} \cap \overline{L_2} = L_1 \cap \overline{L_2}$

complement & union are regular ⇒ set diff is regular

TUTORIALS//

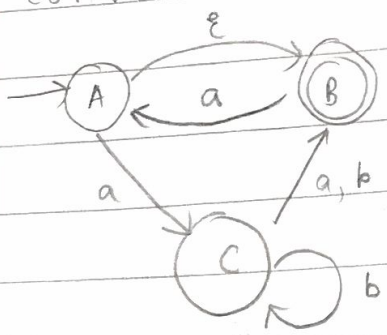
Q1. $L = \{w \mid$ such that $w_i w_{i+1} w_{i+2} w_{i+3} w_{i+4}$ will have atleast 2 zeroes

Q2. $L = \{w \mid w$ does not contain a pair of 1s that are separated by an odd no. of symbols

Q3. $L = \{w \mid w$ contains an even no. of 1s and an odd no. of 0's and does not contain the substring '10'

Q4. $L = \{w \mid$ every odd position of w is 1$\}$

Q5. convert nfa → dfa



Q6. The difference of 2 regular languages is regular

Q7. If L is regular then so is L^R

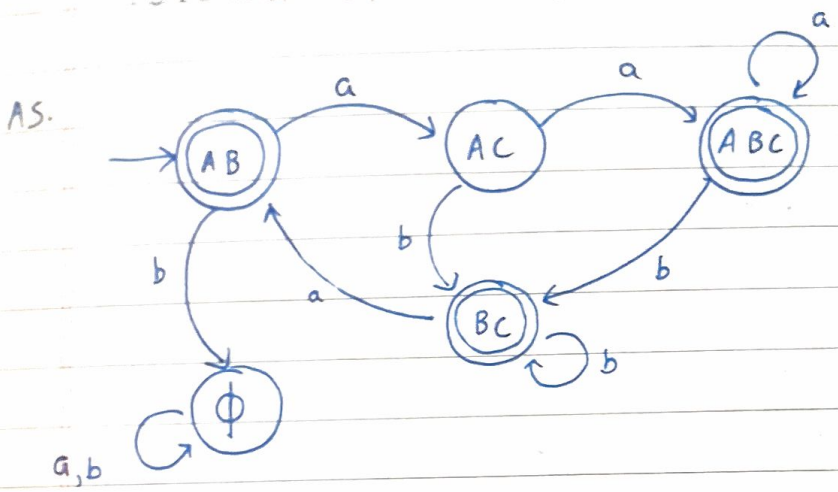
$L^R = \{w_k w_{k-1} \rightarrow w_k \mid \forall k \in L\}$

Q8) If L is a regular language and 'a' is a symbol then L/a, the quotient of L and a is the set of strings w such that wa is in L

Eg: $L = \{a, aab, baa\}$ then $L/a = \{\varepsilon, ab, ba\}$

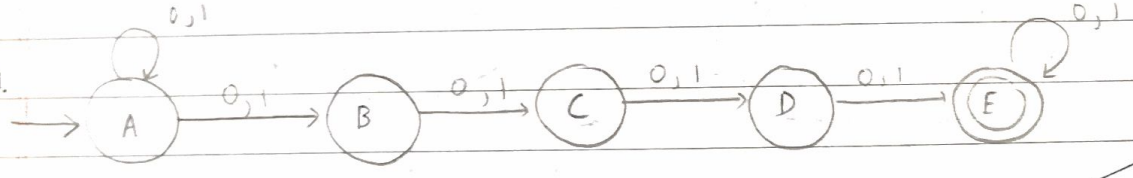
Theorem: L/a is regular

A5.



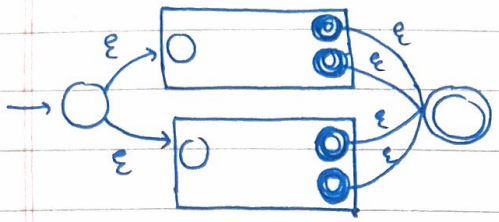
$\delta(AB, a) \rightarrow C$
$\delta(AB, b) \rightarrow \phi$
$\delta(AC, a) \rightarrow ACB$
$AC, b \rightarrow C$
$ABC, a \rightarrow AC$
ABC, b
$BC, a \rightarrow AB$
$BC, b \rightarrow BC$

A1.

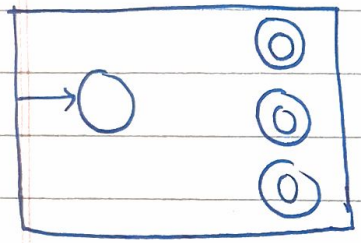


$S = \{L_1, L_2, L_3 \cdots \cdot L_K\}$

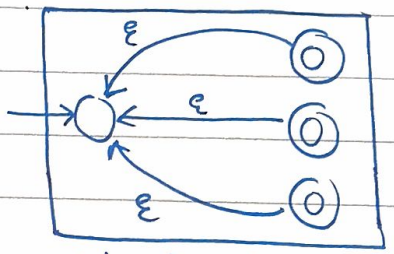
$L_i \cup L_j = L'$



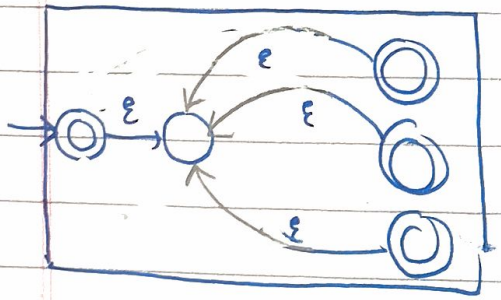
$$L^* = \bigcup_{K=0}^{\infty} L^K = L^0 \cup L^1 \cup L^2 \cdots \cdots L^\infty$$



L

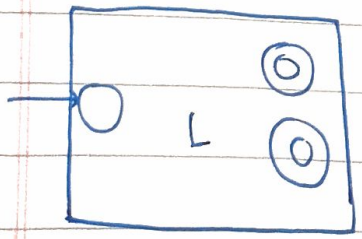


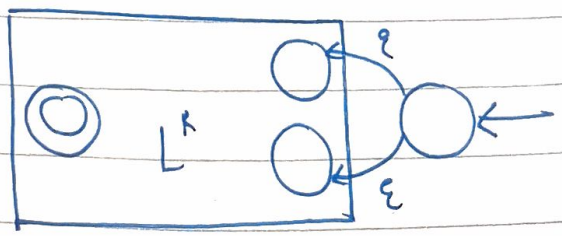
$L^1 \cup L^2 \cup L^3 \cdots = L^+$



$L^* = L^0 \cup L^+$

Q. Reversal



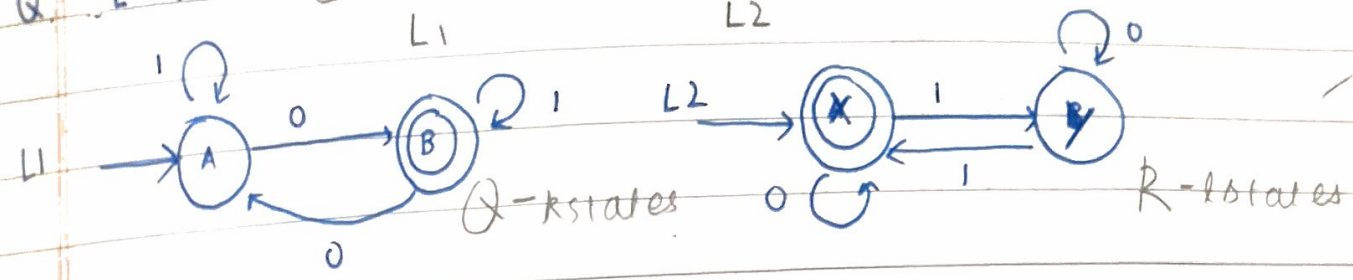


only 1 initial state required

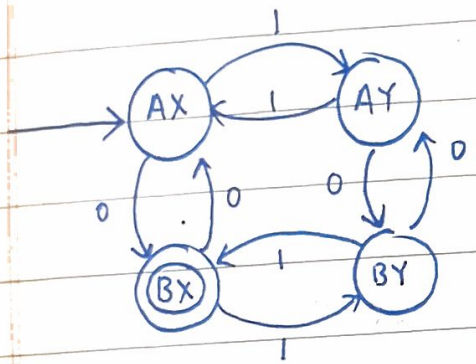
store
67

Q. $L = \{w \mid w$ has odd 0's & even 1's$\}$ $\quad L = L1 \cap L2$

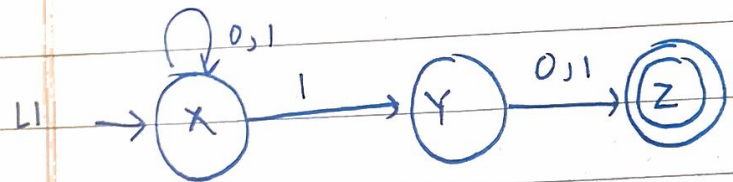
($L1$: has odd 0's; $L2$: even 1's)

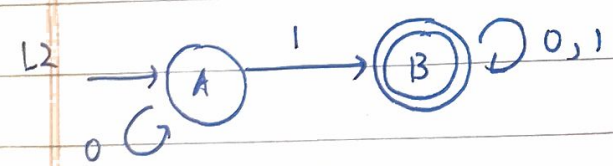


$L = Q \times R$ states (total no. of states = k^2)



Q. NFA that accepts all strings with 2nd last bit = 1 = L1

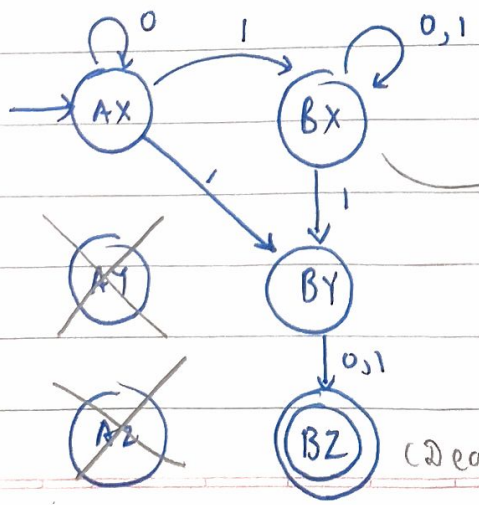




Accepts all strings containing atleast one 1

$L1 \subset L2$

$\Rightarrow$ NFA = L1 itself



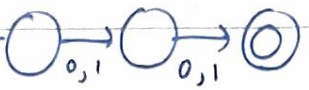
cannot minimize an NFA
can only minimize a DFA

◇ $\Sigma = \{0,1\}$

Q $L = \{w \mid |w| \leq k\}$ Is it regular? $k \in \mathbb{N}$

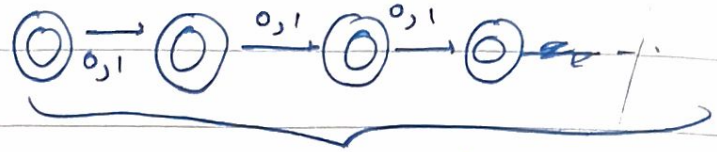
$L_i = \{w \mid |w| = i\}$ — can construct DFA for eac

by using i+1 states

$L_2 \rightarrow$ 

then ∪ all of them

OR



i+1 states

YES, REGULAR

Q. $\Sigma = \{0,1\}$ $L = \{k \text{ strings}\}$ $k \in \mathbb{N}$

Is L regular?

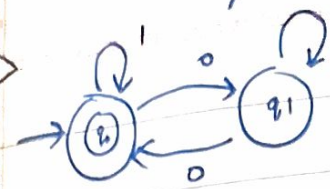
→ If a language has finite no. of strings - it is regu

HWQ $L = \{0^n 1^n \mid n \geq 0\}$ construct a DFA

HWQ. $L = \{w \mid w$ has equal no. of 0's and is $\}$

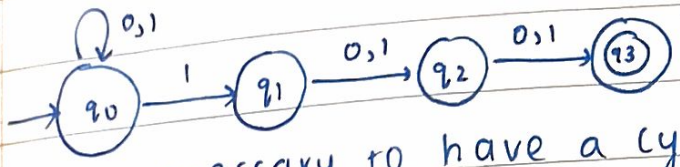
store
67

→ $(1^*01^*0)^*$

EX. 

DFA that accepts even no. of 0's
Infinite lang but regular lang.

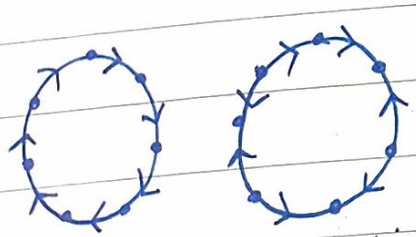
CYCLES: q_0q_0, $q_0q_1q_0$, q_1q_1



3rd bit from end is 1

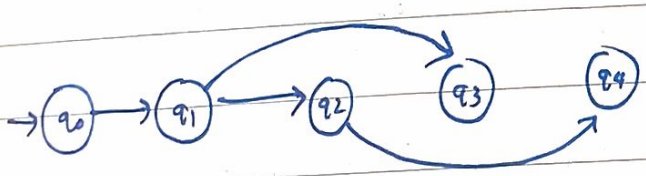
CYCLE: q_0q_0

Is it necessary to have a cycle for an ∞ lang to be a regular lang? YES

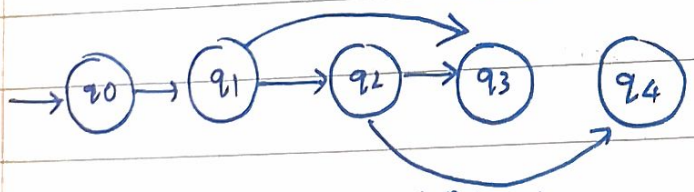


Directed cycle

undirected cycle [Not a cycle in directed sense]



TREE (UNDIRECTED ACYCLIC)



$q_1q_2q_3$ is an undirected cycle

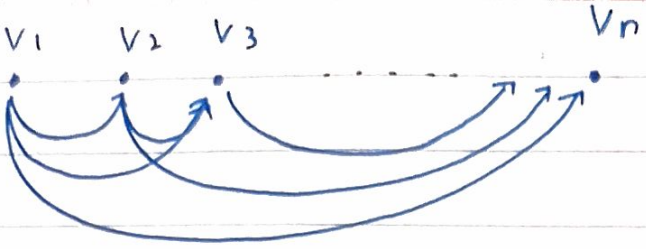
DIRECTED ACYCLIC GRAPH.

Maximum no. of edges in UNDIRECTED ACYCLIC GRAPH if no. of nodes = n is $\underline{(n-1)}$

If n nodes are present in Directed acyclic graph

$$= {}^nC_2 = \frac{n(n-1)}{2} = \sum_{i=0}^{n-1} i$$

1 2 3 . . n
n-1 n-2 n-3 0



$V_1 : n-1 \quad V_2 : n-2 \quad V_3 : n-3 \quad \ldots \quad V_n = 0$

No. of edges $= \sum_{k=0}^{n-1} k$

Take a string $w \ni |w| \geq n$

∴ There should be a walk (vertices can repeat)

The walk should contain n edges but only n-1 are possible

$\sum_{i=0}^{n-1} 2^i = 2^n - 1$ — only these many strings are possible — finite lang — contradiction

PUMPING LEMMA FOR REGULAR LANGUAGES

If L is a regular language then there exists an integer P (the pumping length) such that every string $s \in L$ where $|s| >= P$ maybe decomposed into three parts as $s = XYZ$ obeying the following conditions —

(i) $|XY| <= P$

(ii) $|Y| > 0$ (i.e $|Y| >= 1$)

(iii) $\forall\, i >= 0,\ XY^iZ \in L$ ∴

store
67

In a finite lang, string size is finite — K
choose $P >= (K+1)$ - if not satisfied doesn't need to be decomposed
∴ All finite langs. are regular

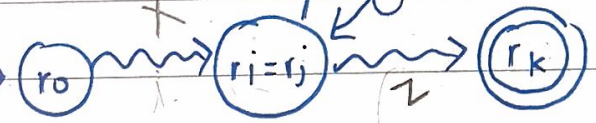
PROOF:

Let the Pumping Length $P = n$ (no. of states in FSA)
Now any string in L (say s) such that $|s| >= P$
will be accepted only after $|s|$ transitions
If $Q = \{q_0, q_1 \cdots q_{n-1}\}$, then the way 's' is accepted would be like $r_0 r_1 \cdots r_k$ where $r_0 = q_0$; $r_j = q_j$ and $r_k \in F$ (walk)

since $K >= n$ and a walk of length K+1 vertices should have a repitition (By Pigeon Hole Principle)

$\Rightarrow |Y| > 0$

Let $r_i = r_j$



$j = i + |Y|$

X = transitions from r_0 to r_i
Y = transitions from r_i to r_j
Z = transitions from r_j to r_k

Let $r_i = r_j$ such that j is minimum
This ensures $|XY| <= P$
T.P : $j <= P+1$
no repeating states b/w 0 to j ∴ $j <= n$
$\Rightarrow j <= P+1$

0 (1) j (j+1) ≤ n (+1)

since $j <= p+1$ we have that $|xy| <= p$ and $|y| \geq 1$
& $xy^i z \in L \quad \forall \; i >= 0$

~~$\delta((q_0, x) \rightarrow (\delta r_i, y^i) \rightarrow \delta \delta r_i, \delta r_j$~~

$\delta(q_0, x) \rightarrow r_i$

$\delta(r_i, y^i) \rightarrow r_i$

$\delta(r_1, z) \rightarrow r_K$

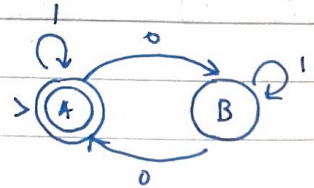
Pumping Lemma for Regular Languages

If L is a regular language, then ∃ an integer p (pumping length) such that any string $s \in L$ and $|s| \geq p$ may be partitioned as $s = xyz$ obeying the following properties –

(i) $|xy| <= p$

(ii) $|y| \geq 1$

(iii) $xy^i z \in L \quad i >= 0$



Q1) Even number of zeroes

Ans Let $s = 1$

Taking $p = 1$

$s = xyz = \varepsilon 1 \varepsilon \Rightarrow x = \varepsilon \quad y = 1 \quad z = \varepsilon$

i	$xy^i z$
0	ε
1	1
2	11
3	111

NOW $s = 0$

i	$xy^i z$
✗ 0	0
✓ 1	00
✗ 2	000

Taking p

$s = 1$ x

$s = 00$ x

↳ Partiti

x

x

Any stri

{ 00 , ...

$s = s_1 s$

- $s_1 s_2 = 00$
- $s_1 s_2 = 01$
- $s_1 s_2 = 10$
- $s_1 s_2 = 1$

store 67

NOW $S=00$ $X=\varepsilon$ $Y=0$ $Z=0$

i	XY^iZ	
✗ 0	0	$\Rightarrow XY^iZ = 0^i0$
✓ 1	00	since XY^iZ is not accepted
✗ 2	000	for $i=0$ / $i=2$, pumping length selected is wrong

---

taking $p=2$

$S=1$ $X=\varepsilon$ $Y=1$ $Z=\varepsilon$ XY^iZ is accepted $\forall\ i>=0$

$S=00$ $X=\varepsilon$ $Y=0$ $Z=0$ $XY^iZ = 0^i0$ not accepted

↳ Partition it in a different way

$X=\varepsilon$ $Y=00$ $Z=\varepsilon$

$XY^iZ = (00)^i$ — accepted $\forall\ i>=0$

Any strings of length 2 can be

{ 00 , 01 , 10 , 11

$S = \underbrace{S_1 S_2}\ S'$

- $S_1S_2 = 00 \Rightarrow S = XYZ$ $X=\varepsilon$ $Y=00$ $Z=S'$
- $S_1S_2 = 01 \Rightarrow X=0$ $Y=1$ $Z=S'$
- $S_1S_2 = 10 \Rightarrow X=\varepsilon$ $Y=1$ $Z=0S'$
- $S_1S_2 = 11 \Rightarrow X=1$ $Y=1$ $Z=S'$

  $X=\varepsilon$ $Y=1$ $Z=1S'$

  $X=\varepsilon$ $Y=11$ $Z=\varepsilon$

Q2. All strings that end with 010

Q3. All strings having '101' as substring

Q4. strings where the number is divisible by 3

Q5 strings whose length is a multiple of 3

Q6. Equal occurrences of '01' & '10'

A2.) P=1 $S: S_1 S_2 010$

$S_1 = \varepsilon$ $S_2 = 0$ / 1

$S = \varepsilon 0^i 010$

$Z = 010$

$P = 1$

$X = \varepsilon$ $Y = 0$ or 1 $Z = S'010$

But if $S = 010$

$X = \varepsilon$ $Y = 0$ $Z = 10$ - not in language

wont be accepted

P=2 Wont work for 010 as $|S| \geq 2$ (010)

P=3 Wont work for 010 as $|S| \geq 3$ (010)

P=4 $S = S_1 S_2 \cdots S_{k-3}\, 010$

$X = \varepsilon$, $Y = S_1$, $Z = S_2 \cdots S_{k-3}\, 010$

$S = 010$ - $|S| = 3$ does not satisfy Pumping lemma

A3. Let $P = 4$

$S = S_1 S_2 S_3 \cdots S_{k-3}\, 1\, 0\, 1\, S_{k+1} S_{k+2} \cdots S_n$

$X = \varepsilon$ $Y = S_1$ $Z = S_2 S_3 \cdots 101 S_{k+1} S_{k+2} \cdots S_n$

Suppose $s = 1010$ / 1011 - doesnt work BT!

out of my

store
67

Q. $L = \{0^n 1^n \mid n >= 0\}$

Let pumping length = P

consider $s = 0^P 1^P$

$x = 0^j \quad (0 \le j \le p-k)$

$y = 0^k \quad (j+k+l = p) \quad (1 \le k \le p)$

$z = 0^l 1^P$

$xy^0z = 0^j 0^l 1^P = 0^{j+l} 1^P \notin L$

$xy^1z = 0^j 0^k 0^l 1^P = 0^P 1^P \in L$

$xy^2z = 0^{P+k} 1^P \notin L$

The no. of zeroes & 1s are not equal when you pump up/down. ∴ By pumping lemma $0^P 1^P$ is not regular language

$x = 0^j \quad y = 0^{\frac{p-j}{2}} 1^{\frac{p-m}{2}} \quad z = 1^m$

$xy^iz = 0(01)(01)1$ ... (0's & 1's are mixed)

Q $L = \{w \mid w$ has equal no. of 0's and 1's$\}$

Let pumping length = P

consider $s = 0^{P/2} 1^{P/2}$

$x = \varepsilon \quad y = 0^{P/2} 1^{P/2} \quad z = \varepsilon$

however we partition it we need to show that it does not belong in the language on pumping it up

$s = 0^{P-2} 1^{P-2}$

$x = 0^{P-3} \quad y = 01 \quad z = 1^{P-3}$

$x = \varepsilon \quad y = 1^{P/2} 0^{P/2} \quad z = \varepsilon$

$s = 0^{P-1} 1^{P-1}$

$x = 0^{P-2} \quad y = 01 \quad z = 0^{P-2}$

Q1. $L = \{0^i 1^j \mid i > j\}$

Q2. $L = \{WW \mid W \in (0+1)^*\}$

Q3. $L = \{0^{n^2} \mid n >= 0\}$

Q4. $L = \{W \mid W \text{ is a palindrome}\}$

Q5. $L = \{WW^R \mid W \in (0+1)^*\}$

Q6. $L = \{0^p \mid p \text{ is prime}\}$

Q7. $L = \{W \mid \text{no. of 0's in } W > \text{no. of 1's in } W\}$

A1. Pumping length = P

$0^{p+1} 1^p$ ~~$m > n$ & $m + n >= p \Rightarrow m >= p - n$~~

$X = 0^j$

$Y = 0^k$ $\quad j + k + l = p + 1$ $\quad k \neq 0$

$Z = 0^l 1^p$ Pumping down

$XY^0Z = 0^j 0^l 1^p = 0^{j+l} 1^p$

$j + l = p + 1 - k$

$j + l \leq p + 1 - 1$

$\therefore 0^{j+l} 1^p \notin L$

$\therefore 0^i 1^j$ where $i > j$ is not a regular language.

A2. Let pumping length = P $\quad W = 0^n 1^n$ $\quad 4n \geq l$

Consider the string : $0^n 1^n 0^n 1^n = s$

such that $|s| > p$

$X = 0^k$

$Y = 0^{n-k} 1^l$

$Z = 1^{n-l} 0^n 1^n$

$n + k - l + l \leq p$

① — $n + k \leq p \Rightarrow$

and $n - k + l >= 1$

② — $n + l >= 1 + k$

XY²Z

NOW

2n-K

① n+

②

⇒

Hence

A3. L = {0

S : (0

X = 0

Y = 0

Z = 0

XY²Z

F

Let s

X = 0

Y = 0

Z = 0

Pumpi

1 ≤

∴ 0 let p

$XY^2Z = 0^k 0^{2n-2k} 1^{2l} 1^{n-l} 0^n 1^n$

$= 0^{2n-k} 1^{n+l} 0^n 1^n$

NOW for XY^2Z to belong to L

$2n-k = n$ and $n+l = n$

① $n+k \leq p \leq 4n \Rightarrow n+k \leq 4n \Rightarrow k \leq 3n$

② $n+l \geq 1+3n$

$\Rightarrow n+l \neq n$

Hence on pumping up, $XY^2Z \notin L$ irregular

A3. $L = \{0^{n^2} \mid n >= 0\}$

Let pumping length = L

$S : (0^n)^n$

$X = 0^p$

$Y = 0^k$

$Z = 0^m$

$p+k+m = n^2$ — ①

$k >= 1$ — ②

$p+k \leq L$ — ③

Pumping length = p

$XY^2Z = 0^p 0^{2k} 0^m$

For XY^2Z to belong to L

$p+2k+m = n^2+k$

Let $s = (0^p)^p = 0^{p^2}$

$X = 0^l$ $\quad k \geq 1 \quad l+k \leq p$

$Y = 0^k$

$Z = 0^{p^2-k-l}$

Pumping up : $XY^2Z = 0^l 0^{2k} 0^{p^2-k-l} = 0^{k+p^2}$

$1 \leq k \leq p \Rightarrow p^2+1 \leq k+p^2 \leq p^2+p < p^2+2p+1$

$p^2 < k+p^2 < (p+1)^2$

$\therefore k+p^2$ is not a perfect square

$\therefore 0^{k+p^2} \notin L$ $\quad \therefore 0^{n^2}$ is not a regular language.

Pumping Lemma is used to prove irregularity of a lang. not regularity

Q4. $S = 0^n 1 0^n$ Pumping length $= n$

$X = 0^i$ $i+j \leq n$ $i+j = n$

$Y = 0^j$ $j \geq 1$

$Z = 10^n$ $i+j \geq i+1 \Rightarrow n \geq i+1 \Rightarrow i \leq n-1$

Pumping down: $XY^0Z = 0^i 1 0^n$

since $i \neq n$ $XY^0Z \notin L$, L is not a regular language

Q8. $L = \{0^n \mid n$ is a fibonacci number$\}$

Q9. $L = \{0^m 1^n \mid m \neq n\}$

Q10. $L = \{ww \mid w \in (0+1)^*\}$

Q11.

A5. $\{ww^R \mid w \in (0+1)^*\}$

$S = 0^k 1^k 1^k 0^k$ Pumping length $= 2k$

$X = 0^i$ $i+j = k$

$Y = 0^j 1^l$ $l+m = k$

$Z = 1^m 1^k 0^k$ $j+l > 0$

$i+j+l \leq 2k \Rightarrow k+l \leq 2k \Rightarrow l \leq k$

Pumping up

$XY^2Z = 0^i 0^{2j} 1^{2l} 1^m 1^k 0^k$

$= 0^{i+j+j} 1^{l+m+l} 1^k 0^k$

$= 0^{k+j} 1^{k+l} 1^k 0^k$

either $k+j \neq k$ or $k+l \neq k$

$\because j+l > 0$

$\Rightarrow l > 0, j \geq 0$

$\Leftarrow j > 0, k \geq 0$

Hence WW^R is not a regular language as $XY^2Z \notin L$

A6. 0^p (p is prime)

consider a string $s = 0^p$ p is prime, $p \geq n+2$

pumping length $= n$

$X = 0^i$ $Y = 0^j$ $Z = 0^k$

$j > 0$ $i+j \leq n$ $i+j+k = p \Rightarrow i+k = p-j$

Pumping up : ~~$XY^2Z \neq 0^{i+2j+k} \neq 0^{p+j}$~~

$XY^{p-j}Z = 0^i 0^{j(p-j)} 0^k$

$|XY^{p-j}Z| = i + j(p-j) + k$

$= (i+k) + j(p-j)$

$= 1(p-j) + j(p-j) = (p-j)(1+j)$

Now we need to prove that none of the factors are 1 as otherwise the string length can be prime (17×1) only composite nos. can be broken down into factors apart from 1 and itself

- $j > 0 \Rightarrow j+1 > 1$ $\therefore (j+1)$ can never be 1
- $0 < j \leq n \Rightarrow -n \leq -j < 0 \Rightarrow p-n \leq p-j < p$

$\Rightarrow n+2-n \leq p-j < p$

$\Rightarrow 2 \leq p-j < p$

$\therefore$ The minimum value of $(p-j) = 2$ $\therefore$ it can never be 1

Hence $XY^{p-j}Z \notin L$ as its strlen is composite

$\therefore 0^p$ is not a regular languge

A7. no.of 0s > no.of 1s

consider the string : $0^{n+1}1^n = s$

Let the pumping length = n

$X = 0^i \quad Y = 0^j \quad Z = 0^{n-i-j+1}1^n$

$i+j \leq n \quad \& \quad j > 0$

on pumping down:

$XY^0Z = 0^i 0^{n-i-j+1}1^n = 0^{n-j+1}1^n$

$j > 0 \Rightarrow -j < 0 \Rightarrow 0 > -j$

$\Rightarrow n+1 > n-j+1$

$\Rightarrow n+1-j < n+1$

since no.of zeroes < no.of ones $XY^0Z \notin L$

∴ The language is not regular.

A8. $L = \{0^n \mid n$ is a Fibonacci number $\}$

$s = 0^p$ $\quad p = (n+1)$ ← pumping length

CONTEXT

Regular

$S \rightarrow aXE$

$S \rightarrow bcD$

$D \rightarrow \varepsilon \mid$

Right &

$S \rightarrow aA$

$S \rightarrow b$

$S \rightarrow abd$

Left line

$S \rightarrow Aa \mid$

$S \rightarrow Ba \mid$

of every p

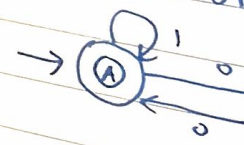
$S \rightarrow aA$

$A \rightarrow A$

$S \rightarrow bD$

Q1. Generate

number o



$\delta(q_i$

store
67

CONTEXT FREE LANGUAGES

Regular Languages using grammars

$S \to aXB$

$S \to bcD$

$D \to \varepsilon \mid EF \mid XYZ$

Right & left linear grammars

$S \to aA$
$S \to b$
$S \to abdD$

} All variables in every production should be on the right end at the RHS

Left linear grammars

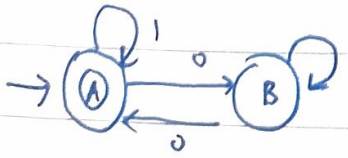
$S \to Aa \mid Sab$
$S \to Ba \mid b \mid \varepsilon$

} The variable on the right side of the production comes at the left end of every production

$S \to aA$
$A \to Ab$
$S \to bD$

} neither left/right linear much more powerful

Q1. Generate a grammar to accept strings with even number of zeroes



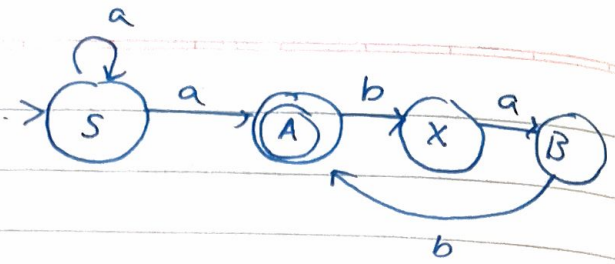
$A \to 0B \mid 1A \mid \varepsilon$

$B \to 1B \mid 0A \mid$

$\delta(q_i, a) \to q_j \quad : \quad q_i \to aq_j$

To produce right linear grammar

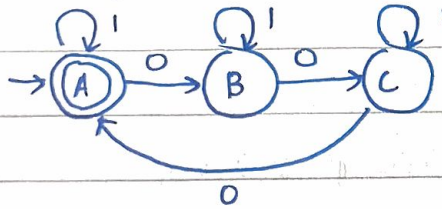
→ S → aA | as
A → baB | ε
B → bA
L : $a^+ + (bab)^*$



Q. Grammars with both right & left linear productions are not regular
Ex : A → ε | aAb
Language : $a^n b^n$ — not regular

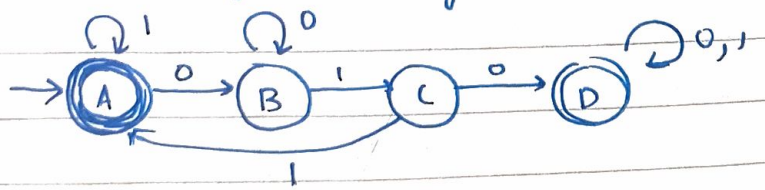
not regular { Ex : A → aB | ε
B → Ab

Q1. L = {w | no. of 0's is a multiple of 3}



1. A → 1A | 0B | ε
2. B → 1B | 0C
3. C → 1C | 0A

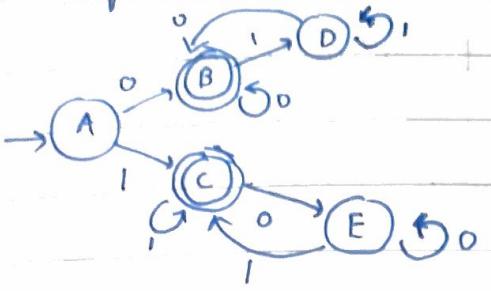
Q2. L = {All strings having '010' as substring}



1. A → 1A | 0B
2. B → 0B | 1C
3. C → 0D | 1A
4. D → 0D | 1D | ε

store
67

Q3. L = Equal occurrences of '01' and '10'



1. $A \rightarrow 0B \mid 1C$
2. $B \rightarrow \varepsilon \mid 0B \mid 1D$
3. $C \rightarrow \varepsilon \mid 0E \mid 1C$
4. $D \rightarrow 1D \mid 0B$
5. $E \rightarrow 0E \mid 1C$

TUTORIAL

HARD Q1. $L = \{a^n b^m \mid m \neq n\}$

Q2. $L = \{w \mid n_a(w) \neq n_b(w)\}$

Q3. $L = \{w \mid n_a(w) = n_b(w)\}$ → $S \rightarrow aSc \mid B$, $B \rightarrow bBc \mid \varepsilon$

Q4. $L = \{a^n b^m \mid n \leq m+2\}$

Q5. $L = \{a^n b^m c^k \mid k = n+m\}$

Q6. $L = \{a^n b^m c^k \mid n=m \text{ or } m<=k\}$

Q7. $L = \{w \in \{a,b,c\}^* \mid |w| = 3 n_a(w)\}$

Q8. $L = \{\text{all valid paranthesis}\}$

A! VALID: a, b, aab, abb, aaab, aaa, bbb

INVALID: ε, ab, aabb, aanbbb

1. $A \rightarrow CB \mid BD$
2. $B \rightarrow aBb \mid \varepsilon$
3. $C \rightarrow aC \mid a$ (FOR $n_a > n_b$)
4. $D \rightarrow Db \mid b$ (FOR $n_b > n_a$)

A3. $n_a(w) = n_b(w)$

$A \rightarrow abA \mid aAB \mid bAa \mid baA \mid \varepsilon$

or

$A \rightarrow aAbA \mid bAaA \mid \varepsilon$

A2. $L: \{n_a \neq n_b\}$

1. $B \rightarrow CA \mid DA \mid AC \mid AD$
2. $A \rightarrow aAbA \mid bAaA \mid \varepsilon$
3. $C \rightarrow aC \mid a$
4. $D \rightarrow bD \mid b$

A4. $n_a(w) \leq n_b(w) + 2$

VALID: b, ab, aab, aaab, bbb

INVALID: aaaab

1. $A \rightarrow CB \mid BD$
2. $B \rightarrow aBb \mid \varepsilon \mid bB$
3. $C \rightarrow aa \mid a \mid \varepsilon$
4. $D \rightarrow bD \mid \varepsilon$

A8. $S \rightarrow [S] \mid (S) \mid \{S\} \mid \varepsilon$

A6. $S \rightarrow S_1 S_2$

$S_1 \rightarrow aS_1bC \mid C$

$C \rightarrow cC \mid \varepsilon$

$S_2 \rightarrow \cdot AbDcC \mid AC$

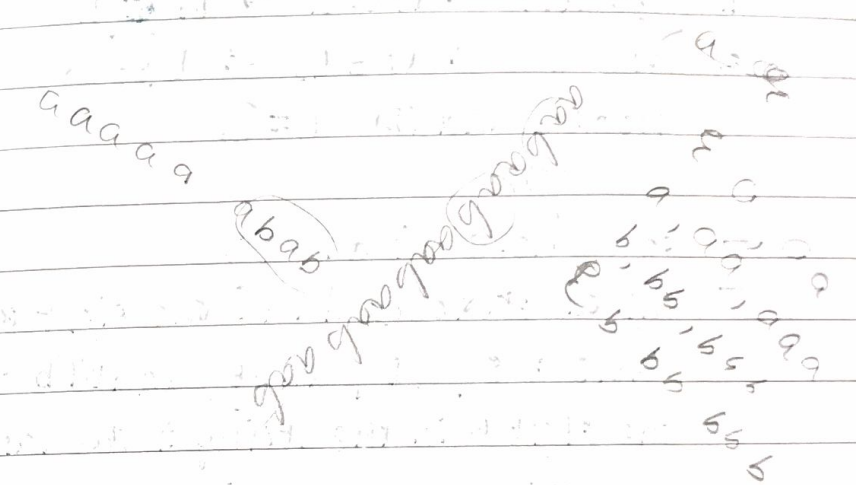
$A \rightarrow aA \mid \varepsilon$

$D \rightarrow bDc \mid \varepsilon$

babbaa

store
67

1. B → DADAC | DACAD | FAFAE | FAEAF
2. A → aAbA | bAaA | ε
3. C → aC | a
4. D → aD | ε
5. E → bE | b
6. F → bF | ε



aabDAC
aabaAC
aabaabCaaaaa

(A7) 1. S → SaB | BaS | ε
B → bb | cc | bc | cb

HOW TO PROVE THAT A GRAMMAR IS CORRECT ?

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

$S \rightarrow aSb \mid \varepsilon$

1. Every string the grammar generates is in L
2. Every string in the language L is generated by the grammar.

$L(S)$ = language generated by the grammar

easier to show

1. $w \in L(S) \Rightarrow w \in L \Rightarrow Ls \subseteq L$
2. $w \in L \Rightarrow w \in Ls \Rightarrow L \subseteq Ls$

From ① and ② $L = Ls$

Q.

- Equal no. of a's and b's
- All a's pre ceed the first B

VALID : ε, ab, aabb, aaabbb ....

The first b in the string is the last b generated - penultimate production

2. Is proved by induction on $|w|$
- $|w| = 0 \Rightarrow w = \varepsilon$ $S \rightarrow \varepsilon$ generates w
- $|w| = 2 \Rightarrow w = ab$ 1. $S \rightarrow aSb$ 2. $S \rightarrow \varepsilon$

$\Rightarrow S \rightarrow asb \rightarrow a\varepsilon b \rightarrow ab$

$w = a w' b \quad |w'| = |w| - 2$

- $|w| = 2k \Rightarrow w = a^k b^k$ $\quad w = aw'b$

$w' = a^{k-1} b^{k-1}$

$S \rightarrow aSb$

by inductive hypothesis can generate w'

$S \rightarrow aw'b$

$\Rightarrow S \rightarrow w$

Q. $L = \{w \mid n_a(w) = n_b(w)\}$ $\Sigma = \{a, b\}$

$L = \{\varepsilon, ab, ba, aabb, abab, abba, bbaa, babba, baaab \ldots\}$

$S \rightarrow aSb \mid bSa \mid \varepsilon$ (1st & last symbol are diff)

But we have strings in which (1st & last symbol a same)

so: $S \rightarrow aSbS \mid bSaS \mid \varepsilon$

$S \rightarrow aSb \mid bSa \mid \varepsilon \mid SS$

- $L(S) \subseteq L$ $\varepsilon, ab, ba, abba, abab \cdots$

(Proved)

- Now P.T $L \subseteq L(S)$

Ext: $|w| = 2k$ $n_a(w) = n_b(w) = k$

case1: the first and last terminal is different.

case 1.1: 1st terminal = a $w = aw'b$ $w' \in L$

Case 1.2: 1st terminal = b $w = bw'a$ $w' \in L$

By inductive hypothesis w' can be generated by the grammar

$S \rightarrow aw'b \rightarrow w$

$S \rightarrow bw'a \rightarrow w$

case 2: 1st and last terminal are same

case 2.1: 1st & last are a: $w = a\ w'\ a$

$w' \notin L$: recursive production can't be used

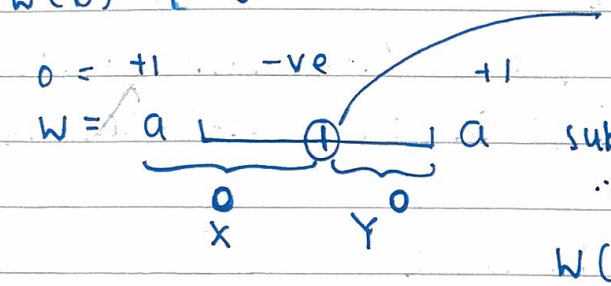
w can be divided into w_1 and w_2 such that

$w = xy$ where $x, y \in L$

weight of a $w(a) = 1$

weight of b $w(b) = -1$

} weight of whole string $= 0$



1st position where weight of substring becomes 0

$\therefore x \in L$

$w(w) = w(x) + w(y)$

$0 = 0 + w(y)$

$w(y) = 0 \Rightarrow y \in L$

$S \rightarrow SS$

$\downarrow \quad \downarrow$

$x \quad y$

$\therefore S \rightarrow asb \mid bsa \mid \varepsilon \mid ss$

starting variable never comes on RHS of production

Chomsky normal form (CNF)

A grammar is said to be in Chomsky Normal form if the productions are of the following form the same

$S \to AB$ where $S, A, B \in V$ & A & B are not S

$S \to a$ where a is a terminal / $a \in T$

In addition you have $S \to \varepsilon$ where S is the starting variable.

Q. $S \to ASA \mid aB$
$A \to B \mid S$
$B \to b \mid \varepsilon$

convert the following grammar to CNF

$B \to \varepsilon$ (ε productions)
$A \to B$ (unit productions)
$A \to S$ } contains starting
$S \to ASA$ } production on RHS (1)
$S \to ASA$ } more than 2 variables on RHS
$S \to aB$ } Mixed production (variable + terminal)

* Step1: create a new start symbol $S' \to S$ to take care of (1)

$S' \to S$
$S \to ASA \mid aB$ if the present one occurs on RHS of some production
$A \to S \mid B$
$B \to b \mid \varepsilon$

* **Step 2**: Eliminate all ε productions

$S^1 \to S$

$S \to ASA \mid aB \mid a$
$A \to B \mid S \mid \varepsilon$
$B \to b$

} replace the production $B \to \varepsilon$

$S^1 \to S$

$S \to ASA \mid AS \mid SA \mid S \mid aB \mid a$
$A \to B \mid S$
$B \to b$

} replace the produ... $A \to \varepsilon$

* **Step 3**: Eliminate unit productions

$S^1 \to S$

$S \to ASA \mid aB \mid a \mid AS \mid SA$
$A \to B \mid S$
$B \to b$

$S^1 \to ASA \mid aB \mid a \mid AS \mid SA$
$S \to ASA \mid aB \mid a \mid AS \mid SA$
$A \to b \mid ASA \mid aB \mid a \mid AS \mid SA$
$B \to b$

* **Step 4**: convert them to the form $S \to AB$ or $S \to a$

$C \to a \qquad B \to b$
$S^1 \to SA \mid AS \mid CB \mid ASA \mid a$
$A \to a \mid AS \mid SA \mid CB \mid b \mid ASA$
$S \to ASA \mid CB \mid a \mid AS \mid SA$

chomsky normal form

$C \to a \quad B \to b \quad D \to SA$

$S' \to SA \mid AS \mid CB \mid AD \mid a$

$A \to a \mid b \mid AS \mid SA \mid CB \mid AD$

$S \to AD \mid CB \mid a \mid AS \mid SA$

Q. $A \to BCDEFG$

$A \to BX$

$X \to CY$

$Y \to DZ$

$Z \to EW$

$W \to FG$

} CNF

Q $S \to aSb \mid bSa \mid SS \mid \varepsilon$

No. of productions required to get abba = 5

$S \Rightarrow SS$

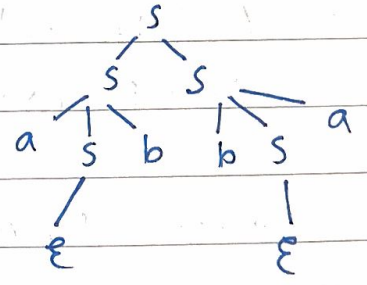
$\Rightarrow aSbS$

$\Rightarrow a\varepsilon bS$

$\Rightarrow abbSa$

$\Rightarrow abba$

} 5



Convert to CNF and find no. of productions to derive a string of length K.

7

$S \rightarrow aSb \mid bSa \mid SS \mid \varepsilon$

#1. $S' \rightarrow S$

$S \rightarrow aSb \mid bSa \mid SS \mid \varepsilon$

#2. $S' \rightarrow S \mid \varepsilon$

$S \rightarrow aSb \mid bSa \mid SS \mid ab \mid ba$

#3. $S' \rightarrow aSb \mid bSa \mid SS \mid ab \mid ba \mid \varepsilon$

$S \rightarrow aSb \mid bSa \mid SS \mid ab \mid ba$

#4. $A \rightarrow a$

$B \rightarrow b$

$S' \rightarrow ASB \mid BSA \mid SS \mid ab \mid ba \mid \varepsilon$

$S \rightarrow ASB \mid BSA \mid SS \mid ab \mid ba$

~~$C \rightarrow SA$~~ ~~~~ ~~~~ ~~~~

$D \rightarrow SB$

$A \rightarrow a$

$B \rightarrow b$

$S' \rightarrow AD \mid BC \mid SS \mid ab \mid ba \mid \varepsilon$

$S \rightarrow AD \mid BC \mid SS \mid ab \mid ba$

To produce abba

$S' \Rightarrow SS$

$\Rightarrow abS$

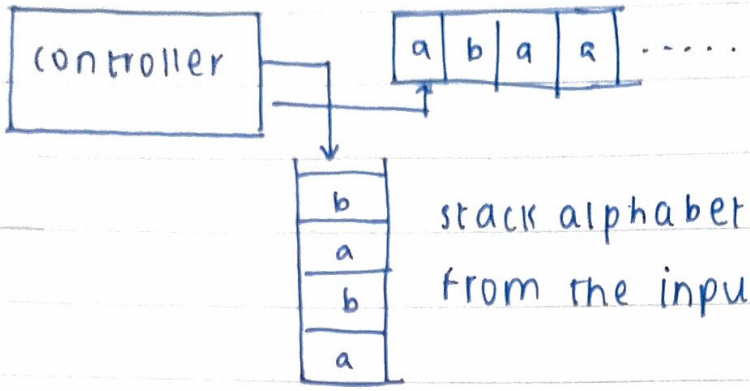
$\Rightarrow abba$

Q1. $A \rightarrow BAB \mid B \mid \varepsilon$
$B \rightarrow 00 \mid \varepsilon$

Q2. $R \rightarrow XRX \mid S$
$S \rightarrow aTb \mid bTa$
$T \rightarrow XTX \mid X \mid \varepsilon$
$X \rightarrow a \mid b$

Q If $W \in L$ is a string of length k and L has a grammar that is in CNF. How many productions are required to derive W? $2k-1$

PUSHDOWN AUTOMATA



stack alphabet can be different from the input alphabet

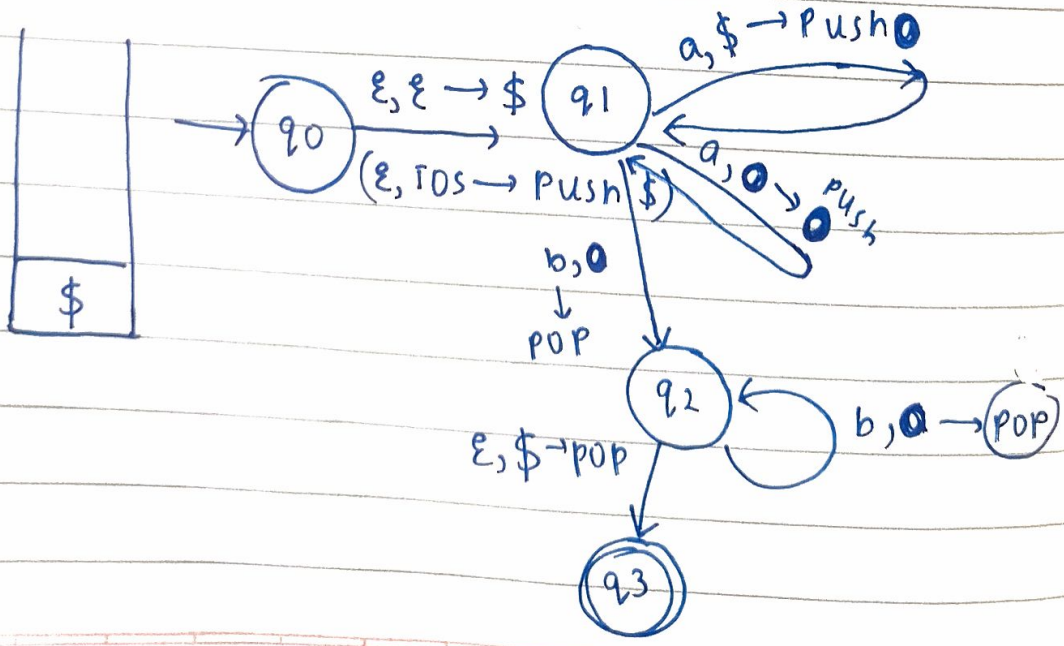
FSM definition : $(Q, \Sigma, \delta, q0, F)$ $\delta : Q \times \Sigma \rightarrow Q$

PDA definition : $(Q, \Sigma, \Gamma, \delta, q0, F)$ $\delta : Q \times \Sigma \times \Gamma \rightarrow Q \times \Gamma$

Γ : stack alphabet

Q. $L = \{ a^n b^n / n >= 0 \}$

Machine cannot detect bottom of stack. so before entering symbols in the stack, a special symbol is inserted first. used to check emptiness of stack. (Here $ is used)

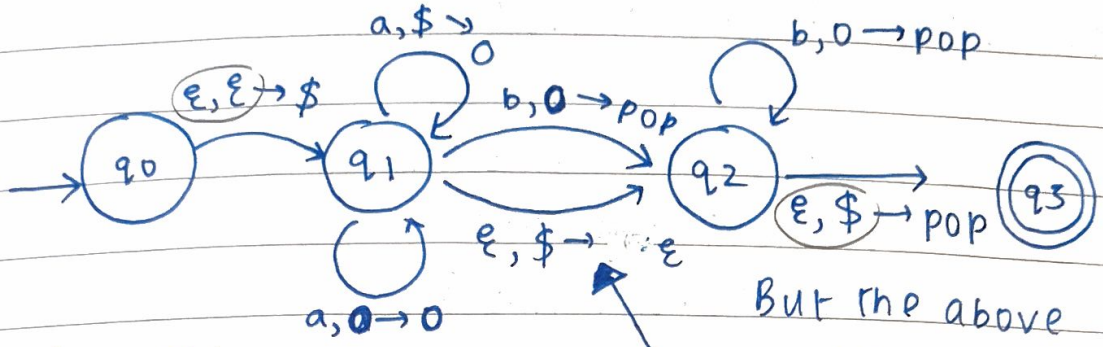


ε transitions are used to detect bottom of stack - does not imply nondeterminism

store 67

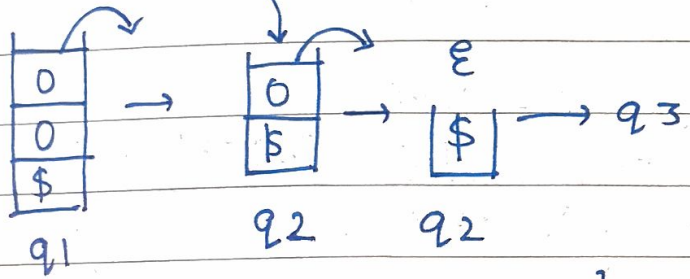
Example : compilers

Deterministic PDA ≠ Non deterministic PDA

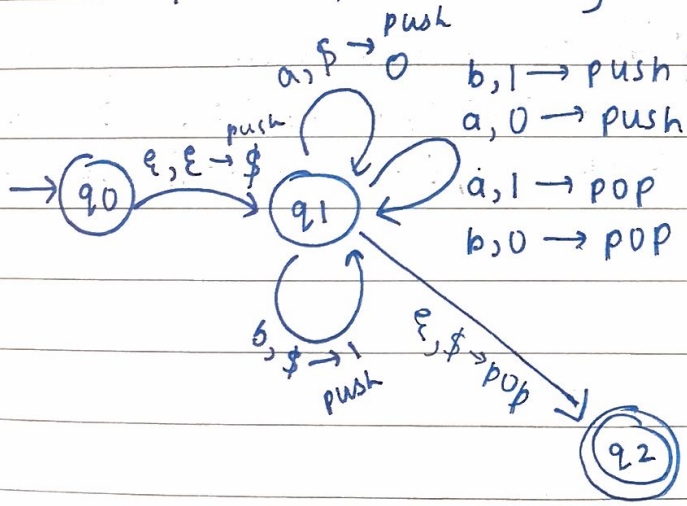


But the above machine rejects ε so include

$a^2b^2 = aabb$



Q2. $L = \{ w \mid n_a(w) = n_b(w) \}$ $\Sigma = \{a, b\}$ $\Gamma = \{0, 1\}$



accepts ε

Two stacks allows

- random access
- provides unlimited memory

2 stacks ≠ 1 stack

(}

Q3. $L = \{$ All balanced paranthesis $\}$

Q4. $L = \{ WW^R , W \in \{0,1\}^* \}$

Q5. $L = \{ w \mid n_a(w) \neq n_b(w) \}$

Q6. $L = \{ a^i b^j c^k ; i=j \text{ or } j=k \}$

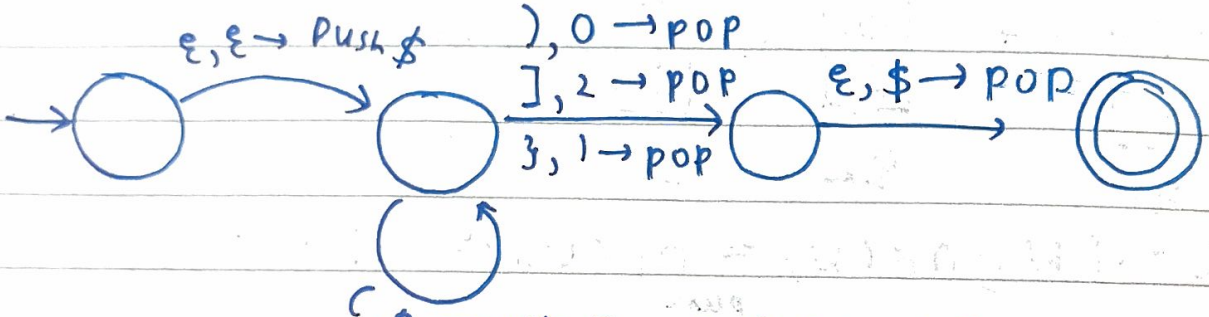
Q7. $L = \{ a^n b^m ; n \neq m \}$

Q8. $L = \{ a^n b^m ; n \leq m+2 \}$

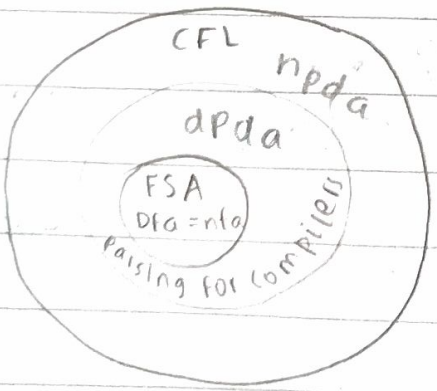
Q9. $L = \{ a^n b^m c^k ; k = n+m \}$

$\Sigma = \{ (, \{, [,], \},) \}$ $\quad \Gamma = \{\$, 0, 1, 2\}$

A3.



(})

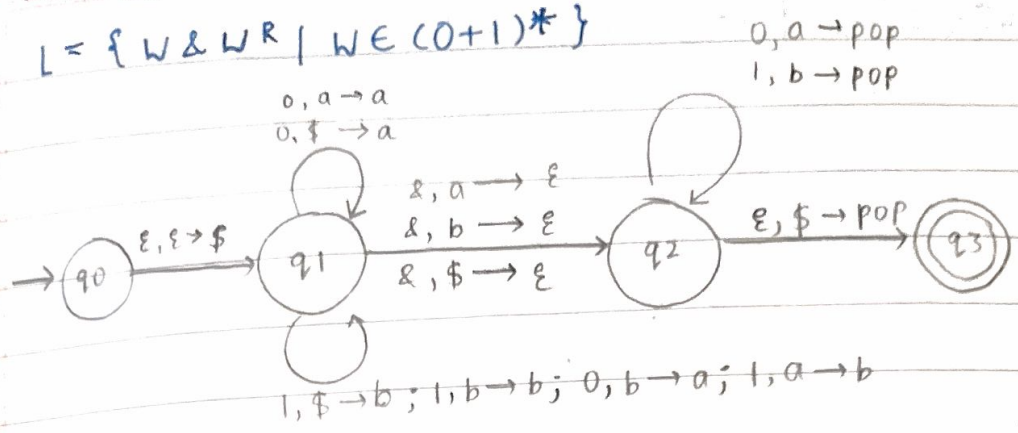


middle of the string is known

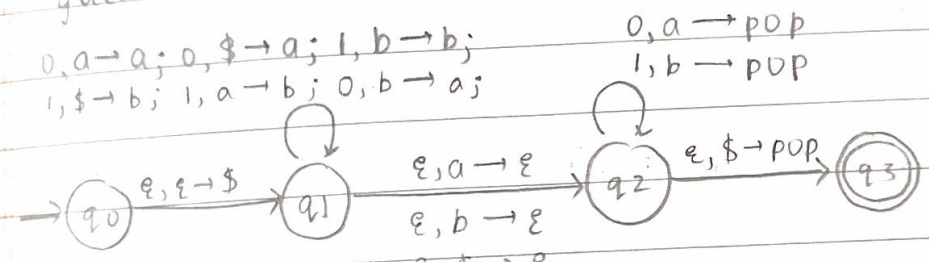
store 67

deterministic pushed down automata

Q $L = \{ W \& W^R \mid W \in (0+1)^* \}$



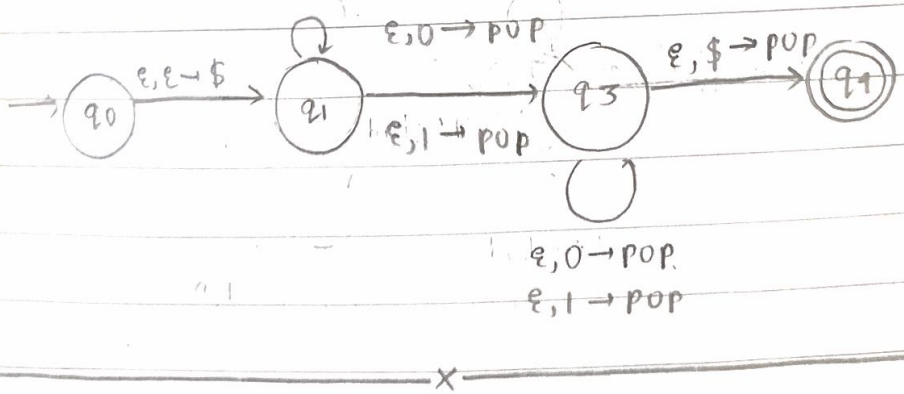
A4. $\{WW^R \mid W \in (0+1)^*\}$ - middle of string needs to be guessed - non-determinism is required.



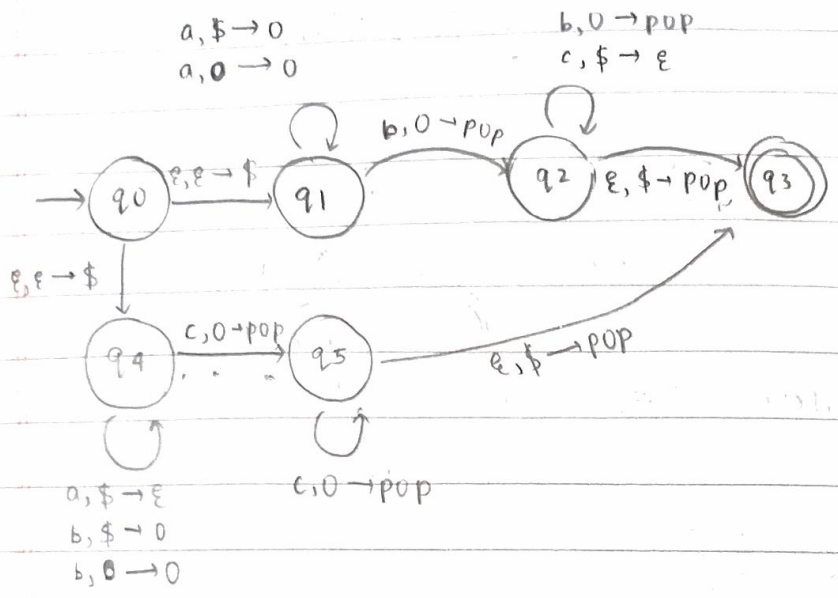
0 ; C, 2 → push 0 ;
1, 2, push 1 ;
, 2 → push 2 ;

A5. $L = \{W \mid n_a(W) \neq n_b(W)\}$

a, $ → 0 ; a, 0 → 0 ; b, $ → 1 ; b, 1 → 1 ; b, 0 → pop ; a, 1 → pop

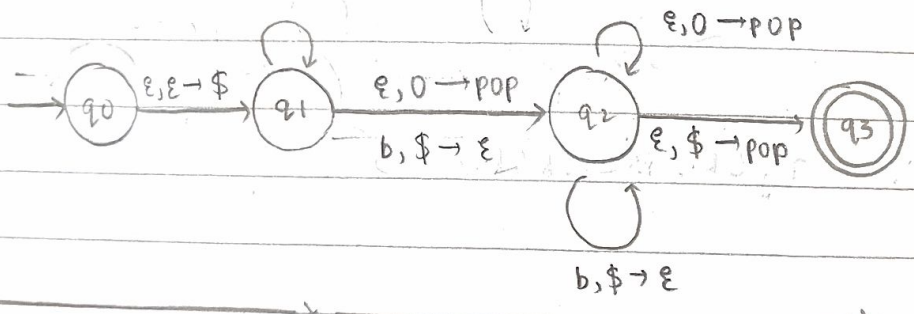


A6. $L = \{a^i b^j c^k \mid i = j \text{ or } j = k\}$

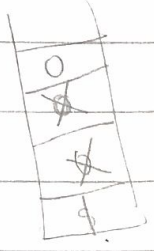


A7. $L = \{a^n b^m \mid n \neq m\}$ $n < m$ $n > m$

$a, \$ \to 0 ; a, 0 \to 0 ; b, 0 \to pop$



A8. $L = \{a^n b^m ; n <= m+2$



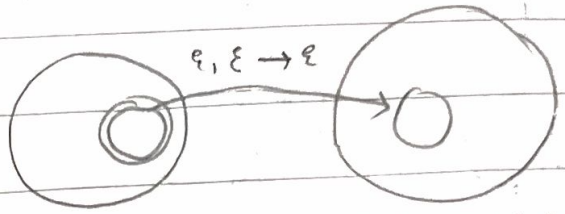
store
67

CONTEXT FREE LANGUAGES

CLOSURE PROPERTIES

#1. Union : L_1 starting symbol S_1
L_2 starting symbol S_2
$L = L_1 \cup L_2$ $\quad S \rightarrow S_1 / S_2$

#2. concatenation :



$L_1 \cdot L_2$
After terminals of 1 language are generated - second start

#3. L_1 S_1 $\quad L_1^*$ star
$S \rightarrow SS \mid \varepsilon$

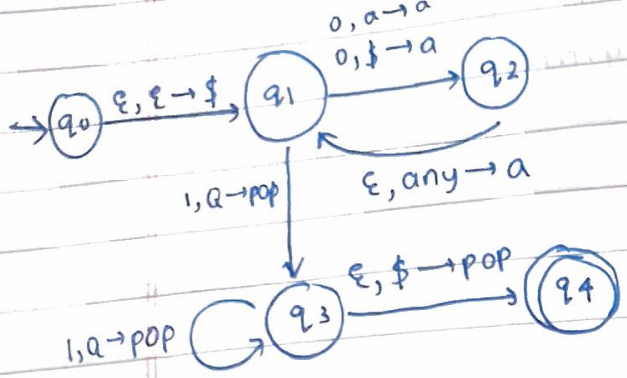
#4. Intersection
$L_1 = \{ a^n b^m c^k : n = m \}$
$L_2 = \{ a^n b^m c^k : m = k \}$
$L_1 \cap L_2 = \{ a^n b^m c^k : n = m \ \& \& \ m = k \ \forall \ n \geq 0 \}$
$= \{ a^n b^n c^n \mid n \geq 0 \}$ – not context free

#5. complementation – Not closed under intersection
$\Rightarrow$ not closed under complementation

DPDA – closed under complementation
$\Rightarrow$ not closed under intersection
$\Rightarrow$ not closed under union

NPDA – closed under union
not closed under complementation
not closed under intersection

Q. $L = \{0^n 1^{2n} \mid n \geq 0\}$



Q. 01, 0011, 011, 001111

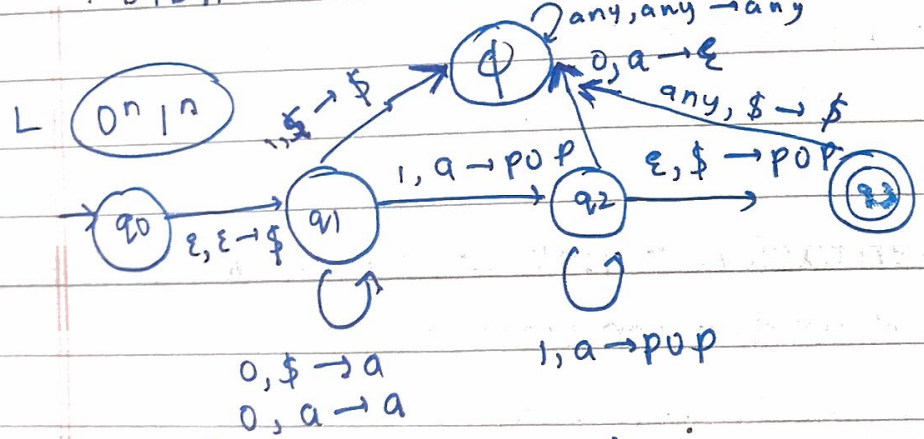
$L_1 = \{0^n 1^n \mid n \geq 0\}$

$L_2 = \{0^n 1^{2n} \mid n \geq 0\}$

$L = L_1 \cup L_2$

DPDA is not possible - needs to make a choice whether to push 1 or 2 0's

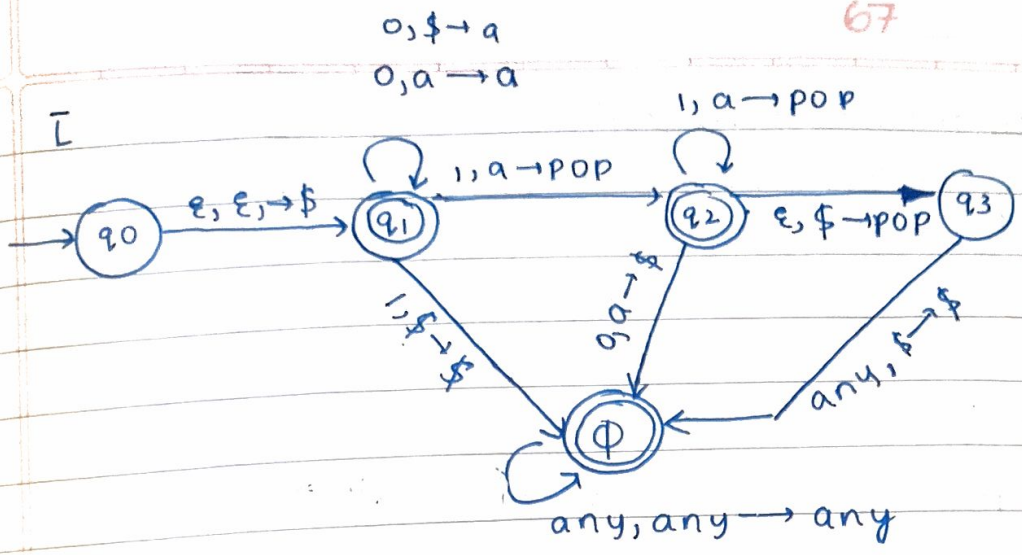
∴ DPDA - not closed under union



$\bar{L}$: $\{0^n 1^m \mid m \neq n\}$

U : {strings having 10 as substring}, {ε}

store
67



Q. $L = \{WW^R \mid W \in (0+1)^*\}$ — NDPDA

$\bar{L} = \{\overline{WW^R} \mid W \in (0+1)^*\}$ — not closed under complementation

↳ length of string is odd

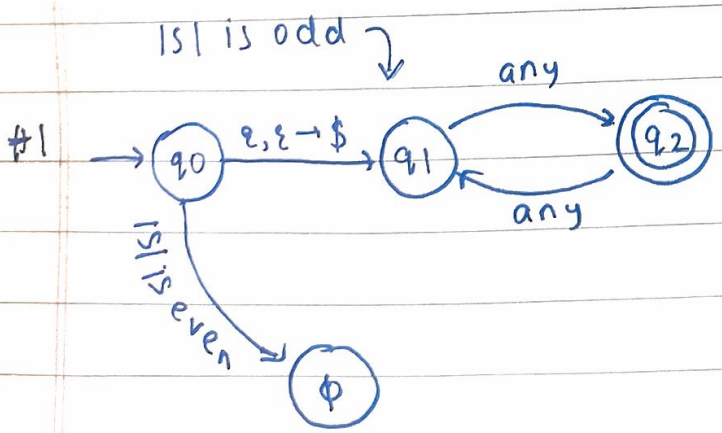
↳ ~~Kth from 1st string~~

Palindrome definition : $a[i] = a[2n-i-1]$

Not an even palindrome if –

#1 – length of string is odd

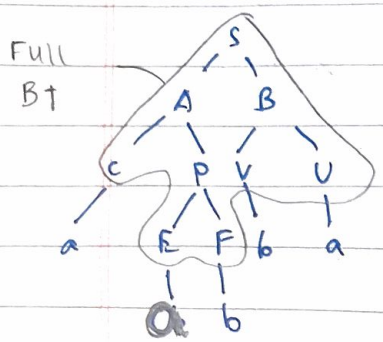
#2 – ∃ some i such that $a[i] \neq a[2n-i-1]$



A grammar is in CNF

How many prods are required to derive w

IDOR at parse tree (Binary)

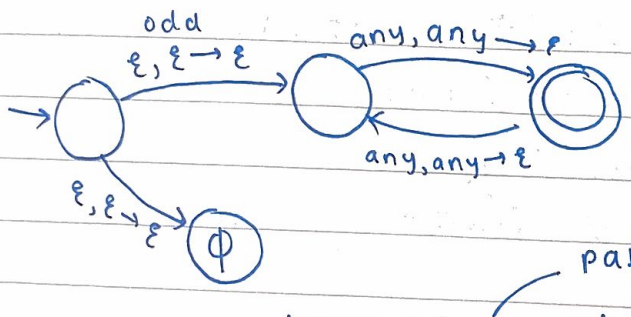


K = NO. OF leaves

K + NO. of internal nodes

K + K-1

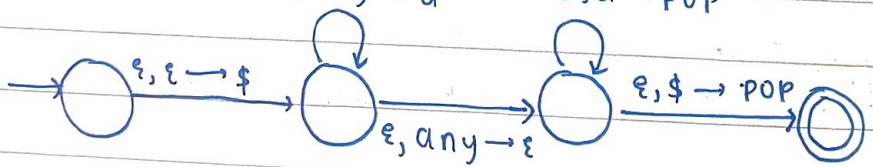
= 2K-1



$L = \overline{WW^R}$

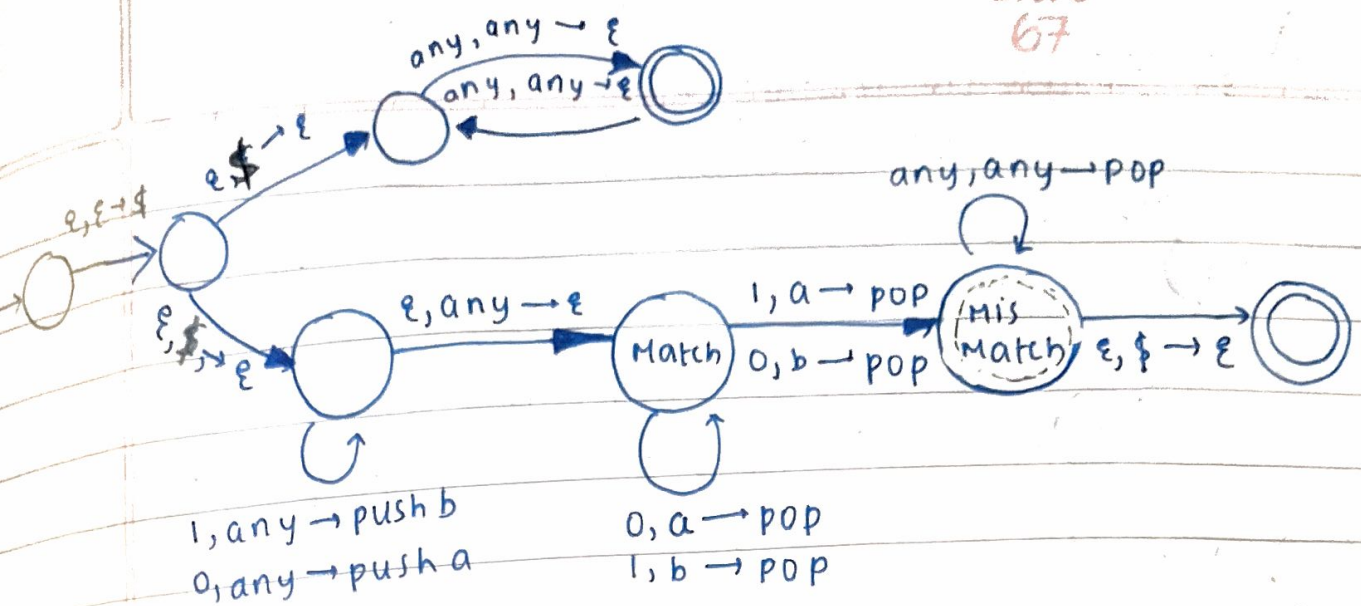
← odd length

even palindrome



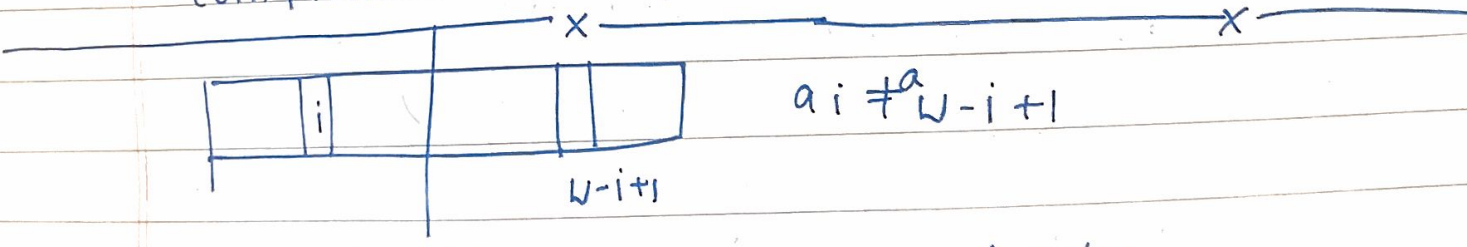
X - to show power of non determinism

store
67



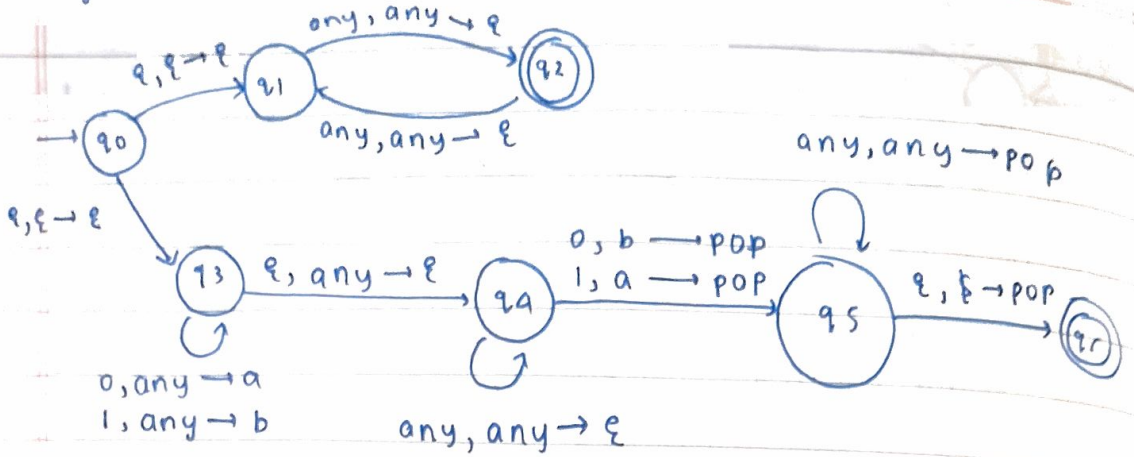
But this machine will accept 1001
Push b. Now 0,b will take you to mismatch from which rest of the stack will be popped.
Should not accept 1001
so don't make mismatch a final state & create an additional final state. This ensures that matching occurs only after pushing half the string into stack

WW^R is a CFL - npda } not true for all CFLs
complement is a npda }



Normal palindrome - not even or odd

guess where the mismatch is

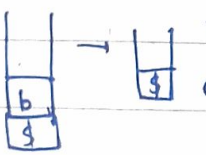


q4: Guessing 1st occurrence of i whether it is the w-i+1 th character or not

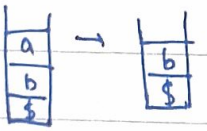
q5: check if guess is correct

The ith character is on top of the stack

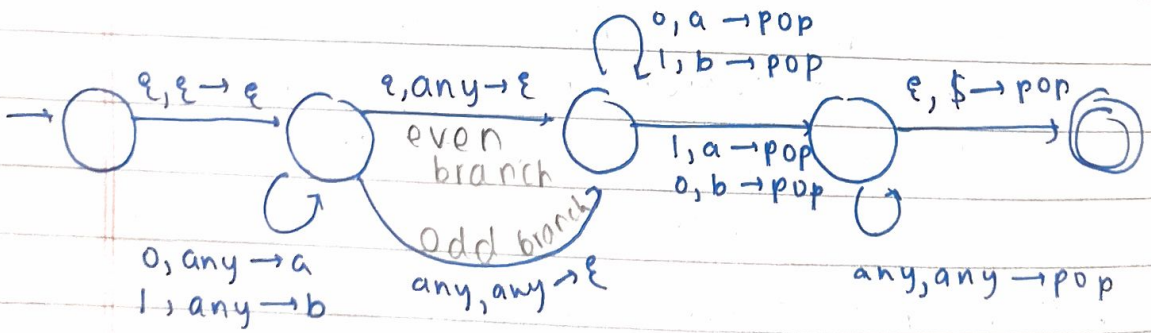
101 :



not accepted
dead state



not accepted

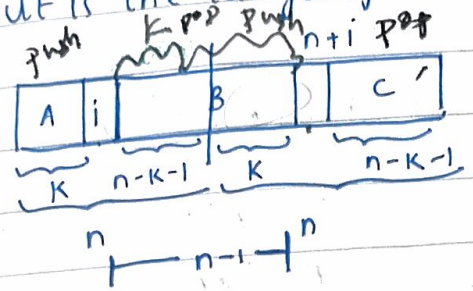


k+1

store
67

Q. P.T the language $L = \{WW\}$ is not a CFL

But is the language $\bar{L}$ a CFL or NOT ? (Ans: YES)



- odd length
- $a_i \neq a_{\frac{w}{2}+i}$

Q. $\{0^n 1^m 0^K \mid m = n+K\}$

Push all 0

Pop all 0's till stack becomes empty

Push all 1's (remaining)

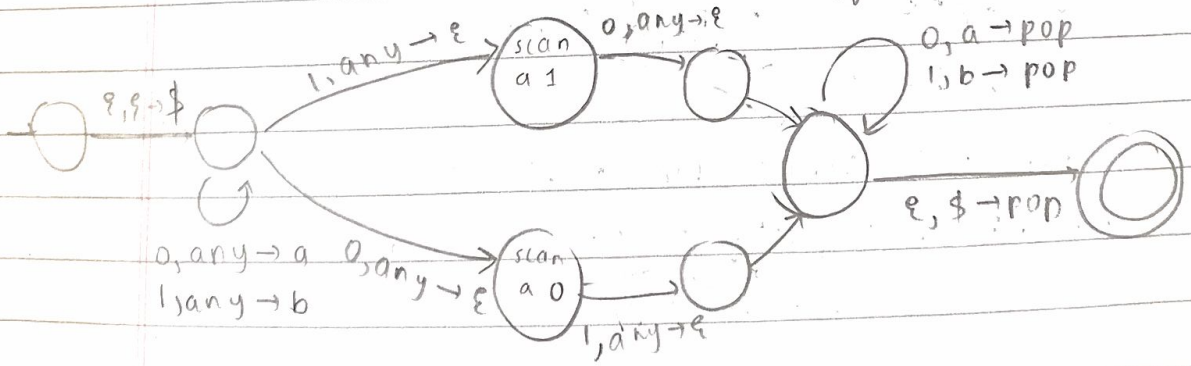
POP all 1's till no input is left

$|A+C| = K + n - K - 1 = n-1$

$|B| = n - K - 1 + K = n-1$

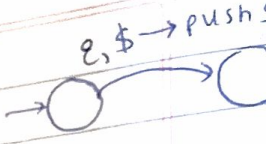
till i push into stack

at i go to another state (different for 1 & 0)



Context free languages $\Leftrightarrow$ Npda

CFG $\Leftrightarrow$ npda



$S \to aSb \mid ab$ $\quad L = \{a^n b^n \mid n \geq 1\}$

$S \to ASB \mid AB$

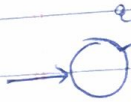
$A \to a$

$B \to b$

$S \to ASB$

$A \to aA$

$B \to bB$



$S' \to AC \mid AB$

$S \to AC \mid AB$ chomskey Normal form

$C \to SB$

$A \to a$

$B \to b$

Suppose we want to construct aabb

$S' \Rightarrow AC$

$\Rightarrow aC$

$\Rightarrow aSB$

$\Rightarrow aABB$

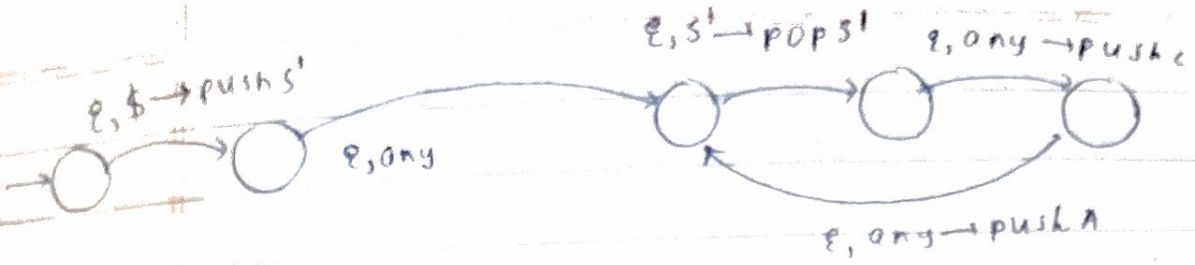
$\Rightarrow aabb$

$2k-1$ productions

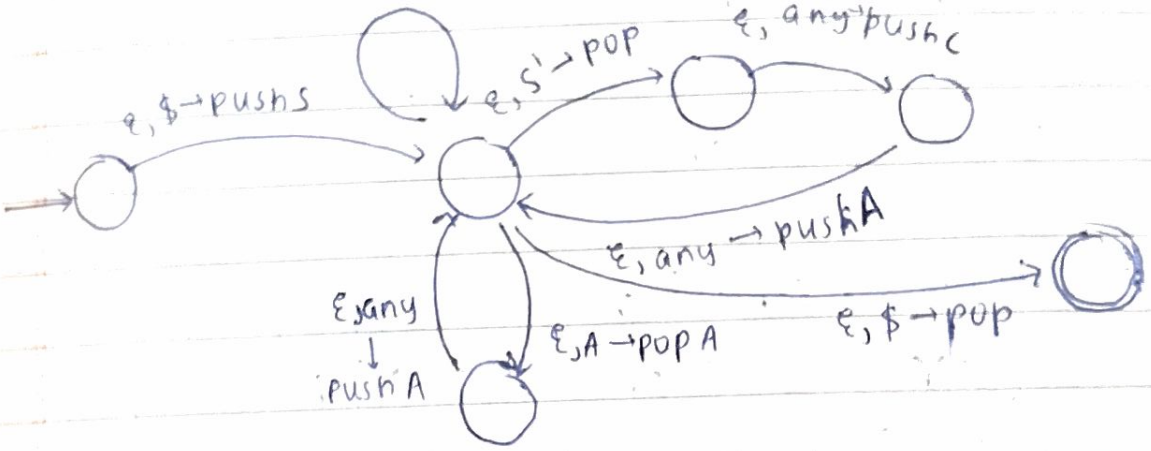
$= 2 \times 4 - 1$

$= 7$ productions

Q. n variables. What should be the length of the string such that in a parse tree, ∃ one path from root to leaf where a variable is repeated twice.



ε, S → push A ε, C → pop C
ε, S → push B
ε, S → push C
ε, S → pop S



TUTORIAL 4

Q1. If L1 is a CFL and L2 is a regular language
a) then is $L1 \cap L2$ regular ?
b) then is $L1 \cap L2$ CFL ?
c) $L1 - L2$?

Ans 1a) Let $L1 = \{a^n b^n \mid n >= 0\}$

$L2 = \{a^n b^m \mid n, m >= 0\}$

L1 is CFL and L2 is regular

$L1 \cap L2 = \{a^n b^n \mid n \geq 0\}$ which is not regular

$\therefore L1 \cap L2$ is not regular

Q2. $L = \{X \# Y \mid X, Y \in (0+1)^*$ and $X \neq Y\}$ P.T L is a CFL

Q3. $L = \{XY \mid X, Y \in (0+1)^*$ and $|X| \doteq |Y|$ but $X \neq Y\}$ — WW
Prove that L is a context free language

Q4. $L = \{a^i b^j \mid j \neq i$ and $2i \neq j\}$ show that L is a CFL WRONG

Q5. $L = \{0^n \mid m\, 0^n \mid n, m \geq 0\}$. show that L is a CFL

Q6. Convert to chomskey Normal form

$S \rightarrow aAa \mid bBb \mid \varepsilon$

$A \rightarrow C \mid a$

$B \rightarrow C \mid b$

$C \rightarrow CDE \mid \varepsilon$

$D \rightarrow A \mid B \mid ab$

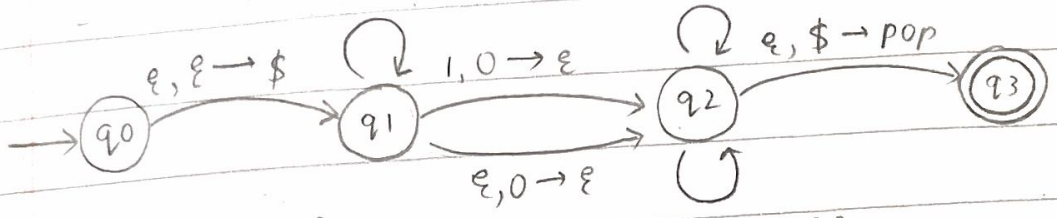
Q7. Let A and B be languages.
Let $A \square B = \{XY \mid X \in A$ and $Y \in B$ and $|X| = |Y|\}$. If A and B are regular show that $A \square B$ is a CFL

store
67

pdn

Q8. Give CFG and production for the language
$L = \{ w \mid |w| \text{ is odd and the middle symbol is } 0 \}$

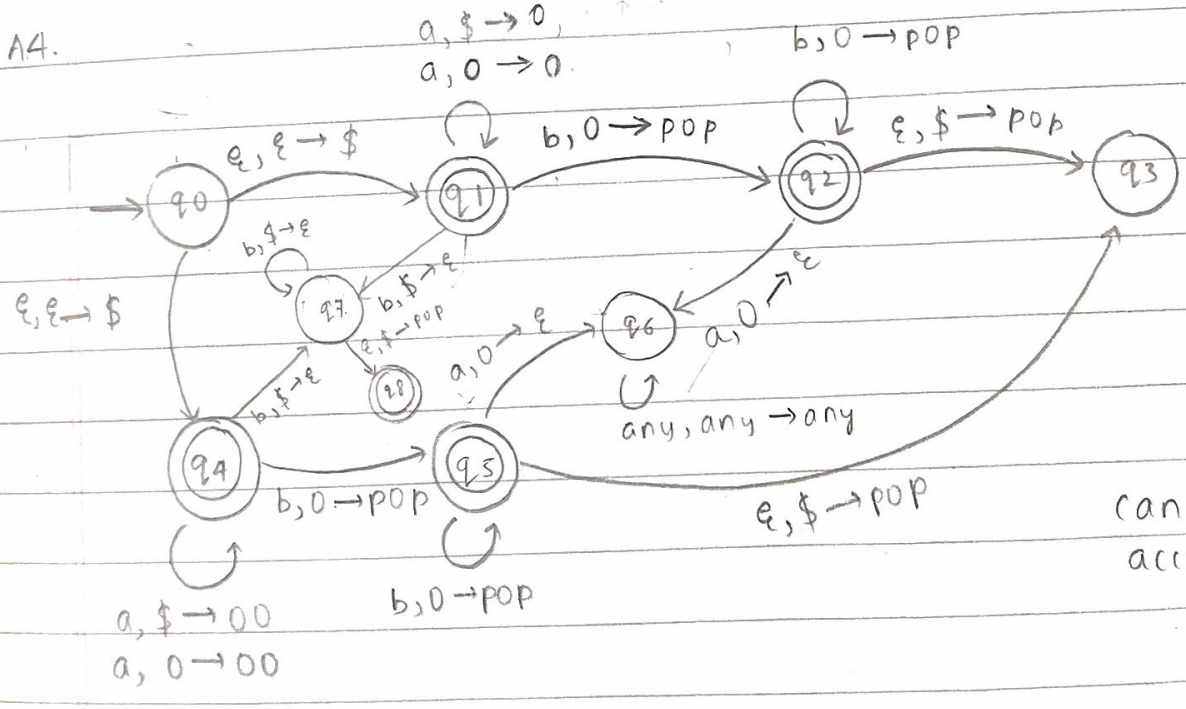
A5.



PRODUCTIONS
$S \rightarrow 0S0 \mid 0A0 \mid A \mid \varepsilon$
$A \rightarrow 1A \mid \varepsilon$

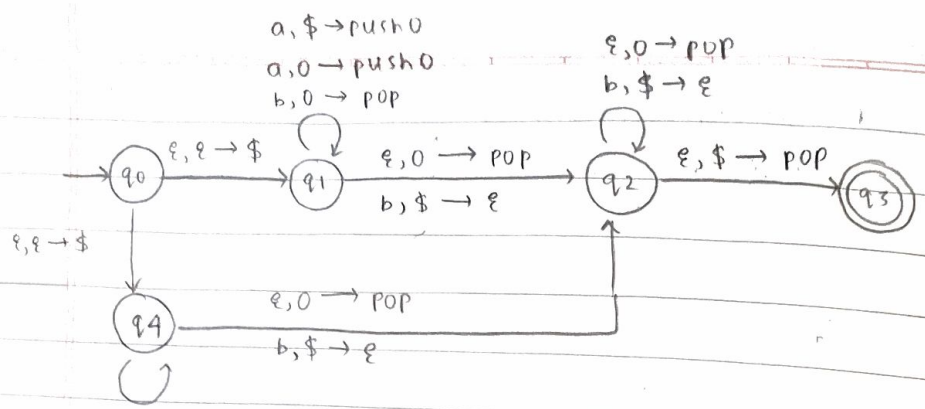
since we can construct a pushed down automaton for L (nondeterministic), L is a CFL

A4.



can it accept b

aaabb
a a bbb



A6. S →
A →
B →
C →
D →

S →
A
B
C
D

A8. S → X0X | Y0Y

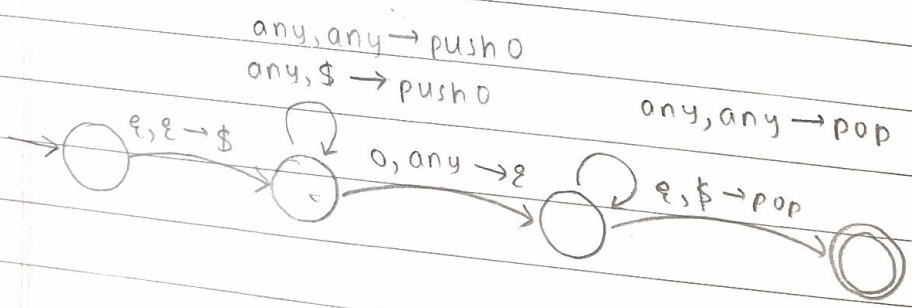
X → BX | ε

B → 00 | 01 | 10 | 11

Y → 0X | 1X | X0 | X1 | X0X | X1X

X gives productions of even length
Y gives productions of odd length

Will not account for 0 to be in the middle



store

Eliminate all ε transitions

A6.

$S \rightarrow aAa \mid bBb \mid \varepsilon$	$S \rightarrow aa \mid aAa \mid bb \mid bBb \mid \varepsilon$
$A \rightarrow \varepsilon \mid a \mid C$	$A \rightarrow a \mid C$
$B \rightarrow \varepsilon \mid b \mid C$	$B \rightarrow b \mid C$
$C \rightarrow CD \mid D$	$C \rightarrow CD \mid D$
$D \rightarrow A \mid B \mid ab$	$D \rightarrow A \mid B \mid \varepsilon \mid ab$

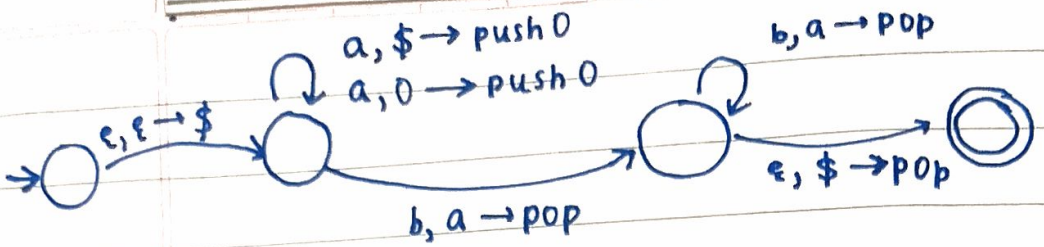
Eliminate unit prods

$S \rightarrow aa \mid bb \mid aAa \mid bBb \mid \varepsilon$	$S \rightarrow aa \mid bb \mid aAa \mid bBb \mid \varepsilon$
$A \rightarrow a \mid C$	
$B \rightarrow b \mid C$	
$C \rightarrow C \mid CD$	
$D \rightarrow A \mid B \mid ab$	

length

length

PUMPING LEMMA FOR CONTEXT FREE LANGUAGES



$a^n b^n$

aa(a)(b)bb ∈ L

y z

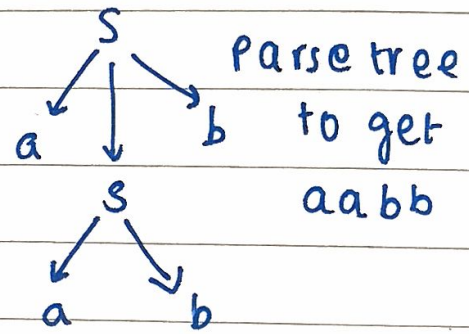
$y^i (w) z^i$

↳ can be ε or any other value

$a^n \# b^n$ | difficult to detect usable loops using PDA. so instead detect using the grammar

Q1) $S \rightarrow aSb \mid ab$

$a^n b^n$



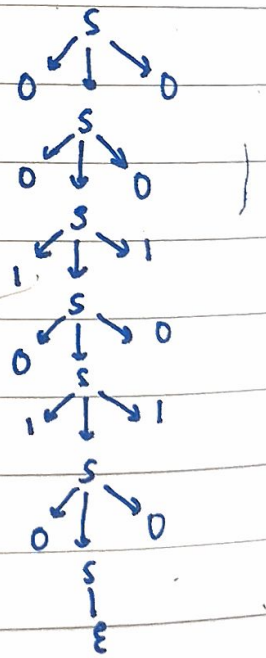
Parse tree to get aabb

Q2) Palindrome (even length)

$S \rightarrow 0S0 \mid 1S1 \mid \varepsilon$

string: 00101 00|0100

Parse tree:

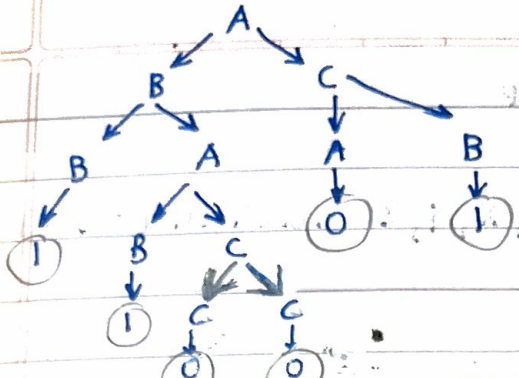


Q3) $A \rightarrow BC \mid 0$

$B \rightarrow BA \mid 1$

$C \rightarrow AB \mid 0 \mid CC$

string : 11000 1



grammar accepts 110001 and (11.00)(11.00)(1) (y) (z)

$y = 1100 \quad w = 0$

$z = 1$

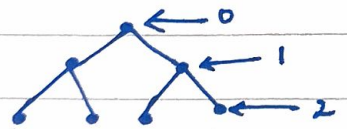
$y^2 w z^2 = 1100\ 1100\ 0\ 11$

Path has length > no. of variables $\therefore |s| \geq n+1$

What should be the no. of leaf nodes in the CNF parse tree so that the above condition is satisfied

i.e

What should be the size of L such that ∃ a path from root to leaf of length n.



Path of length n if no. of nodes = Nodes

$1 + 2^{n-1} \leq \text{Nodes} \leq 2^n$

corresponding push and pop operations are pumped up/down

L is a context free language

$\exists$ P (pumping length)

$\forall$ S with $|S| \geq P$, $\exists V, W, X, Y, Z$ such that $S = VUXYZ$

(i) $|XYZ| \leq P$

(ii) $|WY| > 0$

(iii) $VW^iXY^iZ \in L \quad \forall\ i \geq 0$

PUMPING LEMMA FOR CFL —

Let L be a CFL. Then there exists an integer (called the pumping length) P, such that every string $S \in L$ and $|S| \geq P$ has a decomposition $S = VWXYZ$ obeying the following conditions

(i) $|WXY| \leq P$

(ii) $|WY| > 0$

(iii) $VW^iXY^iZ \in L \quad \forall\ i \geq 0$

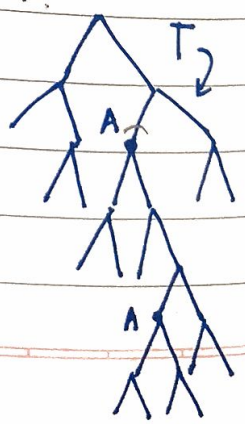
PROOF :

Assumption : The grammar for L is in CNF with n variables.

Let $P = 2^{n+4}$

Take any $S \in L$ such that $|S| \geq P$

Consider the parse tree T, for S



NO. of leaves in the tree $\geq 2^{n+4}$

∵ CNF

⇒ since the tree is binary we can infer that the tree has depth ≥ n+3

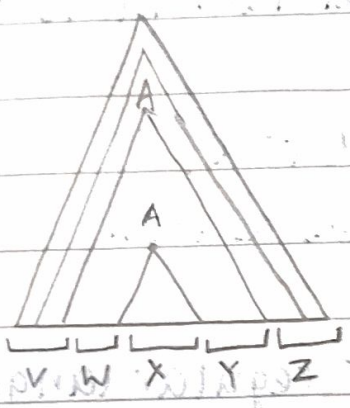
depth (T) ≥ n+3

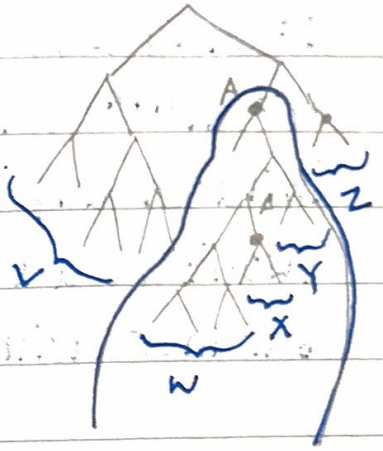
(iii)

⇒ ∃ a path from root to leaf that contains ≥ n+2 vari

(By pigeonhole principle) ↓

⇒ ∃ a variable that is repeated





since |s| > 0, the parse tree will not have a ε producti

as grammar is in CNF

↳ NO. of leaves = |s| ≥ 2^{n+4}

(ii)

Let A be the repeated variable. Take the first occurrenc of node α. The node α has 2 children. only one of them will lead to the second occurrence of A.

Let β be the other child. If β is the left child then w ≥ string produced by the β branch ≥ 1

If β is to the right, then y ≥ string produced by the β branch ≥ 1

⇒ either |w| > 0 or |y| > 0 ⇒ |wy| > 0

(i) Pick the repeating variable starting from the bottom of the parse tree using the deepest leaf.

⇒ Height of subtree rooted at first occurrence $\leq n+2$

⇒ No. of leaves in that subtree $\leq 2^{n+2}$

If language is not in CNF

$P = K^{n+4}$ where K is the maximum number of children any node produces.

(Won't be a binary tree) — parse tree.

But there will be ε productions as well

Q. Prove the pumping lemma for regular languages using grammars instead of FSMs.

HINT: Parse tree.

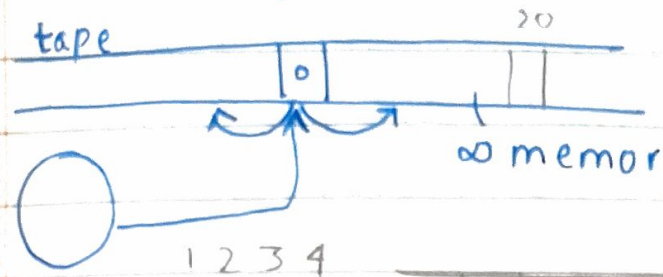
left / right linear grammars

Q1.

TURING MACHINES

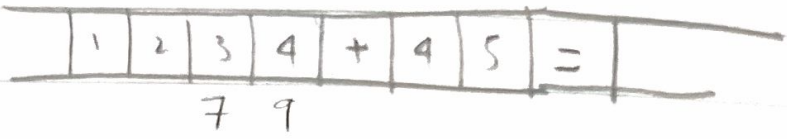
store 67

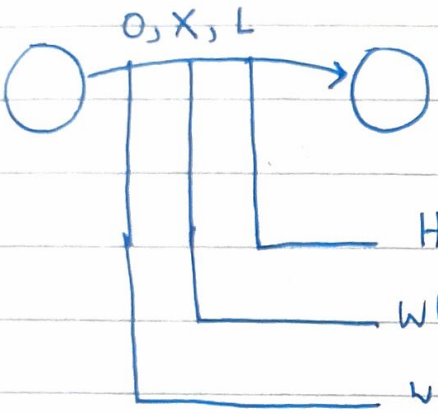
Alan Turing (1936)



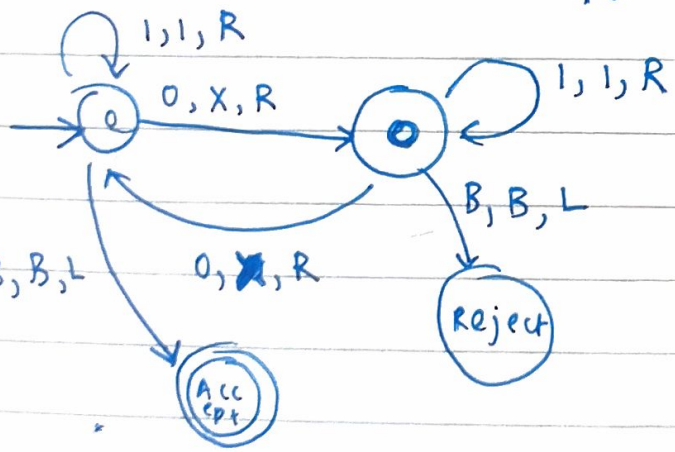
Can read/write?
move left/right?

1 2 3 4
+ 4 5
1 2 7 9



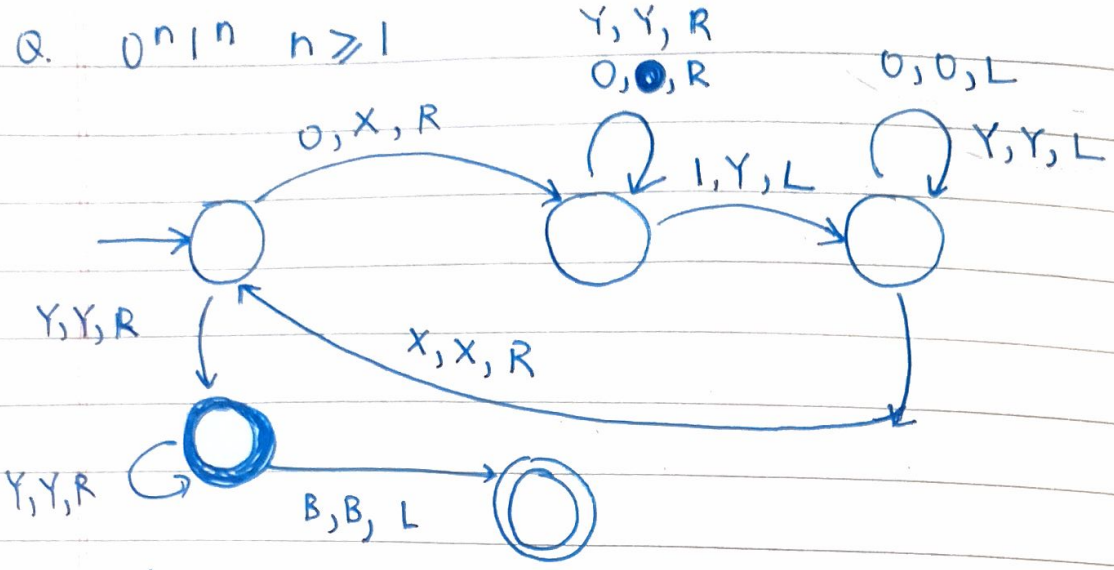


Q1. No. of zeroes is even other than ε



B = Blank space
X - What we write in state

Q. $0^n 1^n \quad n \geq 1$

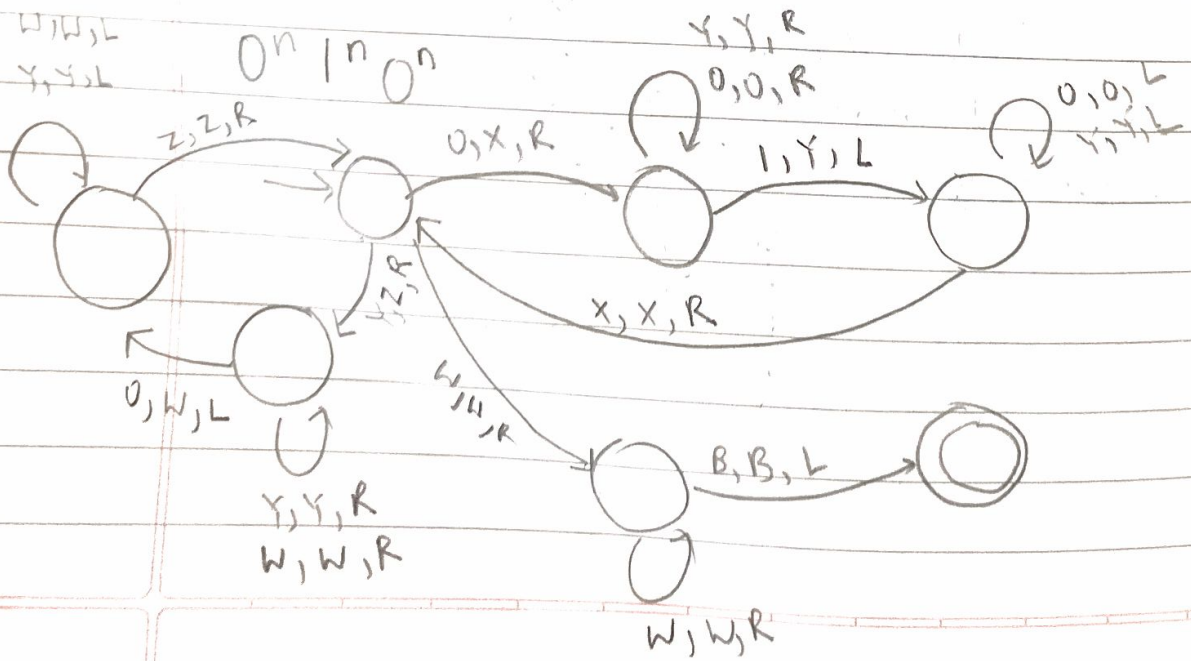


Input is
To check if a no. is prime (PTYPE)
input size = $\log n$
Algorithm : $O(\sqrt{n})$
exponential algorithm

given by Indian scientist

nth fibonnacci no. - exponential (P TYPE)
output requires n bits

$0^n 1^n 0^n$



store
67

7

Q1 W#W

Q2 WW

A1. 1. Read the symbol, remember it and mark it.

2. Go to the 1st unmarked symbol to the right of #

3. If the first & second symbol are different, reject

4. Else mark the second symbol and go back to the first unmarked symbol to the left of #

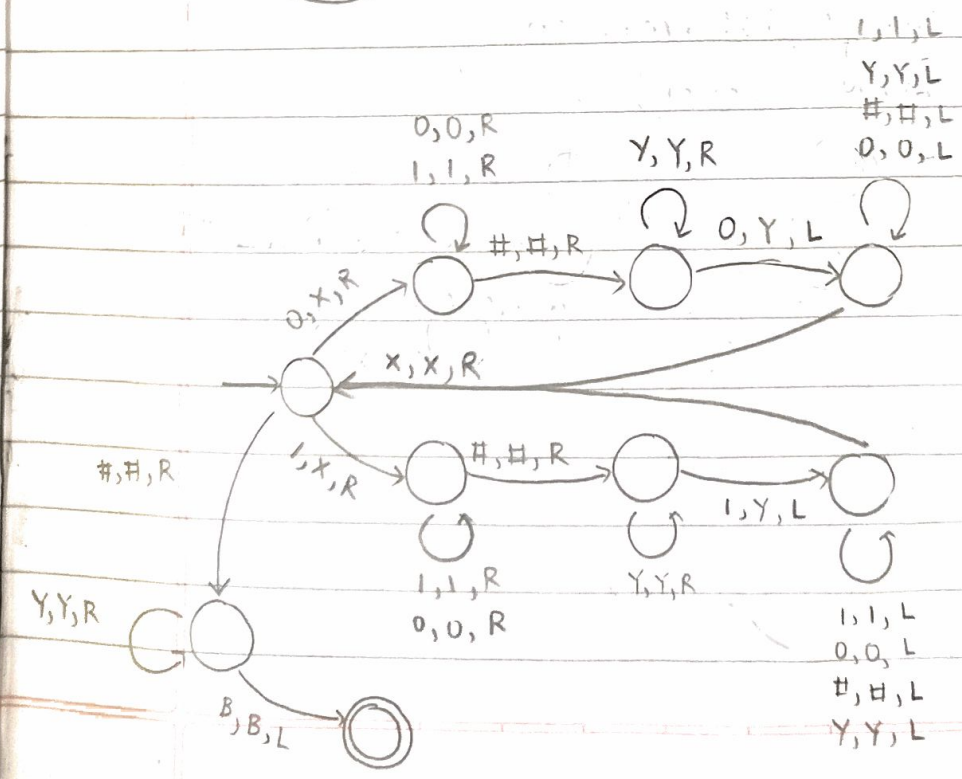
~~5. Go back to step 1~~

5. If all symbols to the left of # are marked then move to the right of # and check if all symbols are marked. If yes then accept. Else reject

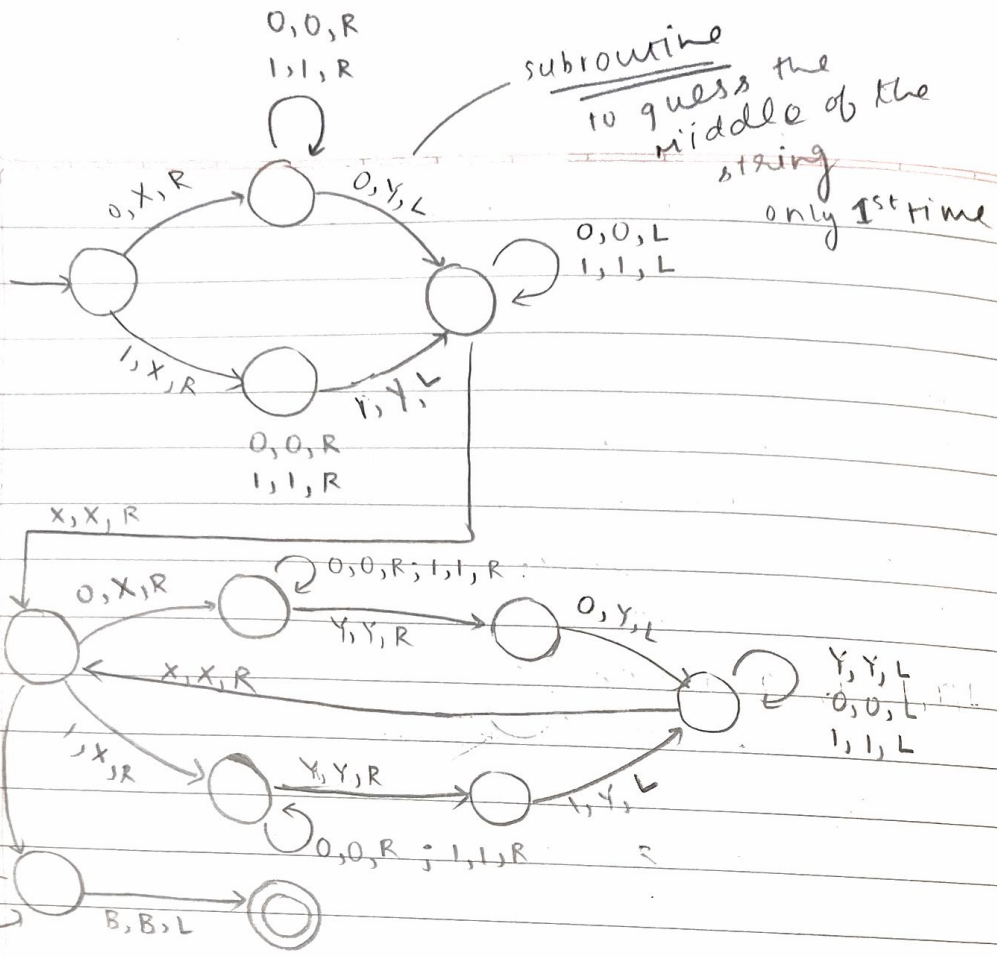
6. Else go back to step 1

~~0 0 X X # 0 0 X X~~

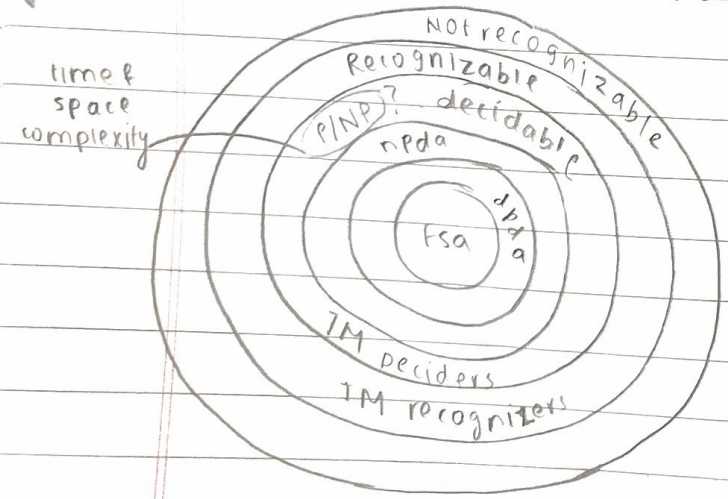
X X X X # Y Y Y Y



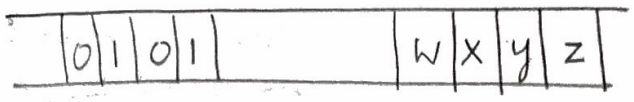
A2.



In Turing machines, deterministic machines are equivalent to non-deterministic machines



How to convert a non-deterministic TM into a deterministic TM ?



otherwise

w = 0 from beg
x = 1 from beg
y = 0 from end
z = 1 from end

convert WW to W#W — for loop

2 types of TMs —

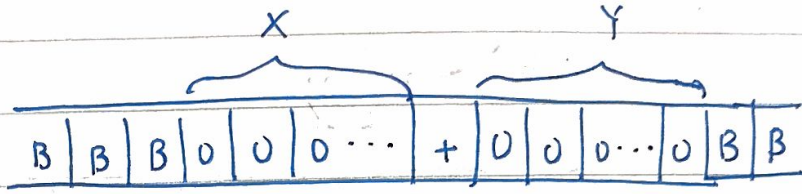
(i) Acceptors (ii) Transducers

Addition

Unary encoding —

X → 0000 ···· (x times)

Y → 0000 ··· (Y times)



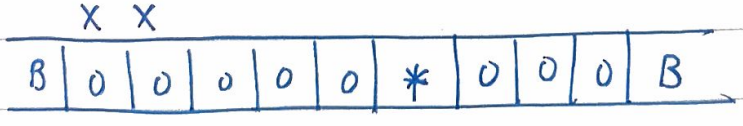
(i) Remove + & concatenate
go till the first + replace with 0. Let all 0's to the right remain same until the last zero is encountered - make it blank

(ii) start copying
↳ 1st zero make it x
move to last 0
next blank - make 0
Go back to X, move right to next 0 and once again copy to end of string until + is reached.

Multiplication

|AddY||AddY|

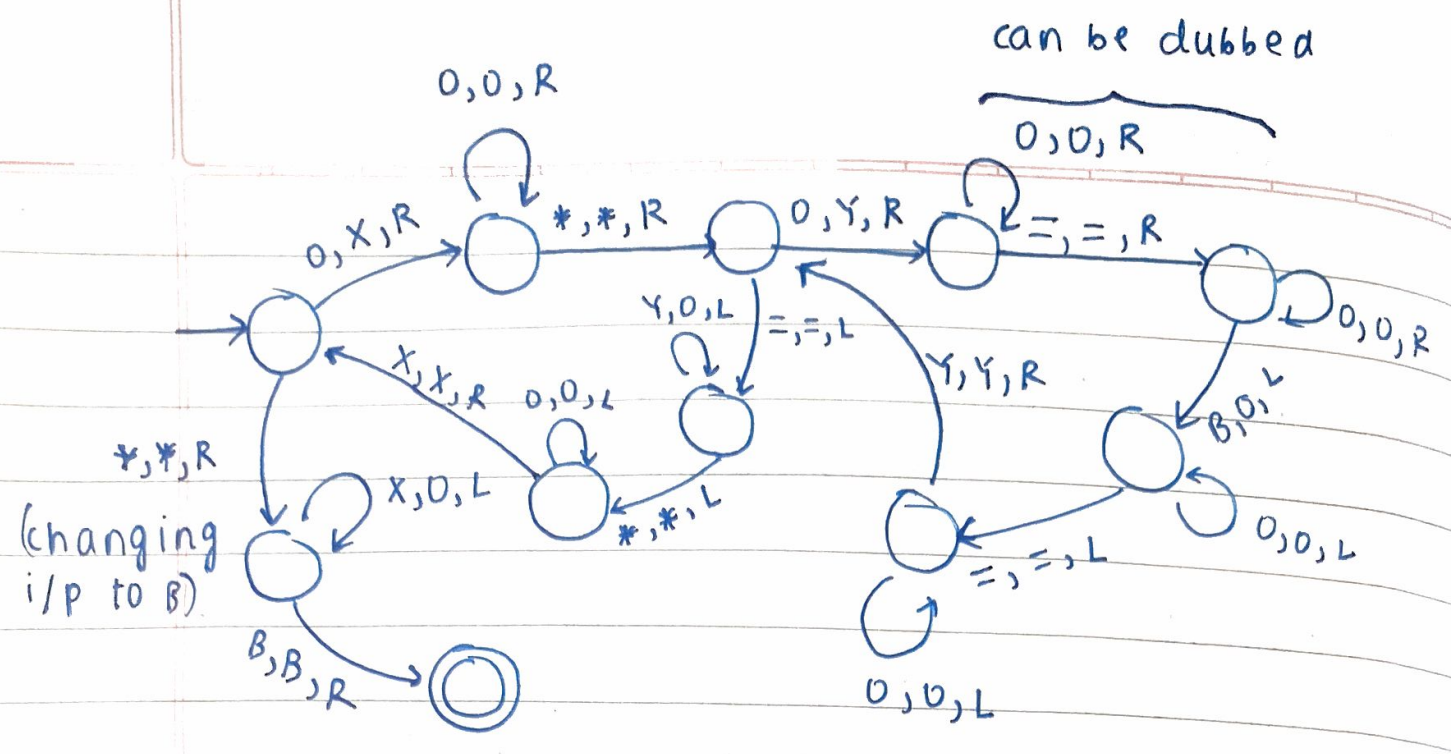
X * Y

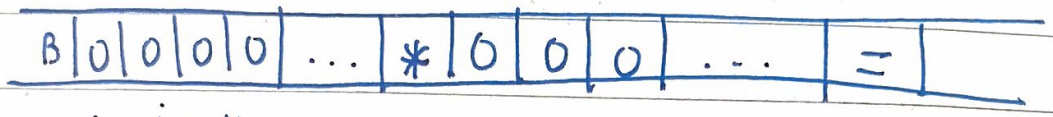


Add X → Y no. of times or vice versa

(1) For each 'x' in the 1st part copy the 2nd number to the right of the present output

$a^i b^j c^k \mid i*j = k$





$0^i 0^j 0^k$ $K = i * j$ regular / context free ?

Regular if $|w|$ = odd ($i = 1$, remove 1 zero

$j = k\left(\frac{j+R}{2}\right)$

Q. 2^n zeroes

Q. Generate nth fibonaccino.

Q. Check whether a no. is a fibonaccino or no

Q. Check whether a no. is prime or not

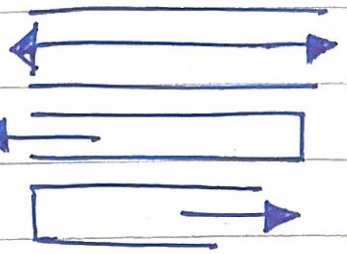
Brussels paradox

read up

VARIANTS OF TURING MACHINES

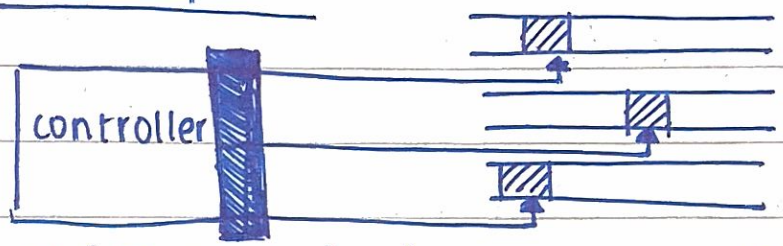
1) 3 operations are possible on the tape —
   - move right
   - move left
   - stay at current location (go left/right & comeback)

2) Normal TM : 2 way ∞
   variant : 1 way ∞ : L / R



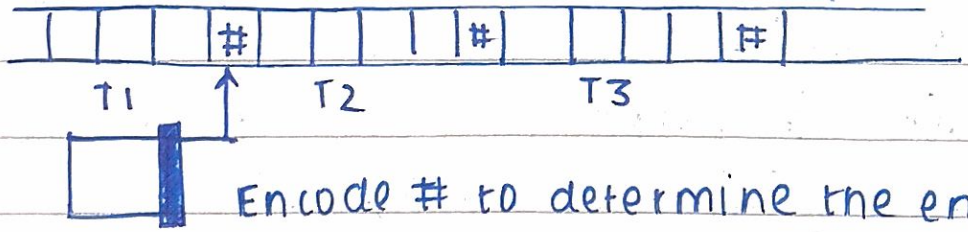
aka Half ∞ tape

3) Multitape TM



$\delta((q_0, q_1, q_K), a) \rightarrow (q_{j_0}, q_{j_1}, \ldots q_{j_K}), (a_{j_0}, a_{j_1}, \ldots a_{j_K}), (L, R, L)$

concatenate the contents of the tapes from multitape TM



Encode # to determine the end of a tape.

take the finite block contents of every tappe & then concatenate. We know the length of blocks (k, l, m) hence we can access it.

Reads the whole concatenated tape & the comes back and writes on it.

to get more space - copy contents present elsewhere to get more space

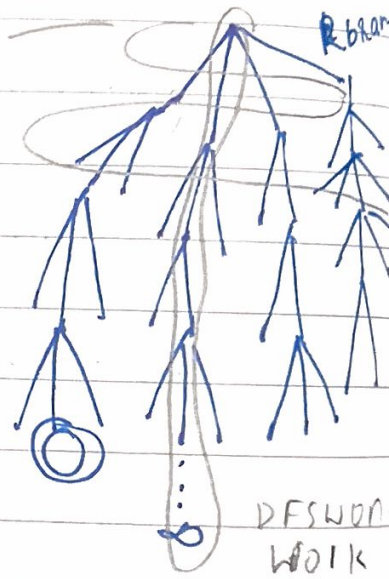
May be more efficient but not more powerful

4) Non-determinism

where the tape read is
What are the contents at the location
what is the state of the tape

check if a branch is accepted/rejected. If accepted the stop. Otherwise check whether other branches are accepted/not.
Depth first search
But what if a path is ∞?

BFS



solution: Level order traversal
└ Requires queue for memory

If a string is rejected - all paths are finite.
If a string is accepted → nothing can be said about finiteness of a path

NDTM takes O(n) time
how much time does a DTM take?
upper bound $O(k^n)$

$O(n^c)$ $P \simeq NP$ $O(n^{ck})$

$P \neq NP \Rightarrow O(n) \neq O(c^n)$

verification $\overset{\text{equivalent to}}{\Longleftrightarrow}$ non deterministic polynomial algorithm

algorithm $\Longleftrightarrow$ Non deterministic TM

Algorithms $\Longleftrightarrow$ TM

CANTOR'S DIAGONALIZATION

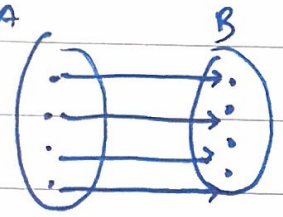
If 2 sets are finite A, & B - comparison is finite

If both sets are ∞ - comparison is ∞

countable : A set is countable if it is finite or it has the same size as the set of integers

One to one (Injective)
Onto (surjective i.e Range = codomain) } Bijective

If 2 sets exist and the function is bijective then $|A| = |B|$ - same cardinality



$f(A) = B$

Uncountable : ∞ not-countable sets

$Z = \{0, 1, 2, \ldots\}$

$E = \{0, 2, 4 \ldots\}$

$E \subseteq Z$ but $|E| = |Z| = \infty$

For any $x \in Z$ $f(x) = 2x = E$

$Z \rightarrow E$

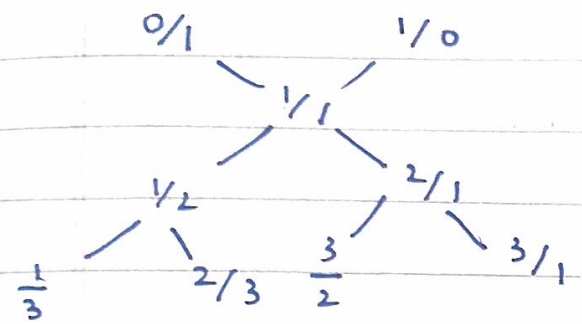
For any $y \in E$ $f(y) = y/2 \in Z$

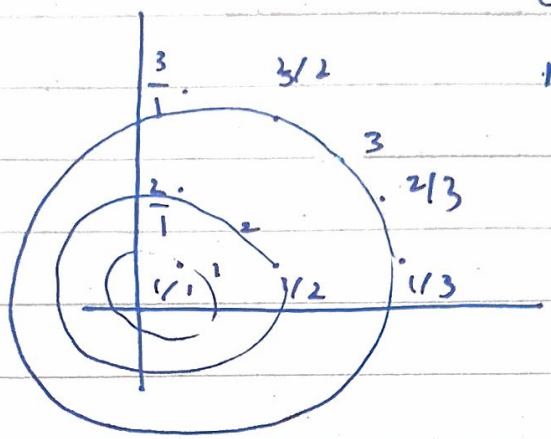
Bijective

LEMMA:

Prove that the set of rational numbers is countable

Stern Brocot tree - levelorder traversal will give mapping from set of integers to set of rationals.





(BFS) not (DFS) ↓ ∞ path

∴ f(n) gives the n^{th} rational number

∴ Bijective function — countable ∵ $|\mathbb{Z}| = |\mathbb{R}| = \infty$

LEMMA: The set of all real nos. is uncountable

PROOF: (By method of contradiction)

Assume that the set of all real nos. is countable

$\mathbb{R} \Leftrightarrow \mathbb{Z}$

$1 \longrightarrow 2.\textcircled{3}456\cdots$

$2 \longrightarrow 3.4\textcircled{8}7893\cdots$

$3 \longrightarrow 0.12\textcircled{3}456$

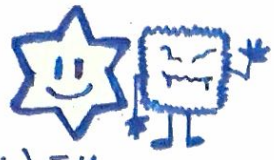
similarly every real no. has some integer index

$f(x) =$ some integer

diff. from 8

$X = 0.\underset{\text{diff from 3}}{5}\,6\,\underset{\text{diff from 3}}{7}\cdots$

The n^{th} digit is different from the n^{th} digit of the n^{th} real



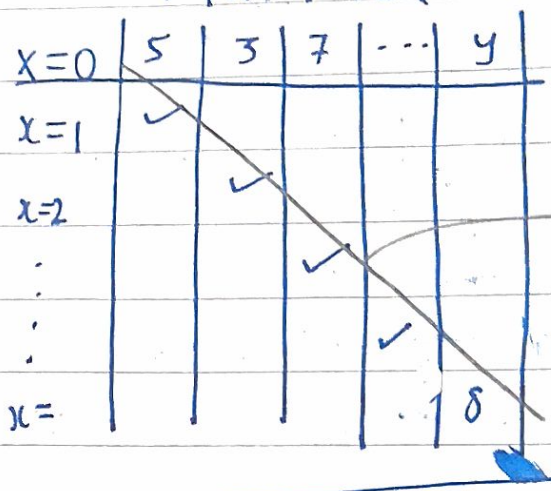
$F(x) = y$

y^{th} digit is different from the y^{th} digit of the y^{th} real. This is not possible. (Paradox)

so such a function cannot be constructed.

$\therefore$ set of reals is not bijective to set of integers

$\therefore |R| \neq |Z|$. R is uncountable.



~~DECIDERS~~ RECURSIVELANGS

DECIDERS : TM that halts and either accepts / rejects

RECOGNISERS ⇒ Acceptors ⇔ R.E.L

TM accepts & halts but can run infinitely if $X \notin L$

└ recursively enumerable languages

NOT EVEN REGOCNIZABLE : run into ∞ loop

THEOREM : ∃ languages which are not even recognizable

PROOF : TM ⇔ Binary string

set of binary strings = Σ^* where $\Sigma = \{0, 1\}$

$\Sigma^* = \{\varepsilon, 0, 1, 00, 01, 10, 11 \ldots\}$

1 2 3 4 5 6 7

mapped to Z

countable

$\{i_1, i_2, i_3 \cdots \} \Rightarrow |\Sigma^*| = |\mathbb{Z}|$

$M = \{$set of all TMs$\}$

$P = \{$set of all languages containing ∞ binary strings$\}$

$x = \underbrace{0\,1\,0 \cdots 0}_{2^k - 1 \text{ to } 2^{k+1} - 1}$ (k)

LEMMA Let A be an infinite countable set. Then the power set $2^{|A|}$ is not countable

PROOF $S \in 2^{|A|}$ $S = (1, 0, 0, 1 \cdots$ (if it has an element; "if it doesn't have an element")

⇓ defining each set in the powerset

characteristic vector

A =	e_1	e_2	e_3	e_4	· · · · · · · ·
set 1	**0**	1	0	1	· · · ·
set 2	1	**1**	0	0	· · ·
set 3	0	0	**1**	0	· · · · · ·
set 4	1	1	1	**0**	· · · · ·

Construct a new set, with the complement of the diagonal elements

New set P = 1 0 0 1 · · · · (differs from n^{th} set in n^{th} position)

↓ Differs from set 1 in 1st position

$\therefore P \in 2^{|A|}$ Bhavishya

$\Rightarrow$ The new set P does not belong to the table.
But P belongs to the table. ~~so table~~ as the nth set.
But then the nth position of P does not differ from the nth set which is P itself.
This is a contradiction.

$\therefore$ Assumption that $2^{|A|}$ is countable is false

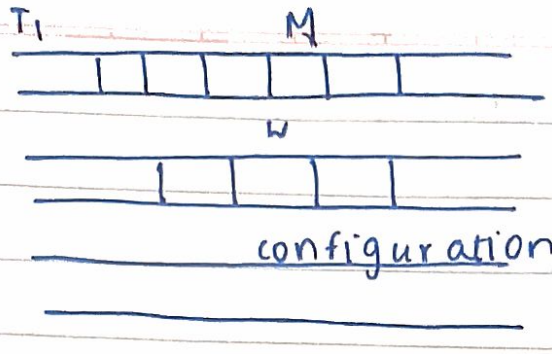
Replacing A with Σ^* — set of all strings
We can show that the set of all languages $= 2^{|\Sigma^*|}$ is uncountable

set of TMs is a countable ∞ value.
$\therefore$ Every lang $\not\Leftrightarrow$ set of TM
(Every lang: larger; set of TM: smaller)
$\Rightarrow \exists$ some languages with no TMs

TM is a sequence/string encoding an algorithm

UNIVERSAL TURING MACHINES - equivalent to a general computer
$U = (M, w)$ U = Processor
M = any algorithm
w = sequence / string (i/p)

Antara @



HALTING PROBLEM: Given an input does a Turing machine halt or not? (undecidable)

LEMMA If L is Turing recognizable and $\bar{L}$ is Turing recognizable then L is decidable.

PROOF ∃ a TM (M1) which recognizes L & a TM (M2) which recognizes $\bar{L}$

construct a TM that runs M1 and M2 in parallel

$w \in L$: In finite time M1 will halt (∵ it is accepted) so then M2 should halt.

$w \notin L$: In finite time M2 will halt (∵ it is accepted) so then M1 should halt.

$\Rightarrow$ L is decidable

$L_{TM} = \{(M, w) \mid \text{The TM } M \text{ accepts } w\}$

Is L_{TM} recognizable? Yes

ara

$L_{IM} = \{(M,X) \mid M$ is a TM and it accepts $x\}$

L_{IM} is recognizable

L_{IM} is not decidable

Halting problem : Does the TM U for L_{IM} halt ?

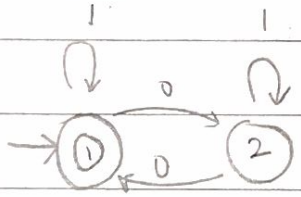
Q can we desing a universal TM which doesn't halt only for not-even recognisable languages ?

NO.

Dhobi washes clothes of all people in town who do not wash their own clothes.

Paradox : Who will wash the dhobi's clothes.

solution : send the Dhobi out of town



Encode the machine

0 for encoding , 1 is a delimiter

2 states — use 2 zeroes

0 goes to state 2

0 goes to state 1

00 | 00 | 0 | 0 | 00 → 1 remains at 2

no. of states | delimiter | 1 remains at state 1

00 | 00 | 0 | 0 | 00 || 0

set of final states

i.e their encoding

L_{FSM} = { All FSMs that do not accept themselves }

Is L_{FSM} regular ?

	0	1	2	. . . . .	ith
0	[1]	0	1	. . . . .	.
1	1	[0]	0	. . . . .	.
2	1	0	[0]	. . . .	
3	0	0	1	[1] . . .	
.					
.					
.					
ith'	0	1	1	0	[?] . .

should be in the set but is not present contradictio

If ? = 0 ∴ It doesn't accept itself.

If ? = 1 ∴ It accepts itself

∟ can't be in the set, but is present in the set

(contradiction)

This paradox can be solved by removing the assumption that L_{FSM} is regular.

∟ ∃ a TM to do this but not a ∧ FSM