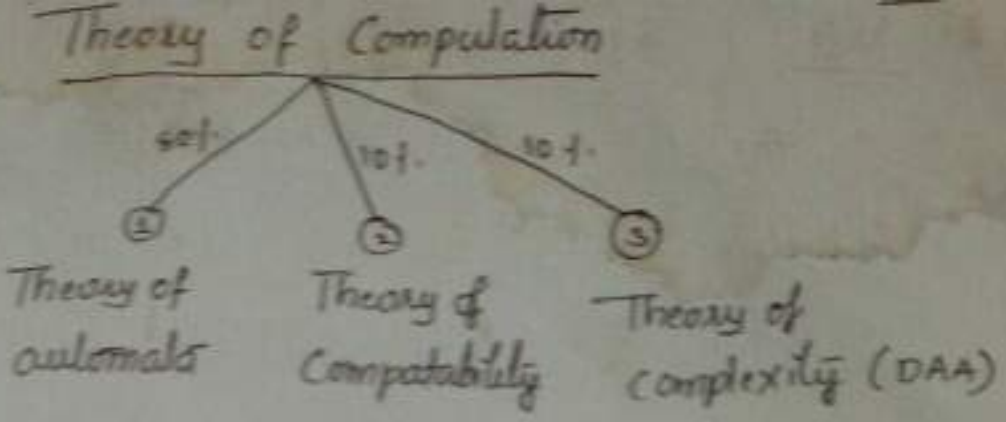
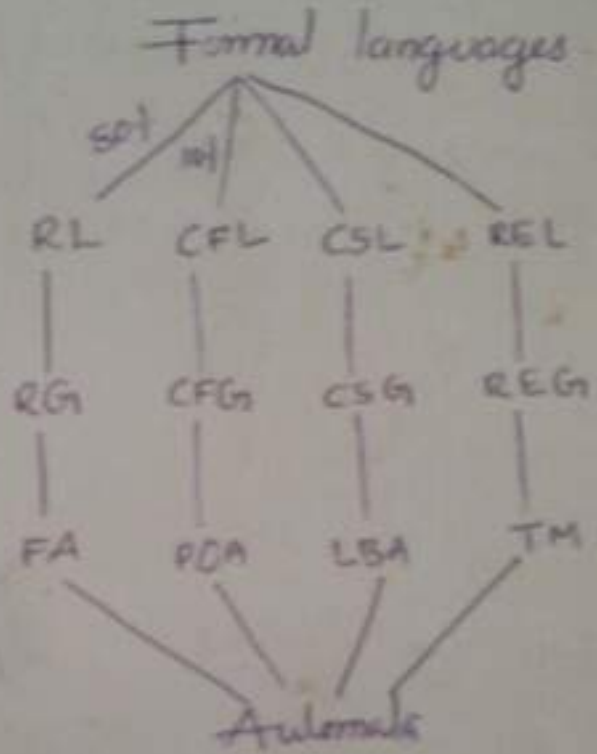




TOC:- the study of mathematical representation of Computation of machine and their computability is TOC

Formal language: The language which has proper alphabet, grammar and a model to recognise a language is called as formal language

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

$L = \{01, 10, 11, 00\}$

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

151

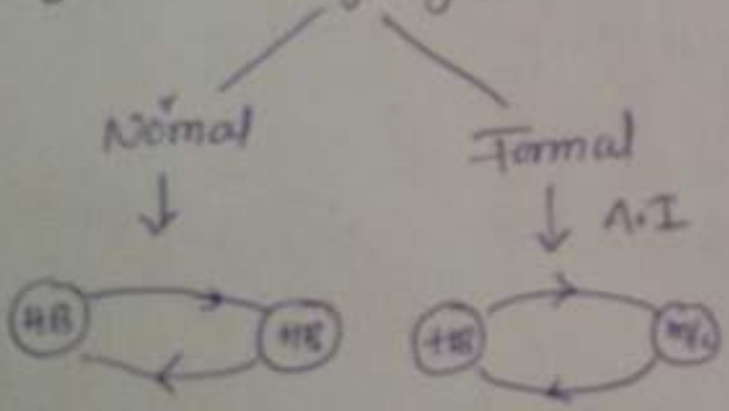
Note:

1. the collection of strings where we can put restrictions and conditions in the formation of a string is called Formal language

2. in the formal language we need to give its importance only for the formation of the string rather than meaning of a string

Ex: OAA, GAA

3. Language



4. Formal languages are more accurate than its natural [i.e free from ambiguity]

Grammar: the collection of rules which are used to generate the strings is called as grammar

Note: Grammar is a generating device

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Grammar Language model.



Expressive power of automata or accepting power of automata or generating power of automata

⇒ No of Languages accepted by automata is called expressive power of automata

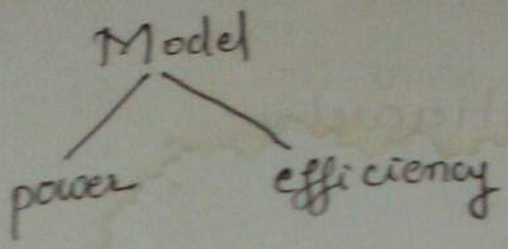
$\epsilon(FA) = 1$ } RL
$\epsilon(PDA) = 2$ } RL, CFL
$\epsilon(LBA) = 3$ } RL, CFL, CSL
$\epsilon(TM) = 4$ } RL, CFL, CSL, REL

$T_3 \leq T_2 \leq T_1 \leq T_0$

Note: what we do with a turing machine something we can do with a computer and vice versa

$\therefore \epsilon(TM) = \epsilon(\text{computer})$

Generated by Smart Document Scanner https://bit.ly/SmartScanner



Note: 1. If m_1 & m_2 be the two models such that $\epsilon(m_1) > \epsilon(m_2)$ then m_1 is more powerfull than m_2

2. If m_1 & m_2 are two models to provide solution for the same problem P and if m_2 takes less time than m_2 then m_1 is more efficient than m_2

Determini

Deterministic & Non-Deterministic

Deterministic	Non-Deterministic
→ what we require & what we don't require also we write	→ what we require only we write
→ most efficient	→ Least efficient
→ Less/equal powerful	→ more powerful
→ Less time & memory	→ more time & memory
→ Implemented in only one way	→ implemented in many ways

Note:

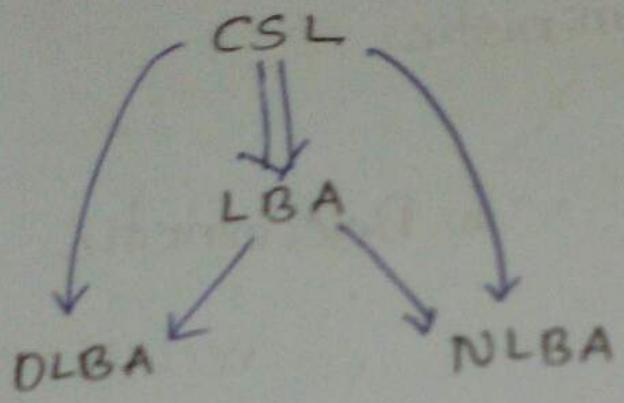
① RL ⇒ FA ⇒ DFA = NFA

- Default DFA
- $L(DFA) = L(NFA)$
- power of DFA = power of NFA
- DFA is more efficient than NFA
- Every DFA is NFA & some NFA can be converted into DFA

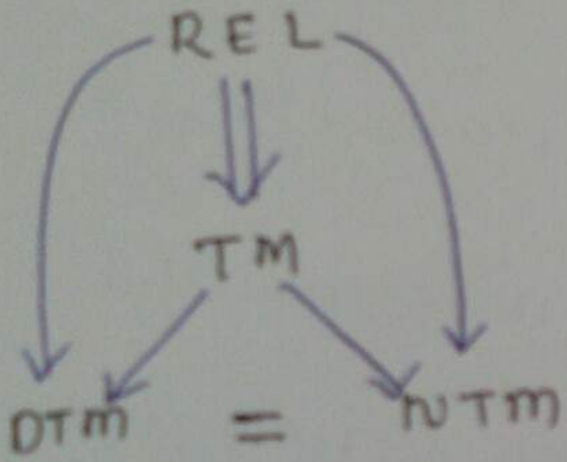
② CFL ⇒ PDA ⇒ DPDA ≠ NPDA

- Default NPDA
- $L(DPDA) \neq L(NPDA)$ / - NPDA is more powerful than DPDA
- DPDA is more efficient than NPDA
- Every DPDA is NPDA but not vice versa

351



— No relationship
— research is going on
— DLBA is more efficient than NLBA



— $\epsilon(DTM) = E(NTM)$
— power of DTM = power of NTM
— DTM is more efficient than NTM
— every DTM is NTM & every NTM can be converted into DTM

352

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Alphabet : set of symbols (finite)

$\Sigma = \{0, 1\}$ or $\Sigma = \{a, b, c, d \cdots z\}$

Sentence | string | word : set of symbols from an alphabet

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

$w = \{0, 1, 01, 1101 \cdots\}$

Length of a string : no of characters in a string

$w = 001$

$|w| = 3$

Null String | Empty String : ϵ, $\wedge$

epsilon, cap

Length of num string $= 0$

$w = \epsilon$ or $w = \wedge$ $\quad |w| = 0$

The empty string is inherently exists beginning and ending of any string

$\epsilon w = w \epsilon = w$

Empty set : $\{\}$

$\emptyset = \{\}$

Generated by Smart Document Scanner https://bit.ly/SmartScanner

empty string / Null string	empty set
ϵ or $\wedge$	ϕ
$\{\epsilon\}$ or $\{\wedge\}$	$\{\}$.
$r \cdot \epsilon = r$	$r \cdot \phi = \phi$
$r + \epsilon = r + \epsilon$	$r + \phi = r$

Prefix: Any no of Leading symbols of a string is called prefix

Eg: Ajay — n length

ϵ
$\wedge$
Aj
Aja
Ajay

} n+1 prefixes

Suffix: Any no of trailing symbols of a string is called Suffix

Ex: Ajay — length n

ϵ
y
ay
jay
Ajay

} n+1 suffixes

USA

Generated by Smart Document Scanner https://bit.ly/SmartScanner

substring: By deleting a prefix (or) suffix (or) both

Types:

1. Trivial (or) Improper ⇒ itself, ϵ
2. Non-Trivial or Proper ⇒ other than itself, ϵ

Eg: Ajay — n

0 – ϵ — ①
1 – A, j, a, y — ④
2 – Aj, ja, ay — ③
3 – Aja, jay. — ②
4 – Ajay — ①

total substrings ↓ ≤ n+2 | length of string

— Total no of substrings ≤ n+1
— Trivial substrings = 2
— Non Trivial = ≤ n+1-2 = ≤ n-1
— non empty = ≤ n
— every prefix & suffix must be a substring converse need not be true
— Trivial substrings are both prefix & suffix

551

Generated by Smart Document Scanner https://bit.ly/SmartScanner

power of Alphabet

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

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

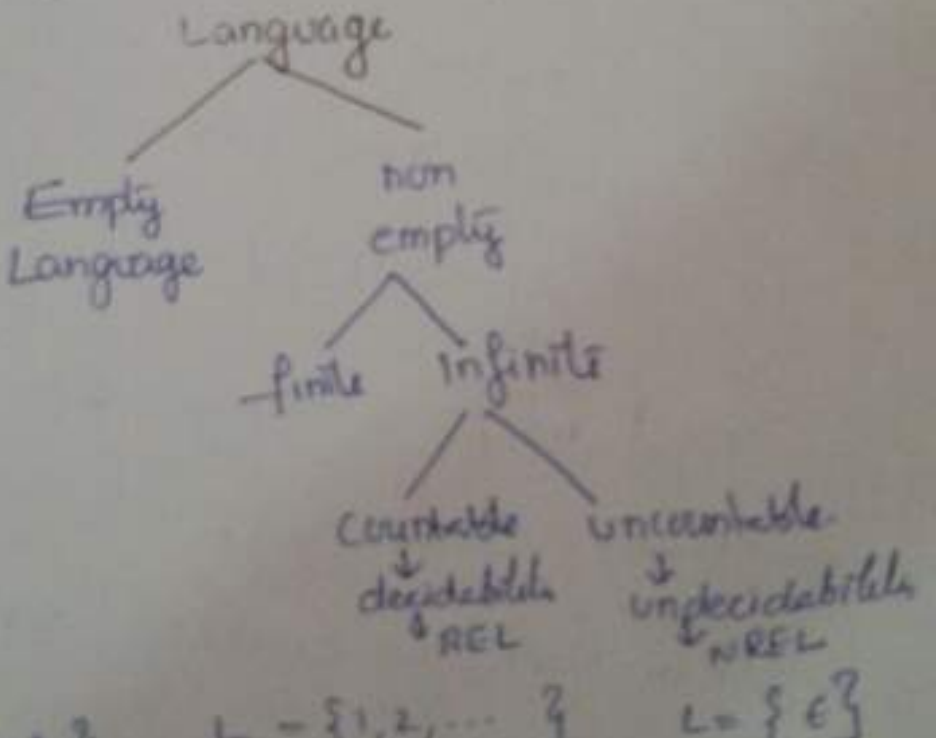
$\Sigma^2 = \{0,1\}\{0,1\} = \{00, 01, 10, 11\}$

$\Sigma^3 = \{0,1\}\{0,1\}\{0,1\} = \{000, 001, 010, 011, 100, 101, 110, 111\}$

$\Sigma^n = \{\text{string}\} \text{ Length } n\}$

$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \ldots$ closure | Kleen | universal

$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \ldots$ positive closure



empty lang: $L = \{\}$

non-empty $L = \{1, 2\}$ Finite $\quad L = \{1, 2, \ldots\}$ infinite $\quad L = \{\epsilon\}$

countable infinite $= \{1, 2, 3 \ldots n \ldots\}$

A set is enfinite, if it can be mapped into 1-1 relationship with N (positive integers) and positive integeres is countable infinite

552

Generated by Smart Document Scanner https://bit.ly/SmartScanner

countable infinite

$L = \{1, 2, 3 \ldots n \ldots\}$

$L = \{2, 4, 6, \ldots 2n \ldots\}$

operations on L

- union
- intersection
- concatenation
- complement $\bar{L} = \Sigma^* - L$
- Inverse $L^- = \{w^R \mid w \in L\}$

Note:- if Σ is any alphabet then Σ^* is called as universal language or universal set

- If L is any language from 'Σ' aphabet then 'L' is subset of Σ^* $\boxed{L \subseteq \Sigma^*}$

- Every Finite Language is accepted by FA
  ie every FL is Regular

- Every subset of countable set is ~~finite~~ countable
- power set of countable is uncountable 'S' is countable

2^S is uncountable

(651)

Generated by Smart Document Scanner https://bit.ly/SmartScanner



Generated by Smart Document Scanner https://bit.ly/SmartScanner

(1) empty set $\emptyset$ is a

(a) Regular but not CFL

(b) CFL " " CSL

(c) RSL " " CFL

(d) REL

(2) that set $L = \{a^n b^n \mid 1 \le n \in \text{google} \}$

a) RL b) CFL but not RL

c) RL but not CFL d) RL but not CSL

(3) Which of the most appropriate language for its finite language L

a) L is regular c) L is CFL

b) L is CSL d) L is REL

(4) Choose the correct statement from following

a) Every RL is finite c) every RL is infinite

b) Every NRL is finite d) Every NRL is infinite

751

Generated by Smart Document Scanner https://bit.ly/SmartScanner

⑤ its empty language ϕ is

a) countable but not RL

b) Regular but not countable

c) both RL & countable

d) none

⑥ $L = \{ B^m G^n \mid m, n \geq 1 \}$

a) RL but not countable b) Regular & countable

c) countable RL d) none

⑦ $L = \{ B^m G^n \mid m = n \}$

a) countable but not RL b) RL but not countable

c) Regular & countable d) none

752

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Finite state machine:

Regular language (Finite numeric program)

FA = RE = RG

The language which is accepted by FA or generated by Regular Expression or generated by Regular grammar is called as Regular language

Ex $L = \{B^n \mid n \geq 1\}$

$= \{B, BB, BBB \ldots\}$



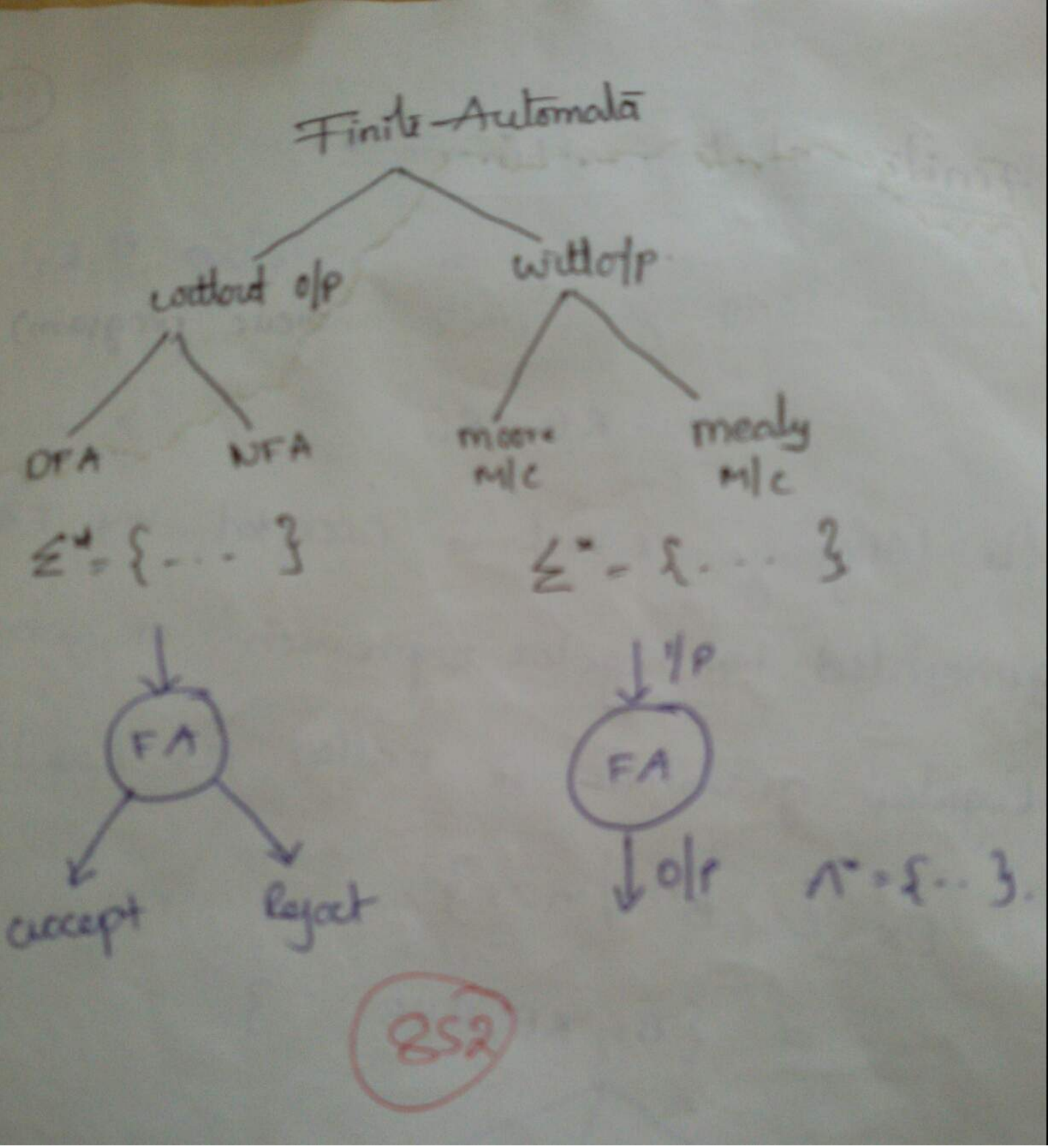
$r = B^+$ $S \to BS \mid B$.

- The language 'L' which is not accepted by FA is called Non Regular language

- Every FL is either RL or NRL

851

Generated by Smart Document Scanner https://bit.ly/SmartScanner



Generated by Smart Document Scanner https://bit.ly/SmartScanner

Deterministic Finite Automata

The mathematical representation of RL is called FA

(or)

The Mathematical system that process the string is called as FA

or

FA is 5 tuple $M = (Q, \Sigma, \delta, q_0, F)$

where Q = set of all states

Σ = input alphabet

$\delta = Q \times \Sigma \rightarrow Q$ is a transition function

q_0 = initial state

F = set of all final states

Note:

- In general the FA is DFA
- FA can be constructed with or without final state
- The no. of final states in FA can be 0 or 1 or more
- DFA is a complete system which responds for both valid and invalid i/p
- the system is complete iff a transition function is defined for each and every i/p symbol at each and every state.

951

Generated by Smart Document Scanner https://bit.ly/SmartScanner

- Total no of transitions in DFA = $|\Sigma| \times Q$
- in DFA every string has only 1 transition path

Non Deterministic Finite Automata (NFA)

- FA which has 0 or 1 or more transitions for any i/p symbol from any state is called as NFA

(or)

NFA is 5 tuple $M = (Q, \Sigma, \delta, q_0, F)$

where

Q = set of states

Σ = input alphabet

$\delta = Q \times \Sigma \to 2^Q$ is a transition

q_0 - initial state

F - set of final states

→ NFA is Rough idea to represent a RL

→ No concept of Dead states & complement NFA

- in NFA a string can have multiple transition path

- DFA — complete system (valid i/p & invalid i/p)
  NFA — incomplete (invalid i/p)

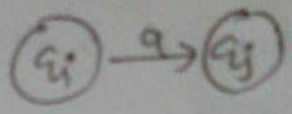
Generated by Smart Document Scanner https://bit.ly/SmartScanner

Instantaneous Description (ID)

- ID describes the moments of FA
- The momements of FA depends on 2 entities
  - current state
  - current i/p symbol.

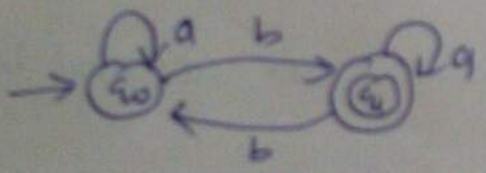
$\delta(q_i, a) = q_j$

(current state, i/p symbol)



Representation of FA

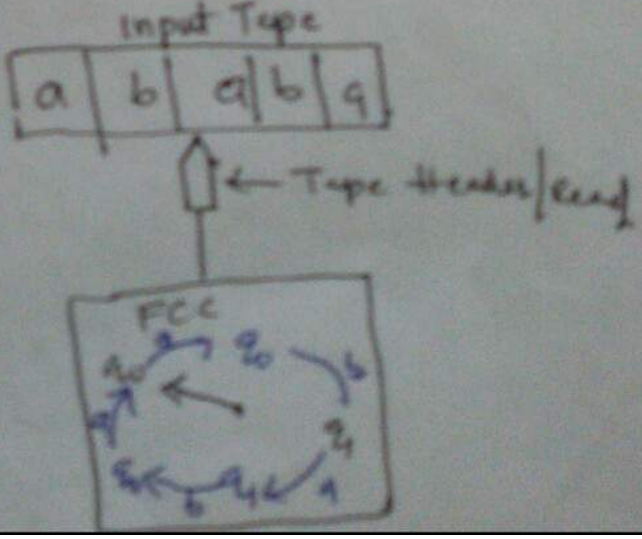
- Transition Diagram
- Transition table



δ	a	b
→ q_0	q_0	q_1
(q_1)	q_1	q_0

Block diagram of FA.

- it consists of 3 components



1. input Tape
2. Tape Header
3. Finite control unit

1051

Generated by Smart Document Scanner https://bit.ly/SmartScanner

→ in FA a symbol can be processed but this symbol cannot be stored

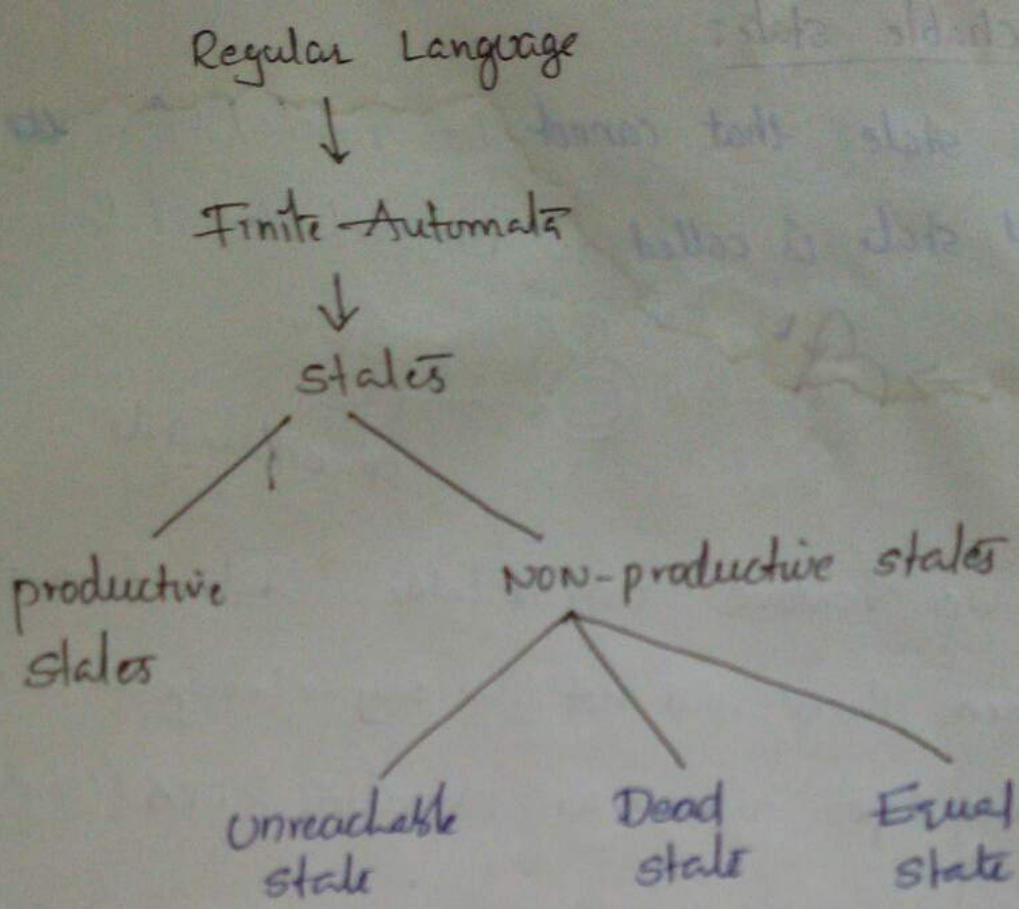
— FA can remember only the current passing symbol

Acceptance of FA

— Let X is any string from the alphabet Σ. corresponding to x if there exist a transition path which starts at initial state and ends in any one of the final state then the string accepted by FA

— the set of all strings which are accepted by FA is called as Language of FA

1052



Productive state:

The state which is involved in the process & valid i/p string is called as productive state

Non Productive state:

The state which is not involved in the accep process & valid i/p string is called as non productive state

[or]

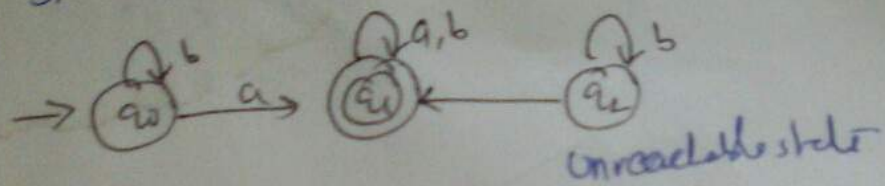
The state which presence or absence will not effect the language & FA is known as Non-productive state.

(11S1)

Generated by Smart Document Scanner https://bit.ly/SmartScanner

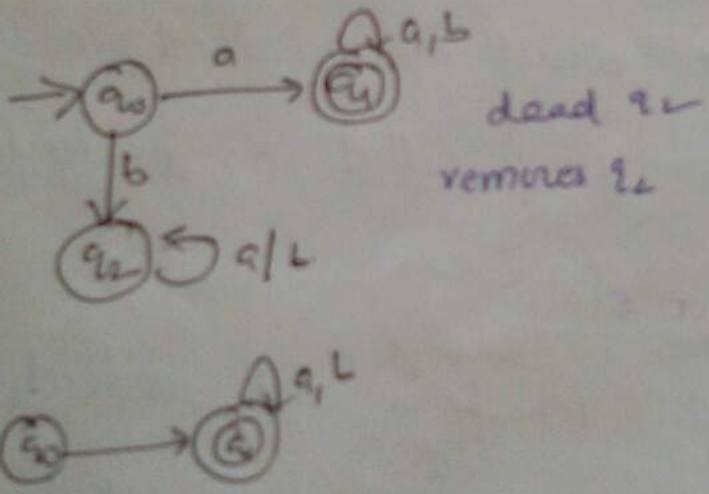
Unreachable state:

The state that cannot be reached from its initial state is called as unreachable state



— If we remove unreachable state from DFA then there will not be any change in its language as well as structure of DFA (completeness)

Dead state: A state from where you cannot come back is called as dead state



→ If we remove dead state from DFA there will be no change in the language but there will be change in structure of system

i.e System is NFA which is incomplete

→ every DFA must contain atmost one DFA

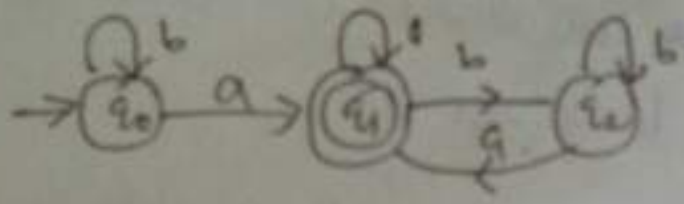
Generated by Smart Document Scanner https://bit.ly/SmartScanner

equal state:

Two states p, q, are said to be equal

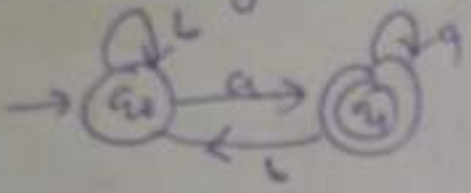
if $\delta(p,x) = \delta(q,x) \quad \forall x \in \Sigma^*$

$\Sigma^* = \{\epsilon, a, b, aa, ab, bb \ldots\}$



$q_0 = q_2$

After removing equal state



→ If we remove equal state from DFA there wouldn't be any change in language as well as structure of DFA



MFA
(minimal FA)

→ The FA free from US & ES is called MFA

Generated by Smart Document Scanner https://bit.ly/SmartScanner

- The DFA which free from Final state accept empty language $\emptyset$

- The where all the final states are unreachable accept empty language $\emptyset$

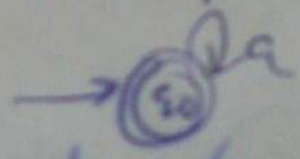
The DFA where all states are final accept universal language

the DFA where all non-final states are unreachable accepts universal language

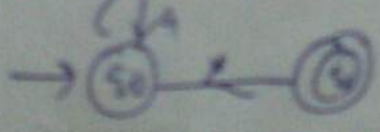
every DFA accept only one RL but a RL can be accepted by more than one DFA

- every RL is accepted by only one MFA i.e FA is not unique but MFA is unique

- No of states in the MFA to accept universal language Σ^* is 1



- No of states in MFA to accept empty language $\emptyset$ is 2



Generated by Smart Document Scanner https://bit.ly/SmartScanner

process of construction of FA

- start construction of system to process valid i/p and then combat the system to process invalid invalid i/p
- simplify the processing system by removing unreachable and equal state

for other possibilities

options

- self loop
- loop (forward / backward)
- Dead / Rejection (worst case)

1351

Generated by Smart Document Scanner https://bit.ly/SmartScanner

→ construct MFA that accept all its strings of 0's & 1's

→ where every string ends with ① 0 ② 01 ③ 101

→ where every string contains its substring (i) 0 (ii) 10 (iii) 010

→ starts & ends with '0'

→ same symbol

→ different symbols

→ start with 11 or 00

→ ends with 11 or 00

→ ends with 10 or 00

→ 2nd symbol from left end is 0

→ 3rd 2

→ 2nd right 1

0

→ 3rd. right

→ at most two 0's

→ at least two 0's

→ no of 0's in string even.

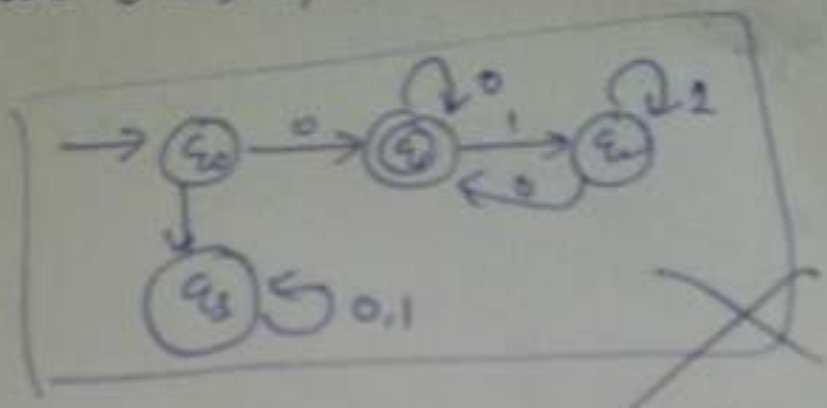
→ odd.

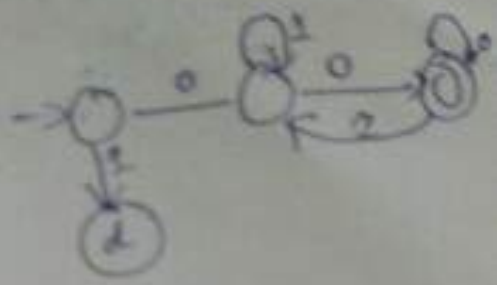
1352

Generated by Smart Document Scanner https://bit.ly/SmartScanner

→ starts & ends with '0'

$\omega = 0 \xrightarrow{X} 0, 0$



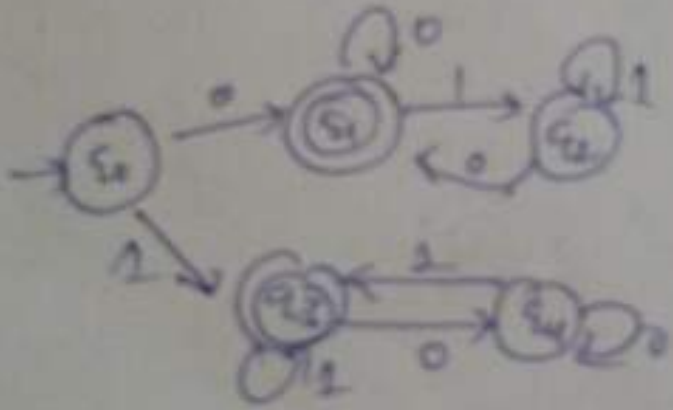


same symbol ⇒ invalid I/P

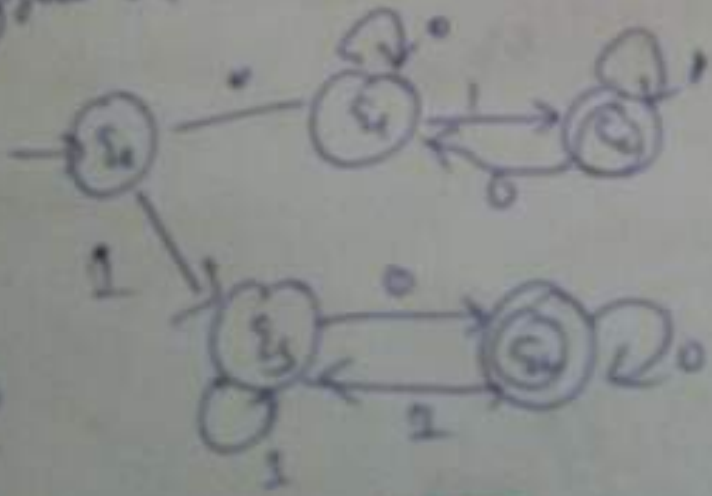
→ dead state

→

$\omega = 0X0, 0$
$1X1, 1$



→ different symbol

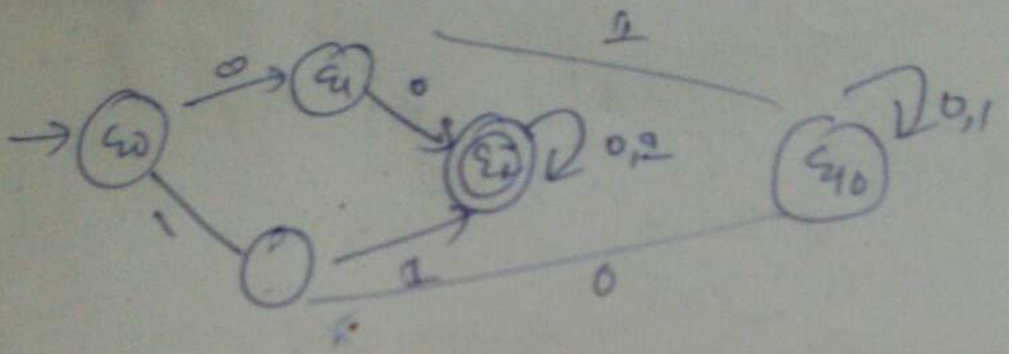


0X1
1X0

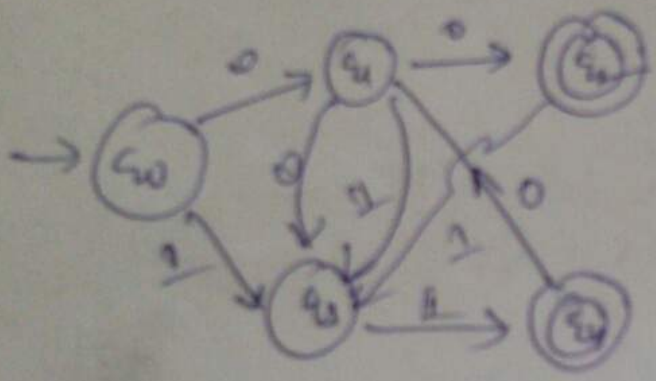
14.51

Generated by Smart Document Scanner https://bit.ly/SmartScanner

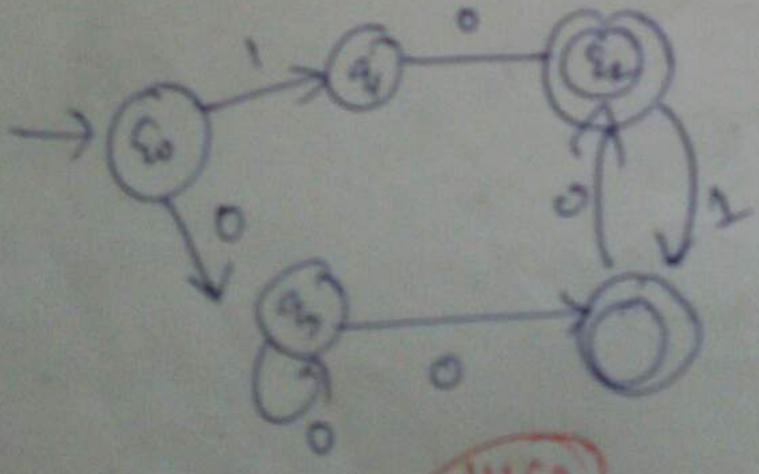
→ starts with 11 or 00



→ end with 11 (or) 00



→ end with 10 or 00

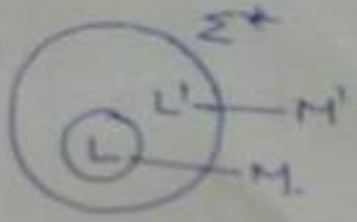


1452

Generated by Smart Document Scanner https://bit.ly/SmartScanner

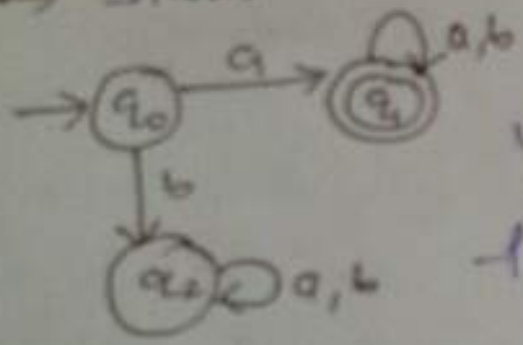
Complimented FA

1. if L is any RL then the compliment of L i.e $L' = \Sigma^* - L$ is also RL

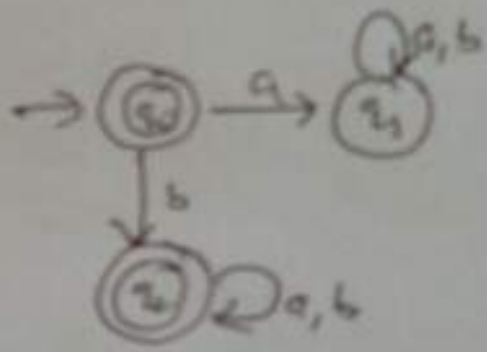


→ if M is the FA to accept the language L then FA for L' can be obtained by interchanging final & Non-final states

→ starts with 'a'

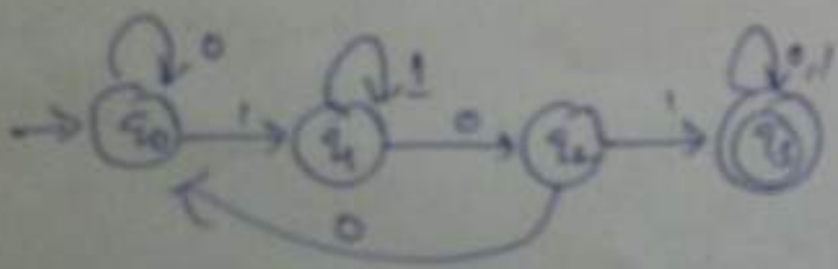


interchanging ⇒ final & non-final

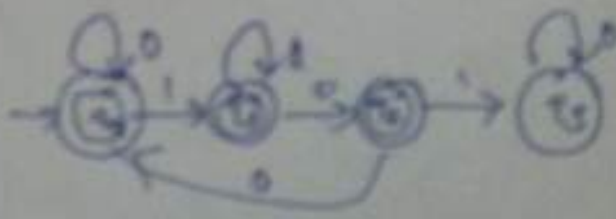


→ not start with a.

→ substring 101



⇒



→ no substring 101

→ ends with 000

→ not ends with 000

1551

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Conversion of NFA to DFA

The process of conversion of NFA into DFA is called as subset construction

NFA $\Longrightarrow$ DFA

n states $\downarrow$ 2^n states

$1 \leq m \leq 2^n$

$m = 1 \rightarrow$ best case

$1 < m < 2^n \rightarrow$ avg "

$m = 2^n \rightarrow$ worst case

Algorithm

Let $M = (Q, \Sigma, \delta, q_0, F)$ be the given NFA

Let $M' = (Q', \Sigma, \delta', q_0, F')$ be the equivalent DFA

Construction of δ':

Start the construction of δ' with initial state and continue for every new state and stop the construction when ever no new state occurs.

Final state: every subset which contains the final state of NFA is the final state in DFA

1651

Generated by Smart Document Scanner https://bit.ly/SmartScanner

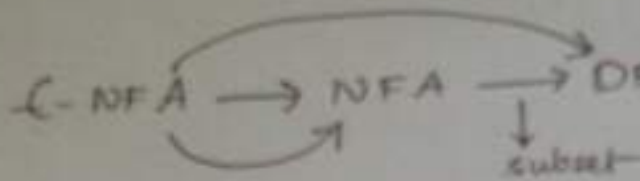
ε-NFA or NFA with ε-moves

The NFA which has a transition even for empty string is called as ε-NFA

(or)

ε-NFA is a 5 tuple $M = (Q, \Sigma, \delta, q_0, F)$

$$\delta = Q \times \Sigma \cup \{\epsilon\} \longrightarrow 2^Q$$

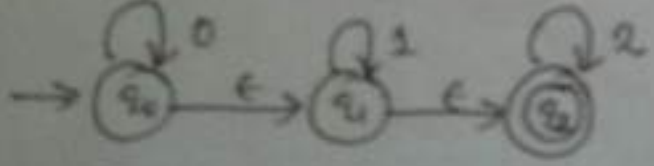


ε-closure (q):

Let 'q' is any state in ε-NFA then the set of all the states which are at '0' distance from state 'q' is called as ε-closure of 'q'

(or)

The set of all the states that can be reached from the state 'q' on the input symbol 'ε' is called as ε-closure of 'q'



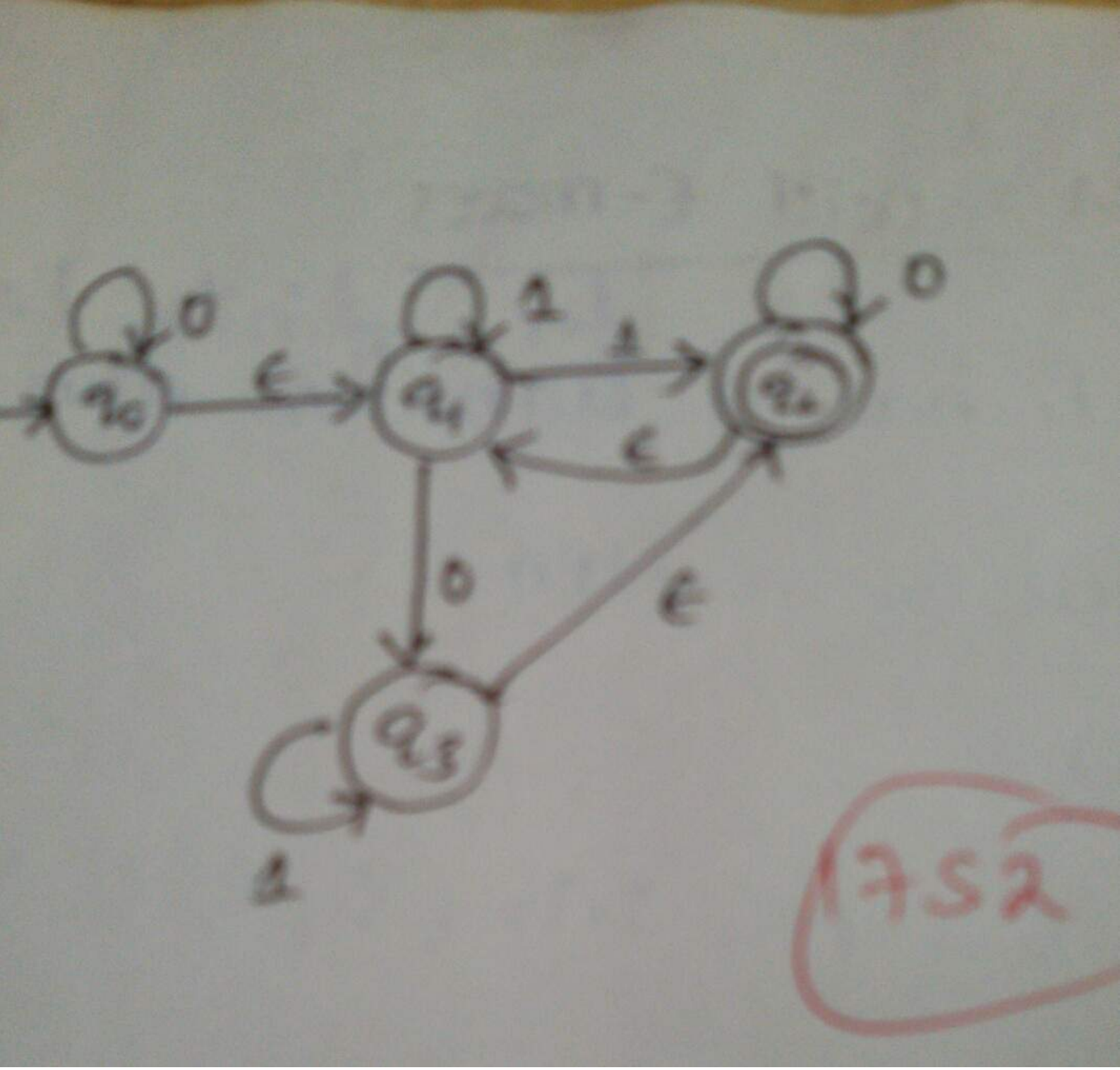
- ε-closure (∅) = ∅

ε-closure (q_0) = $\{q_0, q_1, q_2\}$

ε-closure (q_1) = $\{q_1, q_2\}$

ε-closure (q_2) = $\{q_2\}$

Generated by Smart Document Scanner https://bit.ly/SmartScanner

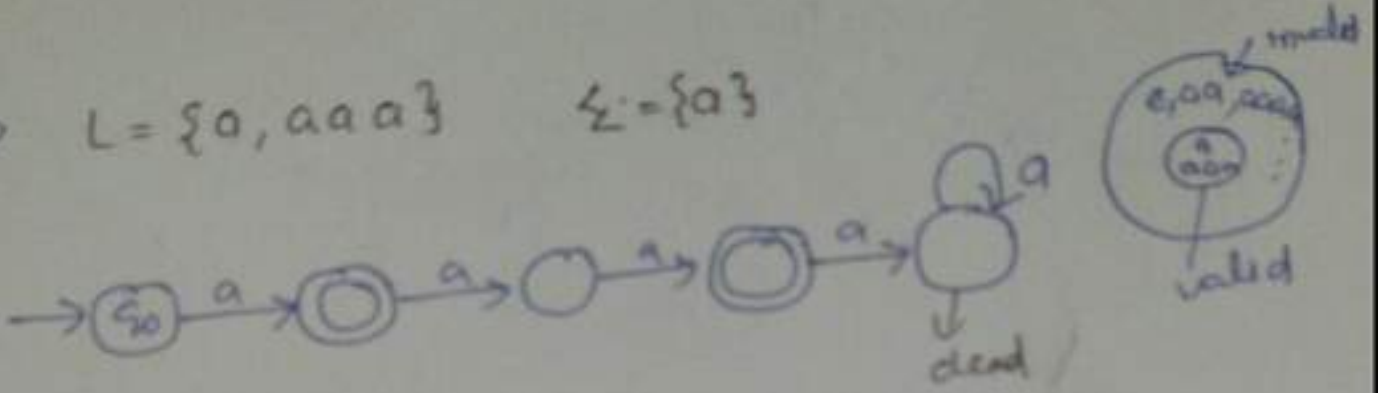


Generated by Smart Document Scanner https://bit.ly/SmartScanner

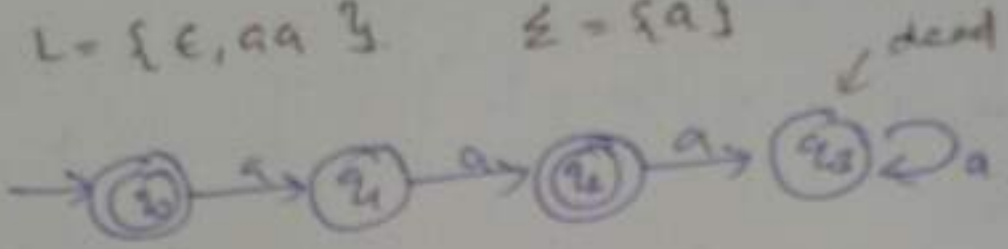
Construction Minimal FA for finite language

MFA

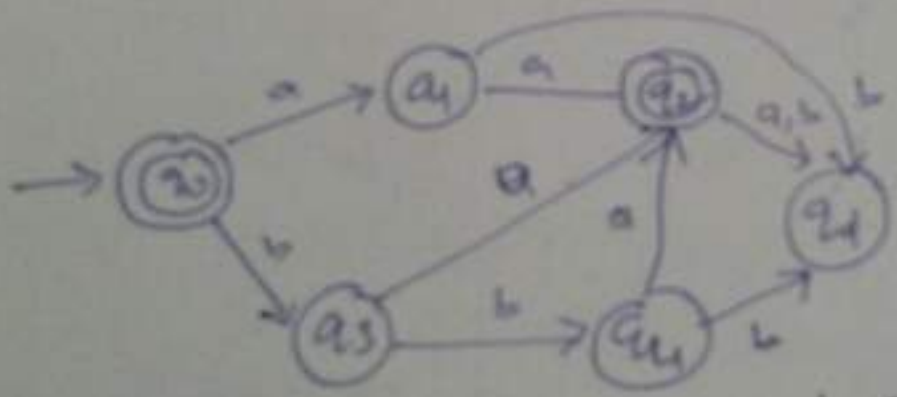
⊕ $L = \{a, aaa\}$ $\Sigma = \{a\}$



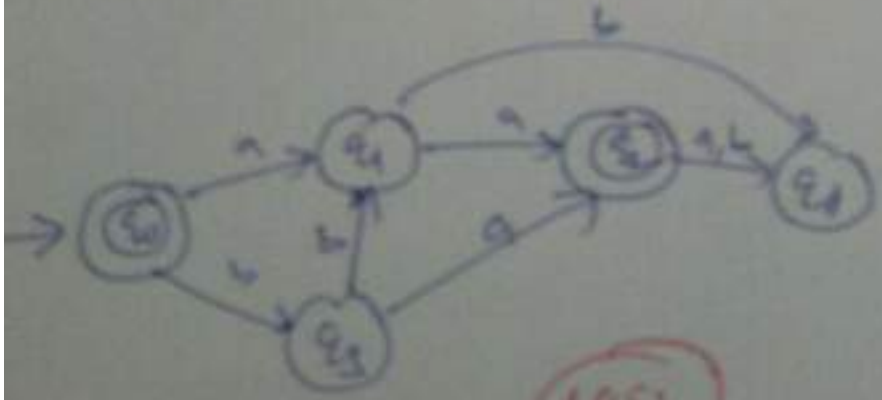
— $L = \{\epsilon, aa\}$ $\Sigma = \{a\}$



— $L = \{\epsilon, aa, ba, bba\}$ $\Sigma = \{a, b\}$



$q_1 = q_4$ avoid equal states to get MFA

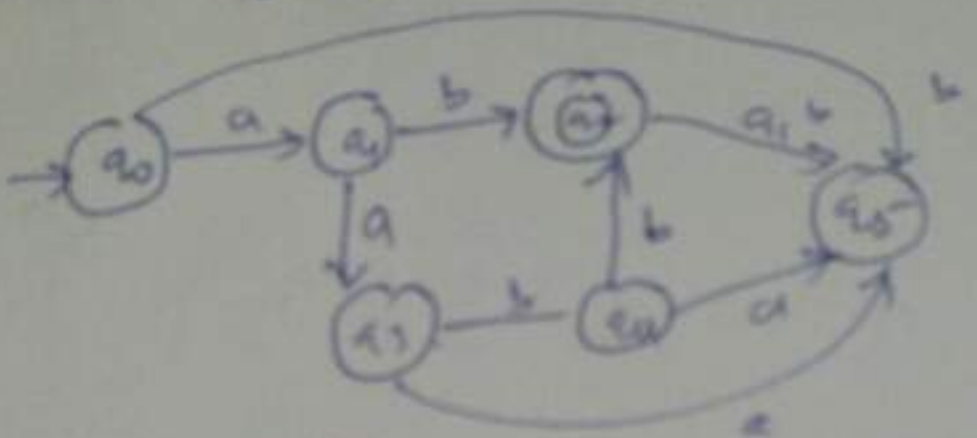


1851

Generated by Smart Document Scanner https://bit.ly/SmartScanner

→ $L = \{a^n b^n \mid 1 \le n \le 2\}$ $\Sigma = \{a, b\}$.

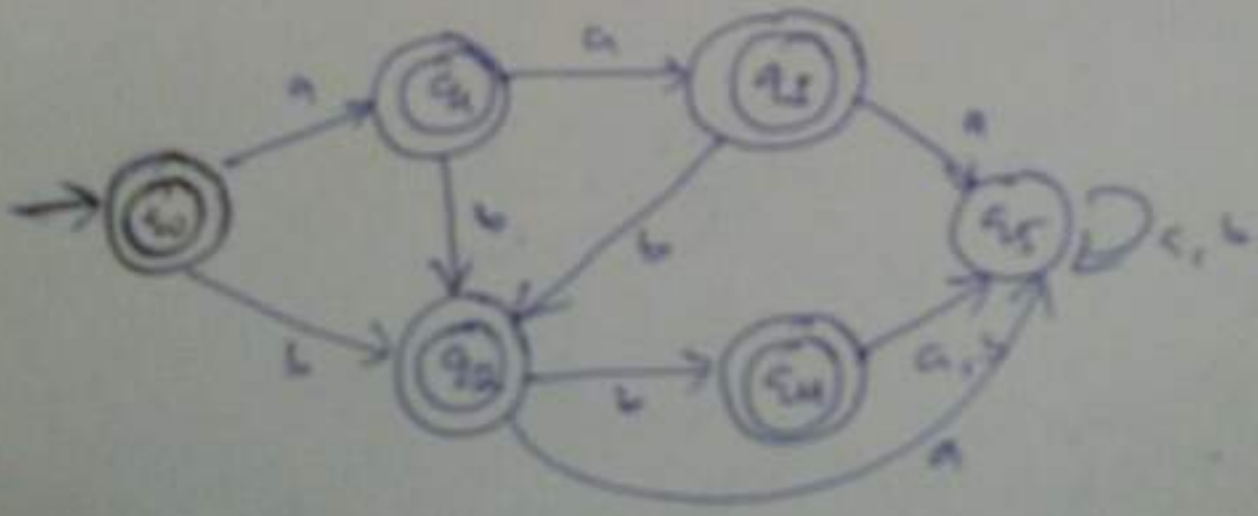
$\{ab, aabb\}$



NFA = 6

$L = \{a^m b^n \mid 0 \le m \le 2 \;\; 0 \le n \le 2\}$ $m = 0, 1, 2$ $n = 0, 1, 2$

$L = \{\epsilon, a, b, aa, bb, ab, abb, aab, aabb\}$



NFA = 6

Generated by Smart Document Scanner https://bit.ly/SmartScanner

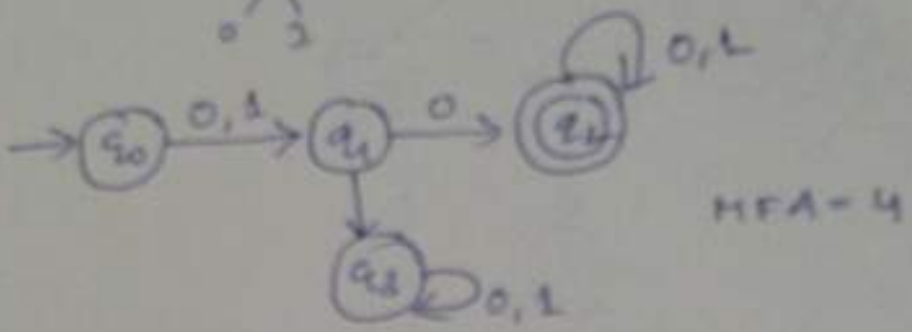
→ construct MFA with 0's & 1's

(i) the second symbol from left end is '0'

(ii) the third symbol from left end is '1'

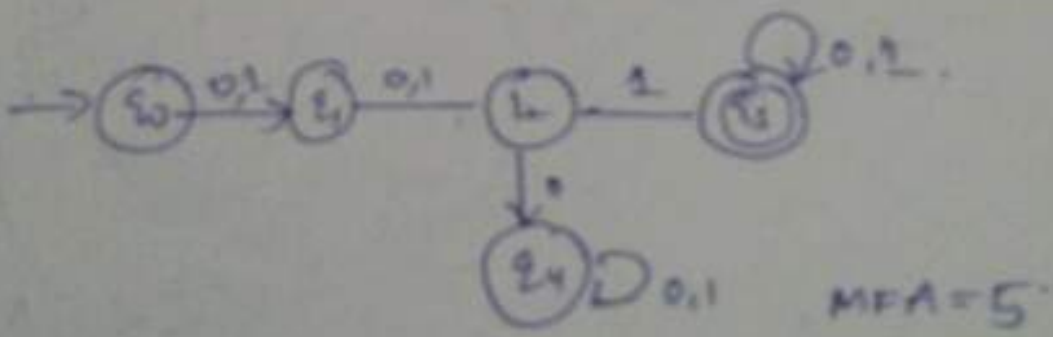
(i) $\Sigma = \{0, 1\}$

$w = x\ 0\ x\ x\ x - - -$



(ii) $\Sigma = \{0, 1\}$

$w = x\ x\ 1\ x\ x$ ——



⇒ the MFA that accept all strings of 0's and 1's where the "nth" symbol from left end is placed. contains exactly "n+2" states

1951

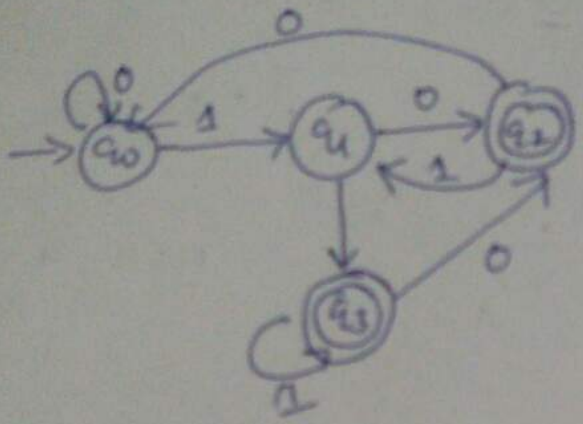
Generated by Smart Document Scanner https://bit.ly/SmartScanner

① the 2nd symbol from right end is 1

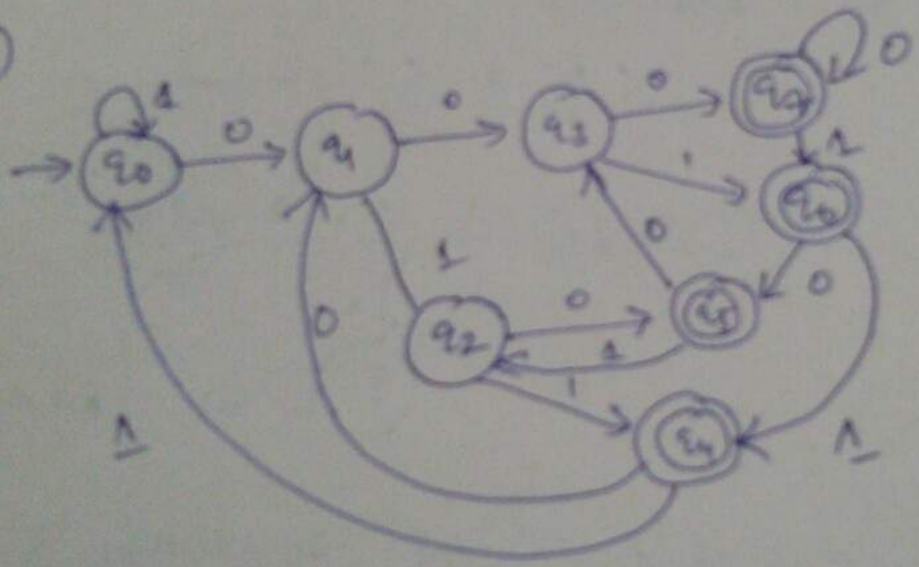
② the 3rd " " " " 0

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

① $w = _____ 1\ X$ (X: 0 or 1)



②



⇒ the MFA that accept all the strings of 0's and 1's where "nth" symbol from right end is placed. contains exactly "2^n" states. 2^{n-1} final states (set of final states)

1952

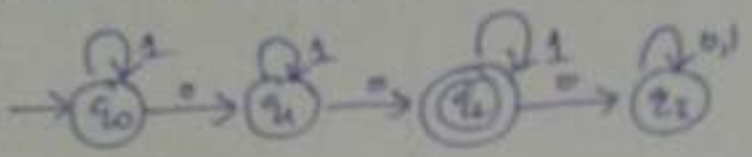
Generated by Smart Document Scanner https://bit.ly/SmartScanner

→ Construct the MFA that accept all strings of 0's and 1's where the no of 0's in a string is

(i) exactly 2 (ii) atmost 2 (iii) atleast 2 &

(iv) even (v) odd (vi) $= 2 \pmod 3$

(i) $w = X0X0X$



$|w|_0 = 2 \rightarrow 2+2$

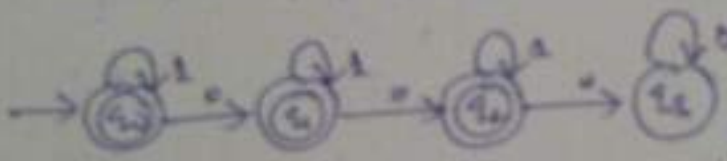
$|w|_0 = 3 \rightarrow 3+2$

$|w|_0 = n \rightarrow n+2$

(ii) atmost 2 $|w|_0 = 0, 1, 2$

0 00

$w = \epsilon, X0X, X0X0X$



$|w| \leq 2 \rightarrow 2+2$

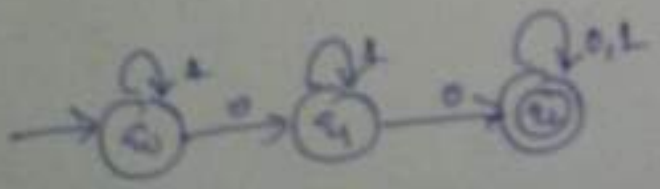
$|w| \leq 3 \rightarrow 3+2$

$|w| \leq n \rightarrow n+2$

non final states

(iii) atleast 2 $|w|_0 \geq 2$

$w = \{X00X, X0X0X\}$



$|w|_0 \geq 2 \rightarrow 2+1$

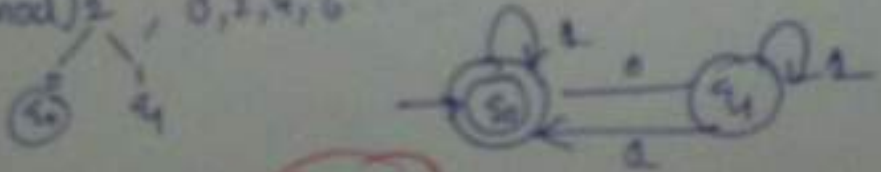
$|w|_0 \geq 3 \rightarrow 3+1$

$|w|_0 \geq n \rightarrow n+1$

one final state

(iv) no of 0's = even

$\equiv 0 \pmod 2$, 0, 2, 4, 6

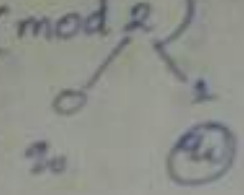


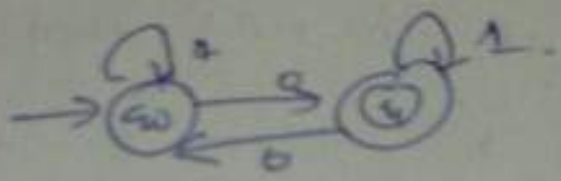
809

Generated by Smart Document Scanner https://bit.ly/SmartScanner

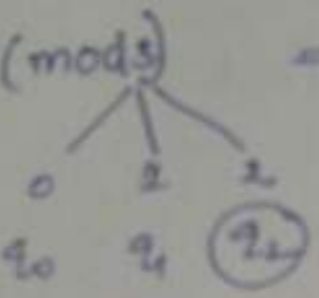
(v) No of 0's is odd

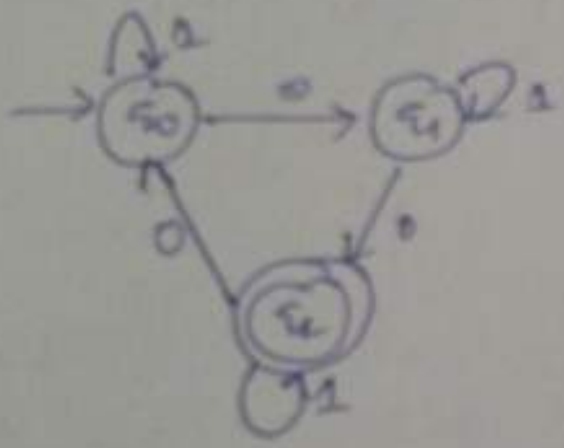
$\equiv 1 \pmod 2$





(vi) $|w|_0 \equiv 2 \pmod 3$ = 2, 5, 8 ....





$|w_0| \equiv r \pmod n$

↓

Finding

S no	language	no of states	no of Final states
1	$\|w\|_0 = n$	$n+2$	1
2	$\|w\|_0 \leq n$	$n+2$	$n+1$
3	$\|w\|_0 \geq n$	$n+1$	one
4	$\|w\|_0 \equiv r \pmod n$	n	one

↓

no of states Length also

or

2052

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Construct the MFA that accepts all the strings of 0's and 1's where every string

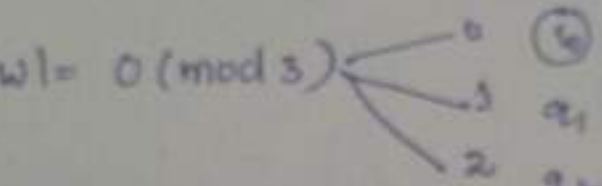
(i) starts with '0' and length of the string is divisible by 3 i.e $|w| = 0 \pmod 3$

(ii) starts with 10 and length of the string is $|w| \equiv 1 \pmod 4$

(iii) . . 00 . . . . . .

$|w| = 2 \pmod 5$

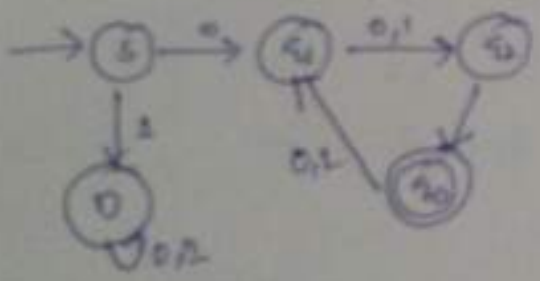
(i) $|w| = 0 \pmod 3$



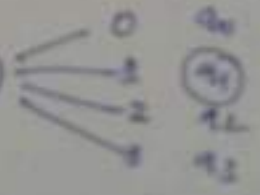
No. of states in MFA

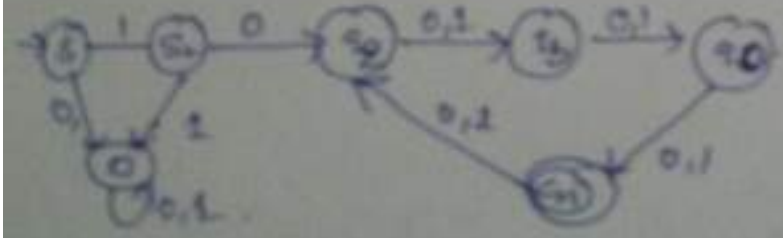
1 + 1 + 3 = 5

↙ Length of string checked ↓ divisible by 3



(ii) $w = 10x$ $|w| = 1 \pmod 4$





MFA = 2 + 1 + 4 = 7

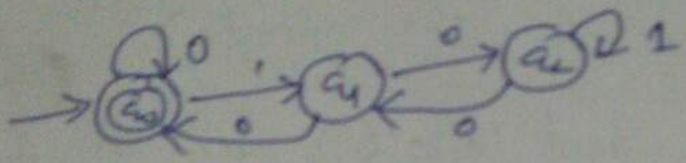
⇒ No. of states in MFA where string starting with 'm' digits and length of string $|w| = r \pmod n$ contains $m + 1 + n$ states (RUSI)

Generated by Smart Document Scanner https://bit.ly/SmartScanner

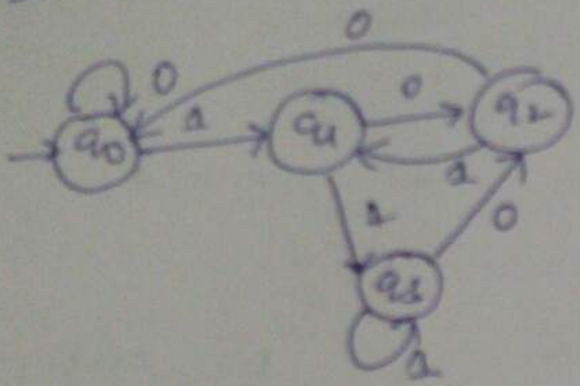
→ Construct the MFA that accepts all binary strings whoose integer equivalent is

(i) divisible by 3 (ii) $\equiv 1 \pmod 4$ (iii) $2 \pmod 5$
$\equiv 0 \pmod 3$

(i)



(ii) $\equiv 1 \pmod 4$



δ	0	1
q_0	q_0	q_1
q_1	q_2	q_3
q_2	q_0	q_1
q_3	q_2	q_3

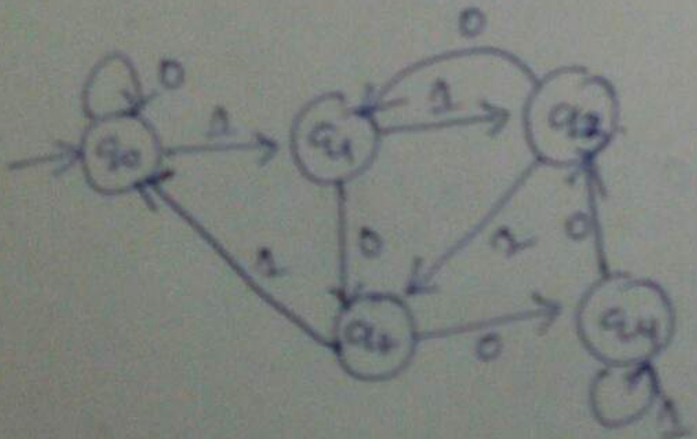
$q_0 = q_2$

$q_1 \neq q_3$

$\neq$ NF

δ	0	1
q_0	q_0	q_1
q_1	q_2	q_3
q_2	q_0	q_1
q_3	q_2	q_4

(iii) $\equiv 2 \pmod 5$



No of min MFA = 5

2152

Generated by Smart Document Scanner https://bit.ly/SmartScanner

→ Construct

— No of states in MFA that accept an binary strings whose integer equivalent to $\cong r \pmod n$

$\cong 1 \pmod{10} = \frac{10}{2} = 5 \Rightarrow 5+1=6$

$\cong 2 \pmod{12} = \frac{12}{\hat{0}} = \frac{6}{2} \Rightarrow 3 \Rightarrow 3+2=5$

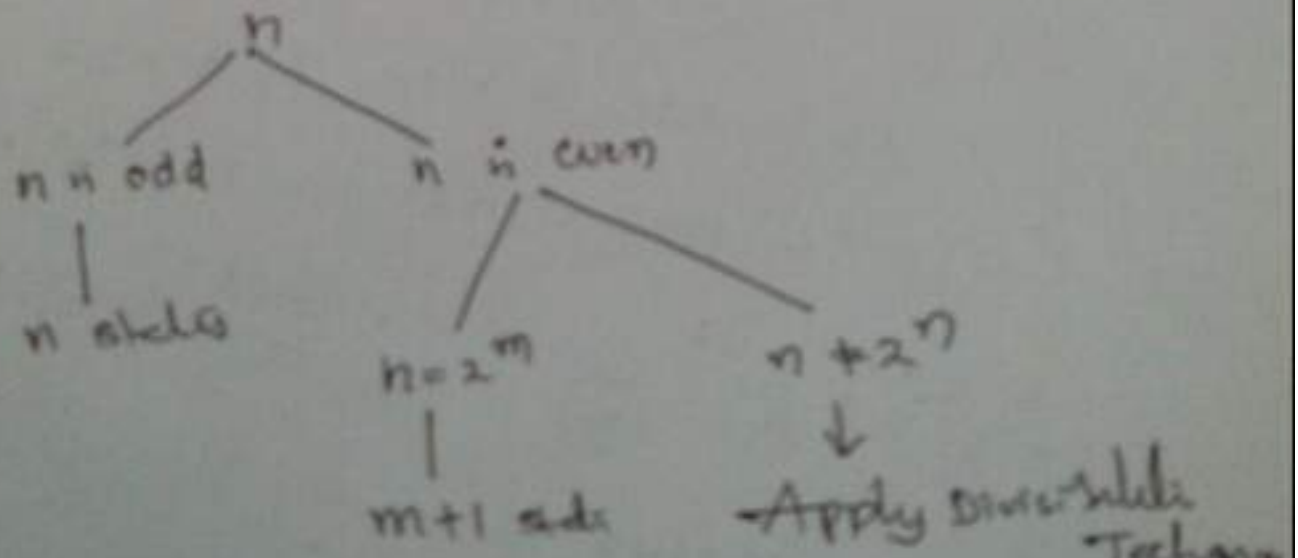
$\cong 10 \pmod{16} = \frac{16}{2} = \frac{8}{2} = \frac{4}{2} = \frac{2}{2}+1 \Rightarrow 1+4=5$

or $2^{16} = 2^4 = 4+1$

$\cong 5 \pmod{28} = \frac{28}{2} = \frac{14}{2} = 7 \Rightarrow 7+2=9$

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

Binary → integer $\cong r \pmod n$



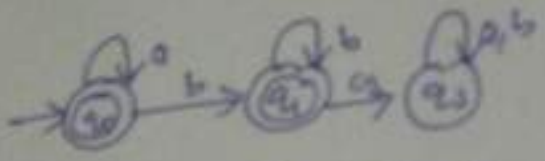
8251

Generated by Smart Document Scanner https://bit.ly/SmartScanner

→ Construct the MFA for the following language and find the no of states in MFA

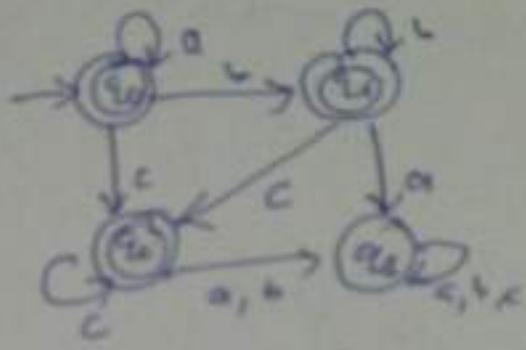
⇒ $L = \{a^m b^n \mid m, n \geq 0\}$ → $L = \{a^m b^n c^p \mid m, n, p \geq 0\}$

$L = \{\epsilon, a^m, b^m, a^m b^n ...\}$ $L = \{\epsilon, a^m, b^n, c^p ...\}$



$\Sigma = \{a, b\} \Rightarrow |\Sigma| = 2$

no of states $= |\Sigma| + 1 = 3$



$\Sigma = \{a, b, c\} = 3$

$|\Sigma| = 3$

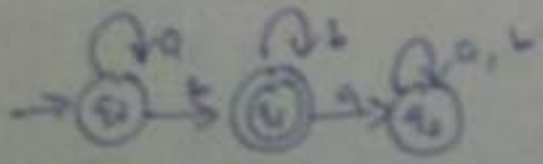
no of states $|\Sigma| + 1 = 4$

$L = \{a^{x_1} b^{x_2} c^{x_3} \ldots z^{x_{26}} \mid x_i \geq 0\}$

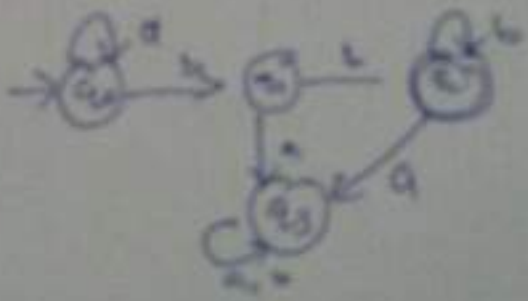
$\Sigma = \{a, b, c \ldots z\}$ $|\Sigma| = 26$

No of states in MFA $= |\Sigma| + 1 = 27$

$L = \{a^m b^n \mid m \geq 0, n \geq 1\}$ → $L = \{a^m b^n \mid m \geq 0, n \geq 2\}$



MFA states $= 0 + 1 + 2 = 3$



MFA states $= 0 + 2 + 2 = 4$

$L = \{a^{x_1} b^{x_2} \mid m \geq 0 n \geq j\}$

MFA states $= \underline{0 + j + 2}$ •

2252

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Case (iii)

$L = \{a^m b^n \mid m \geq 1, n \geq 0\}$ $L = \{a^m b^n \mid m \geq 2, n \geq 0\}$

MFA states = 1 + 0 + 3 = 4 states 2 + 0 + 3 = 5 states

$L = \{a^{m+1} b^{n+1} \mid m \geq 1, n \geq 0\}$

states = 1 + 0 + 3

Case IV:-

$L = \{a^m b^n \mid n \geq 1, n \geq 1\}$ $L = \{a^m b^n \mid m \geq 1, n \geq 2\}$

states = 1 + 1 + 2 = 4 states = 1 + 2 + 2 = 5

$L = \{a^m b^n \mid m \geq 2, n \geq 3\}$

states = 2 + 3 + 2 = 7

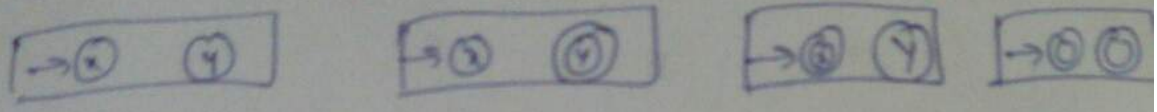
2351

Generated by Smart Document Scanner https://bit.ly/SmartScanner

⇒ How many two state DFA's will designated initial state can be constructed over the alphabet

$\Sigma = \{0, 1\}$ $Q = \{X, Y\}$ (X: initial state)

0 Final state | 1 FS | 1 FY | 2 FS



δ	0	1
→X	X2 / Y1	X2
Y	X2	X2

2^4 = 16 ways

Total.

16 × 4 = 64 ways

⇒ no of 3-state DFA's will designated initial state can be constructed over the alphabet $\Sigma = \{0, 1\}$

$\Sigma = \{0, 1\} \Rightarrow m = 2$

$Q = \{X, Y, Z\}$ $n = 3$ (X: Is)

0 Final state = 1

1 " " = 3

2 " " = 3

3 " " = 1

$2^n = 2^3 = 8$.

δ	0	1
X	Y3	Y3
Y	Y3	Y3
Z	Y3	Y3

$3^{2\times3} = 3^6$

Total cases = 8

Total no of DFA = 8×3^6

$= 2^3 \times 3^6 = 5832$

Shortcut:

$|\Sigma|$ = no of symbols = m

$|Q|$ = no of states = n

no of types = 2^n $\boxed{{}^nC_0 + {}^nC_1 + {}^nC_2 + \cdots + {}^nC_n = 2^n}$

no of DFA's of each type = $n^{m \cdot n}$

2351

Generated by Smart Document Scanner https://bit.ly/SmartScanner

→ no of DFA's of 8 states with designated final state can be constructed over the alphabet of 3 symbols with atleast '4' final states.

$n = 8$

$m = 3$

$$\text{no of types} = {}^8C_4 + {}^8C_5 + {}^8C_6 + {}^8C_7 + {}^8C_8$$

$$\text{no of DFA's of each type} = n^{mn} = 8^{3\times 3} = 8^{24}$$

$$\text{no of total DFA's} = ({}^8C_4 + {}^8C_5 + {}^8C_6 + {}^8C_7 + {}^8C_8) * 8^{24}$$

→ no of DFA of 6 states will designated initial state can be constructed over the alphabet of 3 symbols will Atmost 3 final states

$n = 6$

$m = 3$

$$\text{no of types} = {}^6C_0 + {}^6C_1 + {}^6C_2 + {}^6C_3$$

$$\text{no of DFA of each type} = n^{mn} = 6^{18}$$

$$\text{Total no of DFA} = {}^4C_0 * 6^{18}$$

1 — 2^0

1 1 — 2^1

1 2 1 — 2^2

1 3 3 1 — 2^3

1 4 6 4 1 — 2^4

1 5 10 10 5 1 — 2^5

1 6 15 20 15 6 1 — 2^6

${}^6C_0$ ${}^6C_1$ ${}^6C_2$ ${}^6C_3$ ${}^6C_4$ ${}^6C_5$ ${}^6C_6$

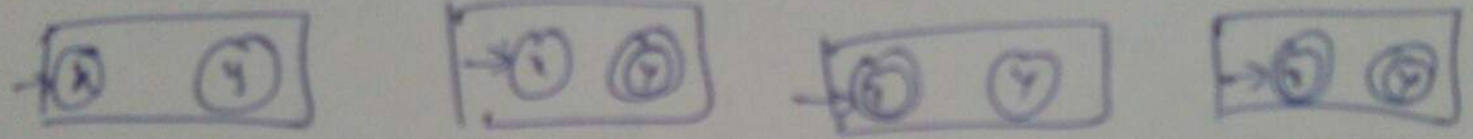
R459

Generated by Smart Document Scanner https://bit.ly/SmartScanner

⇒ no of 2 state DFA will Designated initial state can be constructed over the alphabet $\Sigma = \{0,1\}$ that accept empty language

$n = 2$
$m = 2$

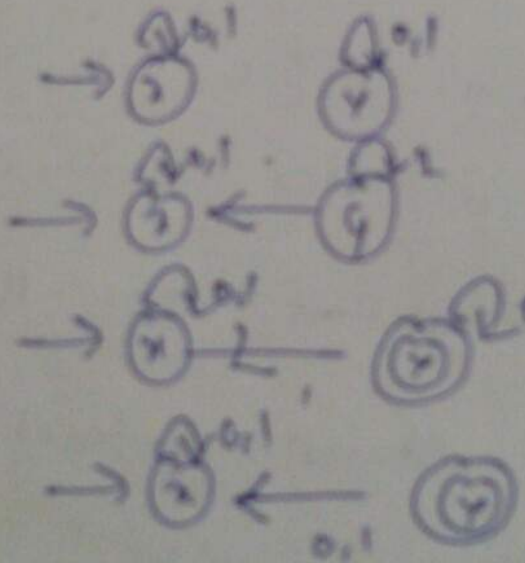
0 FS | 1 FS | 1 FC | 2 fs



16 | 16 | 16 | 16

16 +

$2^{8 \times 2} = 2^{16}$



16 + 4 = 20 st

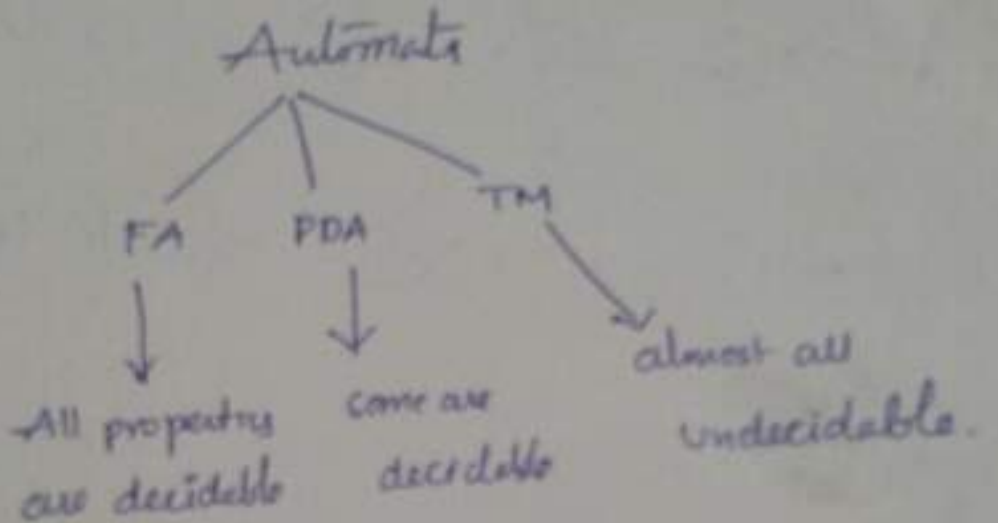
⇒ that accept Σ^* (universal language

8452

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Decision properties of FA

- Emptyness
- finiteness
- equalness
- Membership



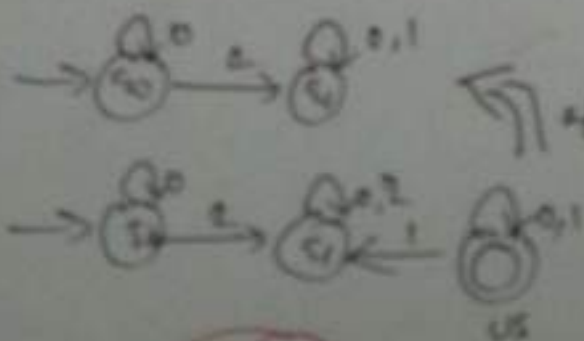
Emptyness:

A FA free from unreachable states contains atleast one Final state is "non-empty" language.

(or)

a FA will no unreachable & no Final state is accepts empty language.

Ex:



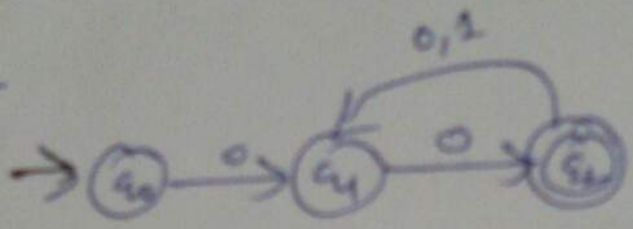
2551

Generated by Smart Document Scanner https://bit.ly/SmartScanner

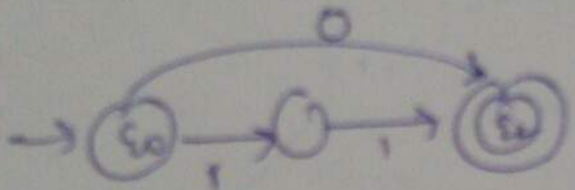
Finiteness :-

A FA free from unreachable & dead state and contains no cycles or loops is called as finiteness

Ex:



⇒ infinite language



⇒ finite language

Membership

2532

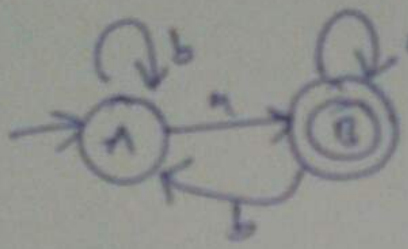
Generated by Smart Document Scanner https://bit.ly/SmartScanner

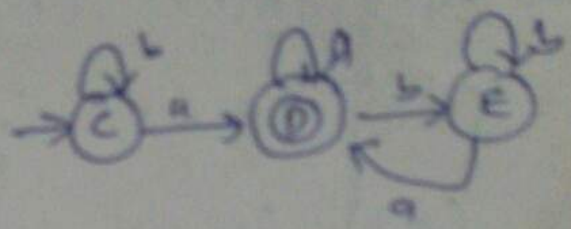
4. Equalness

two FSM's M_1 & M_2 are said to be equal

if $L(M_1) = L(M_2)$

Ex:

$M_1 =$ 

$M_2 =$ 

M_1 & M_2 are equal.

→ No need of same no of states

→ Same language & same no of states then isomorphic

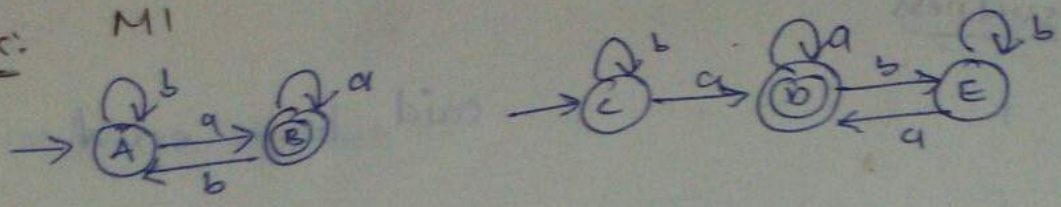
→ isomorphic are equal but equal machines may or may not be isomorphic.

⇒ Comparision Alg

8651

Generated by Smart Document Scanner https://bit.ly/SmartScanner

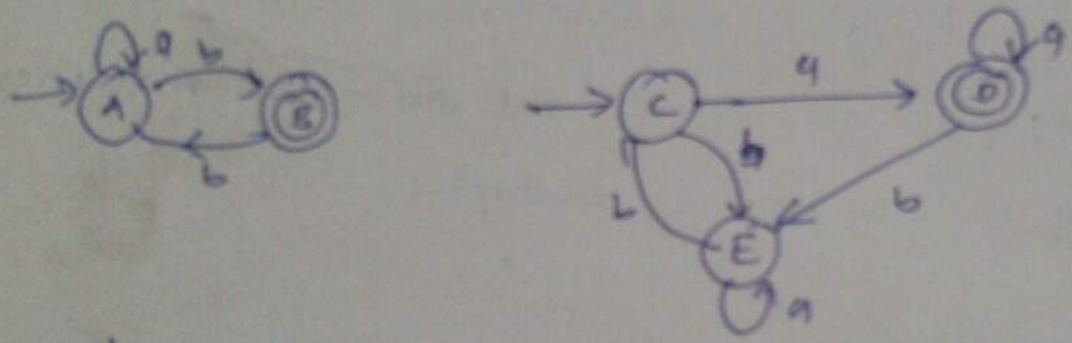
Ex: M1



δ	a	b
→ (A,C) S S	(B,D) F F	(A,C) NF NF
(B,D)	(B,D) F F	(A,E) NF NF
(A,E)	(B,D) F F	(A,E) NF NF

no new state ∴ $M_1 \neq M_2$ equal.

Ex: M_1 M_2



δ	a	b
(A,C)	(A,D) NF, F	

not equal.

→ The relation equality of two states is "Reflexive" Symmetric, and transitive: Hence it is a equivalence relation

∴ the set of all the states can be partition into finite no of mutually disjoint subset and each subset is called as equivalence class

$Q = \{q_0, q_1, q_2, q_3, q_4\}$ $\{(q_0, q_4), (q_1, q_2, q_4)\}$

2652

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Minimization of FA

The process of deletion and elimination of the states whose presence (or) absence will not effect the language as well as FA is called minimization.

→ free from unreachable & equal states

→ MFA is unique but FA is not unique

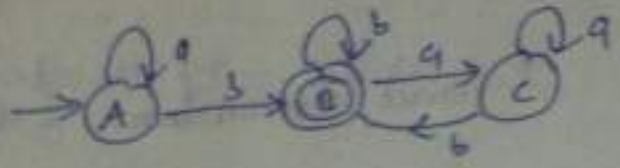
process of minimization

1. select all the equal states and remove them
   - two methods
     - states equivalence method
     - Table filling method
2. find all the unreachable states and remove them

E. Thiru

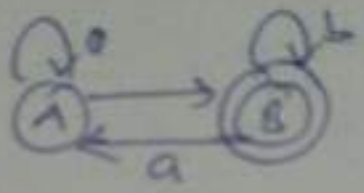
2751

Ex:



State equal Method

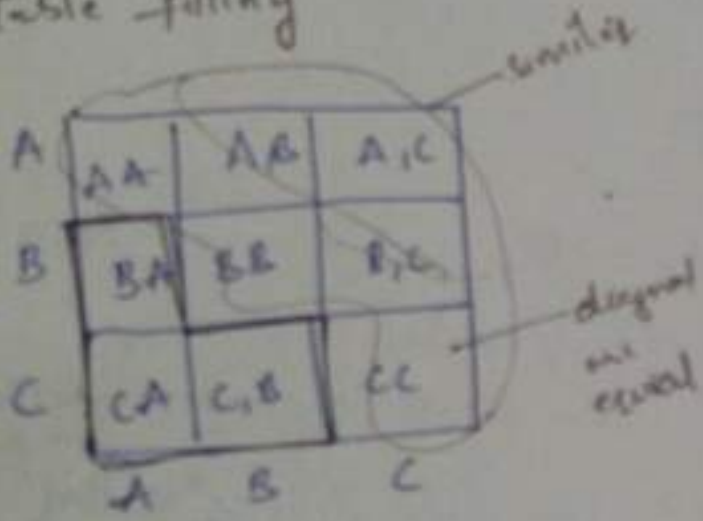
$\Pi_0 = \{(A,C), (B)\}$ ⇒

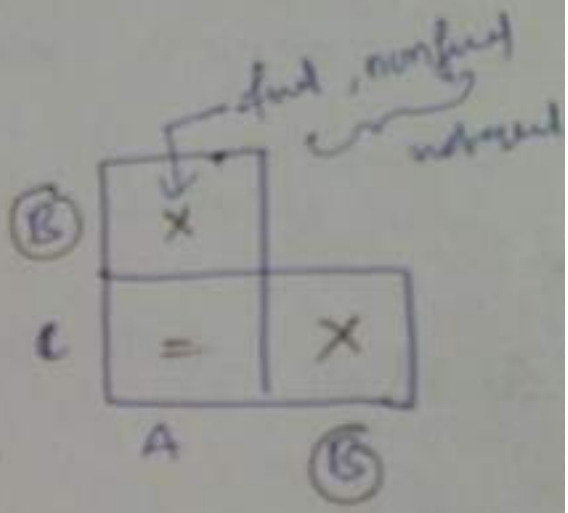


$A = C$

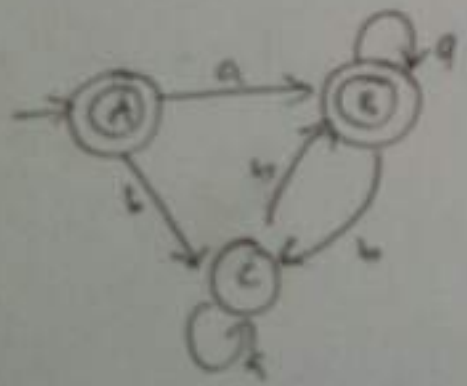
$\Pi_1 = \Pi_0$

Table filling

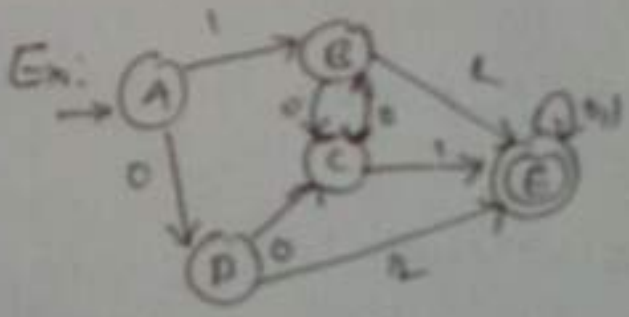




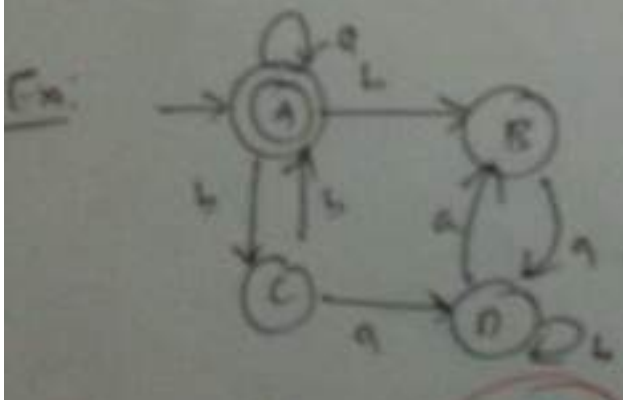
Ex:



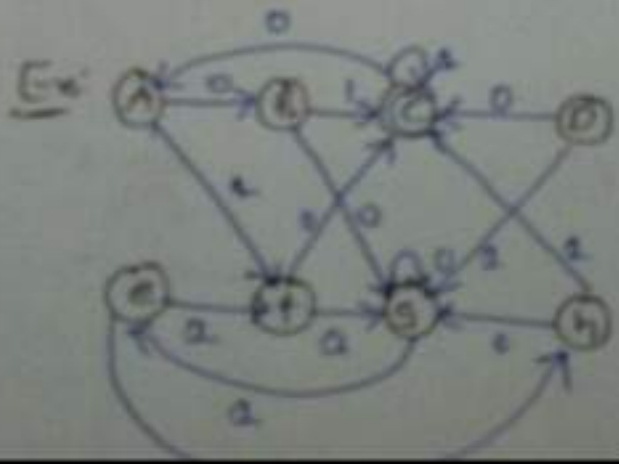
Ex:



Ex:



Ex:



Generated by Smart Document Scanner https://bit.ly/SmartScanner

Regular Expression: (RE)

a well formed formula constructed using closure, concatnation & union is called RE

Regular Operators

*, ·, + preceding order

→ If 'R' is a RE then L(R) is the language generating by R

Ex

S.NO	RE	RL
	r = ∅	L = ∅
	r = ε	L = {ε}
	r = a	L = {a}
	r = a+b	L = {a, b}
	r = a+b+c	L = {a, b, c}
	r = ab	L = {ab}
	r = (a+b)·a	L = {aa, ba}
	r = a*	L = {ε, aa ...}
	r = a+	L = {a, aa ...}

9851

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Types of RE

- Restricted RE $(*, \cdot, +)$
- semi Restricted RE $(*, \cdot, +, \cap)$
- unRestricted RE $(*, \cdot, +, \cap, \sim)$

→ RE is just like NFA (only valid i/p's)

→ If R is RE then R^*, R^+ is also RE

→ If $r = \epsilon$ then $r^* = \{\epsilon\}$ $r^+ = \{\epsilon\}$

→ If $r = \emptyset$ then $r^* = \{\epsilon\}$ $r^+ = \emptyset$

→ r^* represent infinite language except for $r = \emptyset$ or ϵ

2852

Generated by Smart Document Scanner https://bit.ly/SmartScanner

→ Construct the RE that generates all the strings of 0's and 1's $\Sigma = \{0,1\}$

(i) including ϵ $L = \Sigma^*$ $r = (0+1)^*$

(ii) Excluding ϵ $L = \Sigma^+$ $r = (0+1)^+$

(iii) starts with 10 $10(1+0)^*$

(iv) ends with 01 $(1+0)^*01$

(v) contains substring 101 $(0+1)^*101(0+1)^*$

(vi) starts and ends with 0 $0(0+1)^*0$

(vii) same symbol $0|0+0(0+1)^*0+1(0+1)^*1+1$

(viii) different symbol. $0(0+1)^*1+1(0+1)^*0$

(ix) 3rd symbol from left end is 1 $(0+1)^2 1(0+1)^*$

(x) 4th symbol from right end is 0 $(0+1)^*0(0+1)^3$

(xi) where the length of string is exactly 3
$(0+1)^3$

Atmost 3
$(0+1+\epsilon)^3$ (or) $\epsilon+(0+1)+(0+1)^2+(0+1)^3$

Atleast 3
$(0+1)^3(0+1)^*$

even $([0+1]^2)^*$

odd $(0+1)([0+1]^2)^*$

$\equiv 0 \pmod 2$

2951

Generated by Smart Document Scanner https://bit.ly/SmartScanner

— starts with 11 or 00 $(11+00)(0+1)^*$

— ends with 00 or 11 $(0+1)^*(00+11)$

— contain substring 00 or 11 $(0+1)^*(00+11)(0+1)^*$

→ start with 1 and length of string is even.

$1(0+1)[(0+1)^2]^*$

→ ∴ odd

$0(0+1)^2)^*$

→ construct RE for following languages.

→ $L=\{a^n \mid n \geq 0\}$ a^*

→ $L=\{a^n \mid n \geq 1\}$ $a^+ \Rightarrow aa^*$

→ $L=\{a^m b^n \mid m,n \geq 0\}$ a^*b^*

$L=\{a^m b^m \mid m,n \geq 1\}$ $a^+b^+ \Rightarrow aa^*bb^*$

$L=\{a^m b^n \mid m \geq 0, n \geq 1\}$ $a^*b^+ \Rightarrow a^*bb^*$

$L=\{a^m b^n \mid m \geq 1, n \geq 0\}$ $a^+b^* \Rightarrow aa^*b$

$L=\{a^m b^n \mid m \geq 1, n \geq 2\} - a^+bb^+ \Rightarrow aa^*bbb^*$

$L=\{a^m b^n c^p \mid m,n,p \geq 0\}$ $a^*b^*c^*$

$L=\{a^m b^n c^p \mid m,n,p \geq 1\}$ $a^+b^+c^+ \Rightarrow aa^*bb^*cc^*$

2/9/21

Generated by Smart Document Scanner https://bit.ly/SmartScanner

→ construct RE for the following

$L = \{a^m b^n \mid m+n = \text{even}\}$

$R = (aa)^*(bb)^* + (aa)^* a (bb)^* b$

$L = \{a^m b^n \mid m+n = \text{odd}\}$

$(aa)^* a (bb)^* + (aa)^* (bb)^* b$

→ ~~Regular~~ $\Sigma = \{a, b\}$ start with 'a' and no consecutive two b's

$r = (a + ab)^*$ ~~or~~ $a(\epsilon + ab)$

→ do not contain 'bb' or 'aa'

$r = (b+\epsilon)(ab)^*(a+\epsilon)$

or

$(a+\epsilon)(ba)^*(b+\epsilon)$

→ no of a's in string

exactly 2 $b^* a b^* a b^*$

atmost 2 $b^*(a+\epsilon) b^* (a+\epsilon) b^*$

atleast 2 $(a+b)^* a (a+b)^* a (a+b)^*$

even $(b^* a b^* a b^*)^*$

odd $b^* a b^* (b^* a b^* a b^*)^*$

$\equiv 0 \bmod 3$ $(b^* a b^* a b^* a b^*)^*$

3051

Generated by Smart Document Scanner https://bit.ly/SmartScanner

1. $\emptyset + R = R$

2. $\emptyset . R = R . \emptyset = \emptyset$

3. $\in R = R . \in = R$

4. $\wedge^* = \wedge \quad \emptyset^* = \wedge$

5. $R + R = R$

6. $R^* R^* = R^*$

7. $(R^*)^* = R^*$

8. $R R^* = R^* R = R^+$

9. $\wedge + R R^* = \wedge + R^* R = R^*$

10. $(PQ)^* P = P(QP)^*$

11. $(P+Q)^* = (P^* Q^*)^* = (P^* + Q^*)^* = (P^* + Q)^* = (P + Q^*)^*$

12. $r = \in$ then $r^* = \{\in\} \quad r^+ = \{\in\}$

13. $r^+ = r^*$ then $r = \in$

14. $r = \emptyset$ then $r^* = \{\in\} \quad r^+ = \{\emptyset\}$

15. $r^{***\cdots*} = r^* \qquad r^{+++\cdots+} = r^+$

16. $(\wedge + R)^* = R^*$

17. $R + S = S + R \rightarrow$ commutative

$(R+S)+T = R+(S+T) \rightarrow$ Associative

$R.(S+T) = (RS).T$

$R(S+T) = RS + RT$

$(R+S)T = RT + ST$

(3151)

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Algebraic properties of RE

- closure property : w.r.t $(+, \cdot, *)$ If r is r.e then $r^*, r+r, r \cdot r$
- Associative property $(r_1 \cdot r_2) r_3 = r_1 (r_2 \cdot r_3)$ $(r_1 + r_2) + r_3 = r_1 + (r_2 + r_3)$
- identity $r + \phi = r$, $r \cdot \epsilon = r$, ϕ & ϵ are identities
- Annilator $r \cdot \phi = \phi$ 'ϕ' is annilator w.r.t '$\cdot$', not for '$+$'
- idempotent $r + r = r$, but $r \cdot r \neq r$
- commutative $r_1 + r_2 = r_2 + r_1$ $r_1 \cdot r_2 \neq r_2 \cdot r_1$
- Distributive $(r_1 + r_2) \cdot r_3 = r_1 r_3 + r_2 r_3$

  $r_1 (r_2 + r_3) = r_1 r_2 + r_1 r_3$

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Conversion of FA to RE

- ARDEN's Lemma
- State elimination method

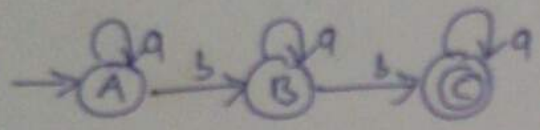
Arden's Lemma

- only for DFA & NFA

- if P, Q, R be the 3 RE's such that $R = Q + RP$ then $R = QP^*$ if P is free from ϵ

→ $R = Q + RP$ has infinite solutions if P contains 'ϵ'

Ex:



$A = Aa + \epsilon$

$R = RP + Q$

$A = \epsilon a^* = a^*$

$B \Rightarrow Ab + Ba$

$C = Bb + Ca$

$B = a^*b + Ba$

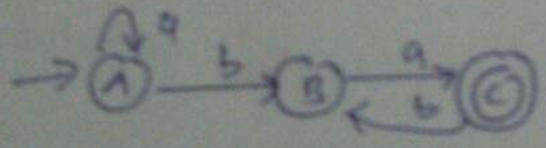
$R \quad Q \quad RP$

$B = a^*ba^*$

$\epsilon = a^*ba^*b + Ca$

$R = \quad Q \quad RP$

$C = a a^*ba^*b a^*$



$A = \epsilon + Aa = a^*$

$B = Ab + Cb = a^*b + Cb$

$C = Ba$

$B = a^*b + Bab$

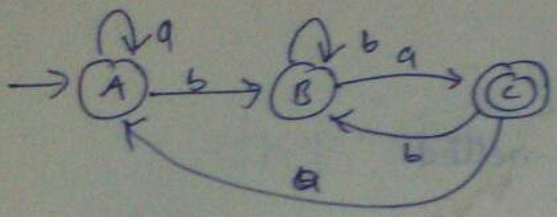
$B = a^*b(ab)^*$

$C = Ba$

$C = a^*b(ab)^*a$

3251

Generated by Smart Document Scanner https://bit.ly/SmartScanner



$A = Aa + Ca + \epsilon$ — I

$B = Ab + Bb + Cb$ — II

$C = Ba$ — III

III in II

$B = Ab + Bb + Bab$

$B = \underset{Q}{Ab} + \underset{R}{B}\underset{P}{(b+ab)}$

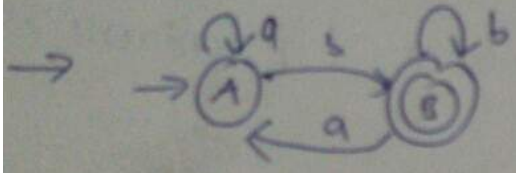
$B = Ab(b+ab)^*$

$C = Ab(b+ab)^*a$

$A = Aa + Ab(b+ab)^*a + \epsilon$

$= \underset{R}{A}\underset{P}{(a + b(b+ab)^*a)} + \underset{Q}{\epsilon}$

$= (a + b(b+ab)^*a)^*$



$A = Aa + Ba + \epsilon$

$B = Ab + Bb$

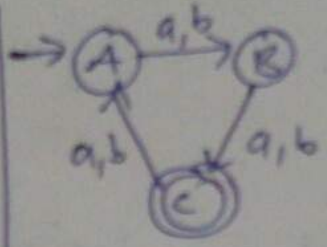
$B = Abb^*$

$A = Aa + Abb^*a + \epsilon$

$= A(a + bb^*a) + \epsilon$

$= (a + bb^*a)^*$

$B = (a + bb^*a)^* bb^*$



→ if DFA contains non-productive states Remove and · non-productive states from DFA and obtained

Drawbacks

→ can't be applied for ε-NFA

→ finding language at each and every non final state

3259

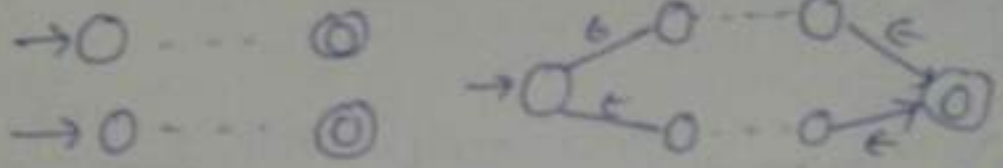
Generated by Smart Document Scanner https://bit.ly/SmartScanner

State elimination method

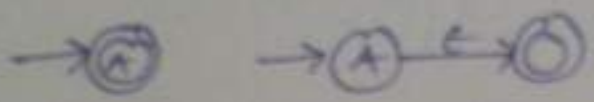
→ both for FA and Transition graph

Algorithm

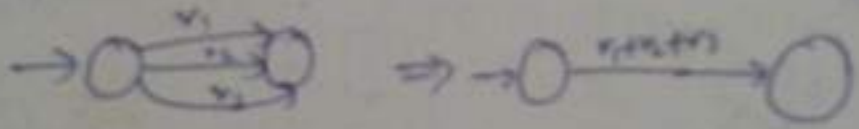
1. Simplify the FA to have only one initial state & final state



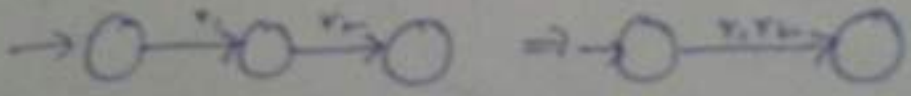
2. Simplify the transition system to have different initial and final states



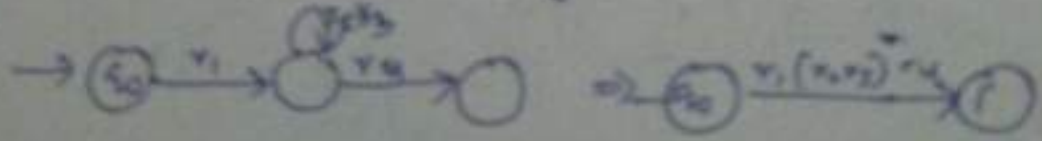
3. parallel edges can be combined using '+'



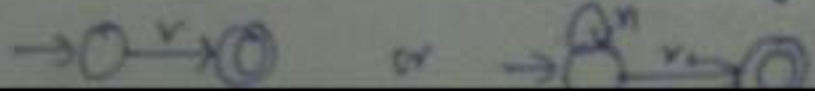
4. Sequence edges can be combined using .



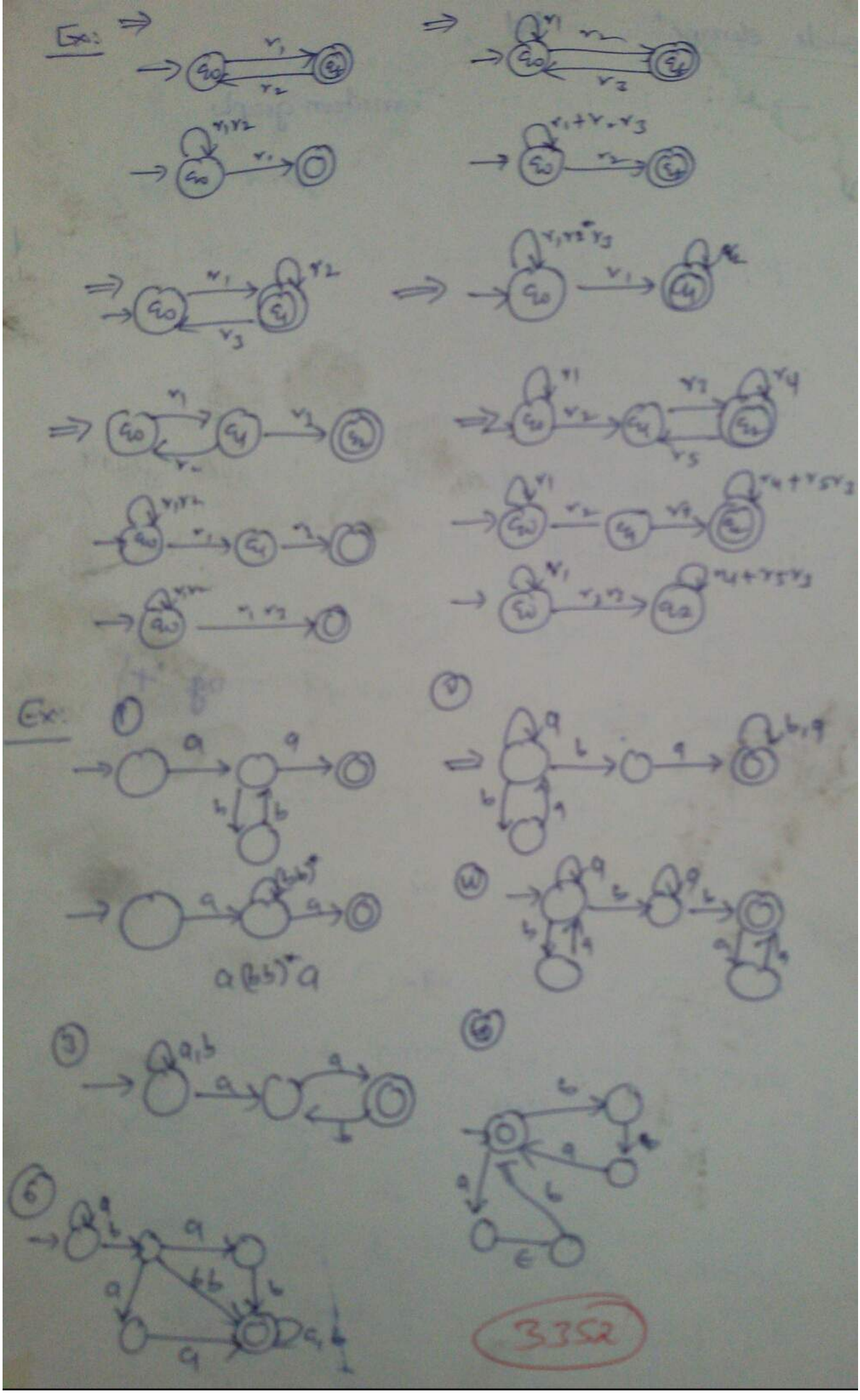
5. Remove self loop using *



6. Continue the process & state elimination until takes any one of the following



Generated by Smart Document Scanner https://bit.ly/SmartScanner



Generated by Smart Document Scanner https://bit.ly/SmartScanner

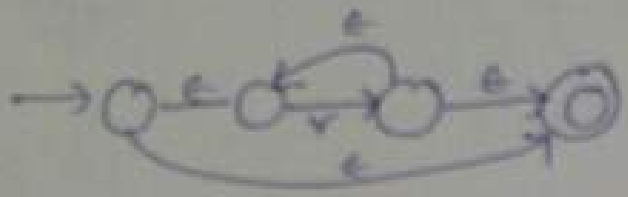
Conversion of RE to FA

— Method of Synthesis

— Method of decomposition (state creation method)

Method of Synthesis

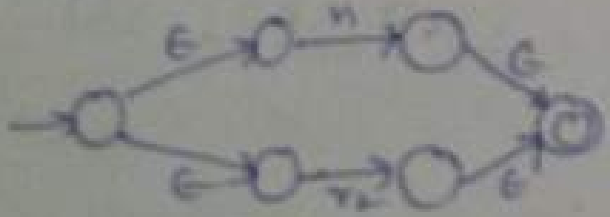
① kleen closure (r^*)



② concatenation ($r_1 r_2$)



③ union ($r_1 + r_2$)



Ex: ① $r = (ab)^*$ ② $r = a^* b^*$ ③ $a^* + b^*$ ④ $ab^* + (ba)^*$

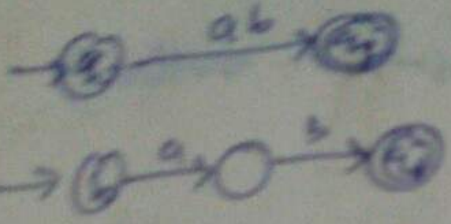
⑤ $((a^*b + b^*a)^* (a^*b)^*)$ ⑥ $(a + ba)^* b$

3US1

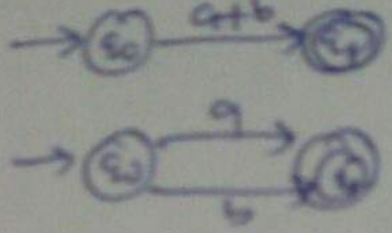
Generated by Smart Document Scanner https://bit.ly/SmartScanner

Method of decomposition

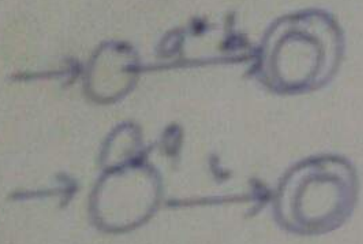
(1) $r = ab$



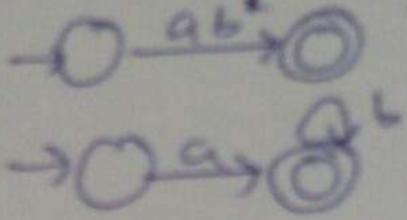
(2) $r = a + b$



(3) a^*b



(4) ab^*



(5) $r = (a+b)^*ab$.

(6) $(ab)^*ba$

(7) $r = a^*b(ab)^*b(a+b)^*$

(8) $r = (a+ba)^*ba^*$

(9) $(ab^*)^*ba^*b$

(10) $(a^*b)^*ba^*(ba)^*$

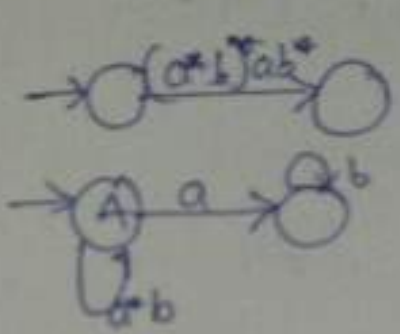
(11) $(a+ba^*)^*ba^*(b+ab)^*$

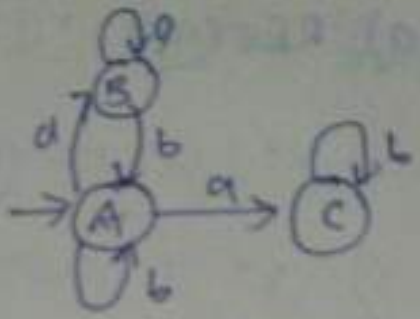
(12)

3452

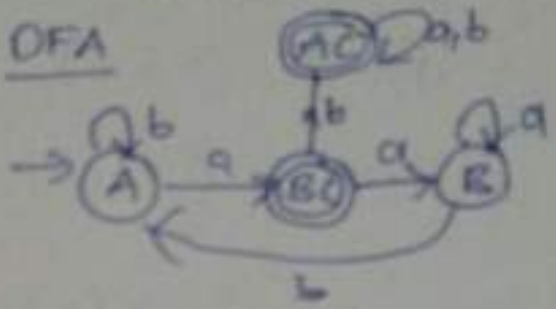
Generated by Smart Document Scanner https://bit.ly/SmartScanner

⇒ No of states in MFA to accept the language generated by R.E $r = (a^*b)^*ab^*$





DFA



→ $r = (aa + aaa)^*$

⇒ $r = (aa + aaaa)^*$ ⇒ $(aa)^*$

⇒ Which of the following R.E denotes the set of all the string not containing 100 as substring

a) $0^*(1+0)^*$ b) 0^*1010^* c) $0^*1^*01^*$ d) $0^*(10+1)^*$

⇒ $r_1 = (1^*01^*)^*$ $r_2 = (1^*01^*)^*1^*$

a) $r_1 = r_2$ b) $r_1 \subset r_2$ c) $r_2 \subset r_1$ d) none

⇒ which are equal

a) 0^* b) $(00)^*$ c) $(0+\epsilon)0^*$ d) $0(00)^*$

3551

Generated by Smart Document Scanner https://bit.ly/SmartScanner

⇒ Choose the correct stmt from the following

a) RE can represent non-RL

b) r^* is always ϵ

c) r^* is always finite

d) $r^* = (r+\epsilon)^*$

⇒ which of the following is correct

a) $r^* = r(*)$

b) $r^* = \bigcup_{i \geq 1} r^i$

c) $r^+ = \bigcup_{i \geq 0} r^i$

d) $(r+)^* = r^*$

⇒ Choose the correct statement

a) $r^* = r^+$

b) $r^* = r^+ \Leftrightarrow r = \phi$

c) $r^* = r^+ \Leftrightarrow r = \epsilon$

d) $r^* = r^+ \Leftrightarrow r = a$

⇒ Choose the correct

a) $r_1^* + r_2^* = r_1^* \cdot r_2^*$

b) $r_1^* \cdot r_2 = r_1 r_2^*$

c) $(r_1 r_2)^* = (r_1 \cdot r_2 + \epsilon)^*$

d) $(r_1 r_2)^* = (r_2 r_1)^*$

3552

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Quotient Operator

If L_1, L_2 be the ~~two~~ RL then $\frac{L_1}{L_2}$ is also a RL where $\frac{L_1}{L_2} = \{x / xy \in L_1, y \in L_2\}$

$\frac{101}{1} = 10$ $\quad X \cdot Y = 101 \quad Y = 1$

$X = 10$

$\frac{1010}{0} = 101$ $\quad \frac{1010}{01} = \not\in$ $\quad \frac{101}{101} = \epsilon$ $\frac{010}{\epsilon} = a_0$

Note: $\Rightarrow \frac{x}{\epsilon} = x \Rightarrow \frac{x}{x} = \epsilon \Rightarrow \frac{x}{y}$ exist $\Rightarrow y$ is suffix of x

$\frac{x}{y} = \not\in \Leftrightarrow y$ is not a suffix of x

Ex: $L_1 = 0^*1$

$L_2 = 1^*$

$\frac{L_1}{L_2} = 0^*1 + 0^*$

$L_2 = \epsilon \quad L_2 = 1 \quad L_2 = 11, 111, 1111 \ldots$

Ex: $L_1 = 0^*10^*$ $\quad \epsilon = 0$ $\quad 0^*1$

$L_2 = 1^*01$ $\quad \epsilon = 0$ $\quad 01$

$\frac{L_1}{L_2} = 0^*$

Ex: $L_1 = 0^*10^*$

$L_2 = 10^*$

$\frac{L_1}{L_2} = 0^*$

Ex: $L_1 = 0^*10^*$

$L_2 = 10^*1$

$\frac{L_1}{L_2} = \not\in$

Generated by Smart Document Scanner https://bit.ly/SmartScanner

if L is any R.E over the alphabet Σ

then a) $\frac{\Sigma^*}{L} = \Sigma^*$

b) $\frac{L}{\Sigma^*} = \text{prefix}(L)$

c) $\left.\begin{matrix}\frac{RL}{RL} \\ \frac{RL}{NRL}\end{matrix}\right\} RL$

d) $\left.\begin{matrix}\frac{NRL}{RL} \\ \frac{NRL}{NRL}\end{matrix}\right\} RL/NRL$

e) $\frac{\gamma^*}{\epsilon} = \gamma^* \quad \frac{\gamma^*}{\gamma} = \gamma^* \quad \frac{\gamma^*}{\gamma^*} = \gamma^*$

$\frac{\gamma^+}{\gamma^*} = \gamma^* \quad \frac{\gamma^*}{\gamma^+} = \gamma^* \quad \frac{\gamma^+}{\gamma^+} = \gamma^*$

$\frac{\gamma}{\gamma^*} = \{\epsilon, \gamma\}$

3-751

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Substitution

Let Σ and Δ be the two alphabets then substitution is a function $\delta : \Sigma \Rightarrow \Delta$ where the symbols of Σ is replaced by a R.L of another alphabet Δ

$\Sigma = \{a, b\}$ $\because$ $\delta : \Sigma \to \Delta$

$\delta(a) = L$
$\downarrow$ $\downarrow$
Σ R.L from Δ

Ex: $\Sigma = \{0, b\}$ $\Delta = \{0, 1\}$

$S(a) = 0^*$
$S(b) = 01^*$

$\Rightarrow \delta(0) = \phi$ $\delta(\epsilon) = \epsilon$ $\delta(x^*) = [\delta(x)]^*$

$\delta(x_1 + x_2 + x_3 + \cdots + x_n) = \delta(x_1) + \delta(x_2) + \cdots + \delta(x_n)$

$\delta(x_1 x_2 x_3 \cdots x_n) = \delta(x_1) \cdot \delta(x_2) \cdots \delta(x_n)$

Ex: $\Sigma = \{0, 1\}$ $\Delta = \{a, b\}$

$\delta(0) = a^*$
$\delta(1) = ab^*$

① $L = \{010\}$

$\delta(L) = \delta(010)$
$= \delta(0)\,\delta(1)\,\delta(0)$
$= a^* ab^* a^*$

② $L = \{01, 10, 11\}$

$\delta(L) = \{a^* ab^* + ab^* a^* + ab^* b^*\}$

③ $L = 01^*$ ④ $(0^*1)^*$

$a^*(ab^*)^*$ $(a^* ab^*)^*$

2752

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Homomorphism

Homomorphism is the kind of substitution from Σ to Δ ($\delta: \Sigma \to \Delta$) where the symbols of '$\Sigma$' is replaced by single string of another alphabet 'Δ'

$h: \Sigma \to \Delta$ such that

$h(x) = y$

$x \in \Sigma$ $\quad y \in \Delta^*$

$\Sigma = \{0,1\}$ $\quad \Delta = \{a,b\}$

$h(0) = a$
$h(1) = ab$
} homomorphism

$h(0) = a$
$h(1) = a \cup b$
$\{a, b\}$
} substitution

Ex: $L = \{010, 00\}$ $\quad h(L) = \{aaba, aa\}$

$\Sigma = \{0,1\}$ $\quad \Delta = \{a,b\}$
$h(0) = a$ $\quad h(1) = ab$

(2) $L = 0^*1$
a^*ab

(3) $L = (01^* + 0^*)^*$
$= (a(ab)^* + a^*)^*$
$= (a(ab)^* + a^*)^*$
$= (a(ab)^*)^*$

(4) $(01^*)^*0^*$
$= (a(ab)^*)^*a^*$

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Inverse Homomorphism

Let $h: \Sigma \to \Delta$ is homomorphism and 'L' is a RL over its alphabet 'Δ' then $h^{-1}(L)$ is also R.L over its alphabet 'Σ' where h^{-1} is called as inverse homomorphism and defined as

$$h^{-1}(L) = \{x / h(x) \in h(L)\}.$$

$$\therefore \quad x \in h^{-1}(L) \iff h(x) \in h(L)$$

$\Sigma = \{a, b\}$ $\quad \Delta = \{0, 1\}$

$h(a) = 0$

$h(b) = 01$

(i) $L = \{01, 101\}$ (ba)

$h^{-1}(L) = \{b\}$

(ii) $L = \{01, 010\}$ (b, ba)

$h^{-1}(L) = \{b, ba\}$

(3) $L = 01^*$

$= 0, 01, 011, 0111, 01111 \ldots$ (a, b, ab, bb)

$h^{-1}(a) = \{a, b\}$ $\quad h^{-1}(L) = a + b$

(4) $L = 0^*1$

$1, 01, 001, 0001, 00001$ (ε, b, aab, bbb)

$L^{-1}(L) = a^*b$

3852

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Moore Machine:

FA where the O/P is associated with the state is called as moore m/c

(or)

6 Tuple $M = (Q, \Sigma, \delta, \Delta, \lambda, q_0)$

Q = set of all states

Σ = input alphabet

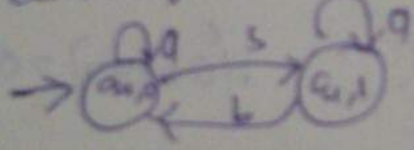
$\delta = Q \times \Sigma \to Q$ is a Transition function

Δ = output alphabet

$\lambda = Q \to \Delta$ is an op/p func

q_0 = initial state

Ex: even no of a's or odd number of b's

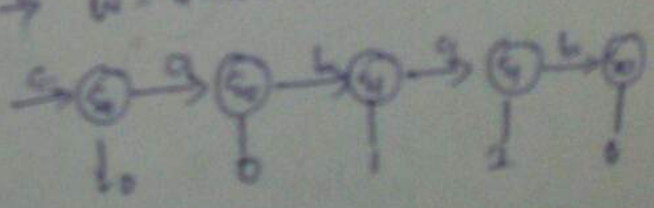


Moore M/c can be represented in two ways

- transition diagram
- transition Table

δ	a	b	O/P
→q_0	q_0	q_1	0
q_1	q_1	q_0	1

→ w = abab



λ(abab) = 01110

→ w = babba

λ(babba) = 011011

3951

Generated by Smart Document Scanner https://bit.ly/SmartScanner

→ Moore m/c responds for every string's 'ϵ'

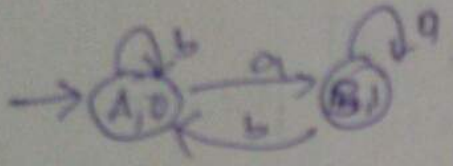
$\lambda(\epsilon) = \lambda$ (initial state)

→ If $|w| = n$ o/p = n+1 bits

Problem on moore m/c

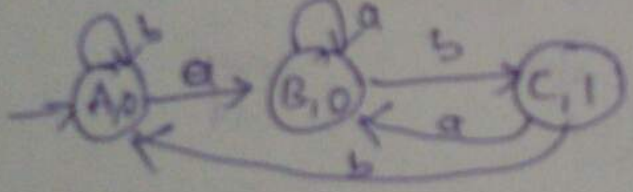
→ $\Sigma = \{a, b\}$

→ count the no of a's in the given i/p string



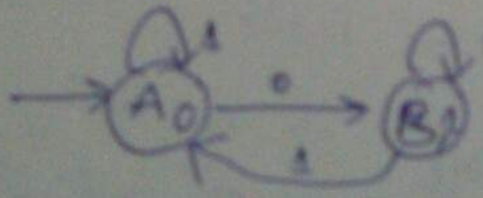
$\lambda(\overset{\epsilon}{a}baaab) = 0\underline{1}0\underline{11}0 = 3$

→ no of occurance of string ab



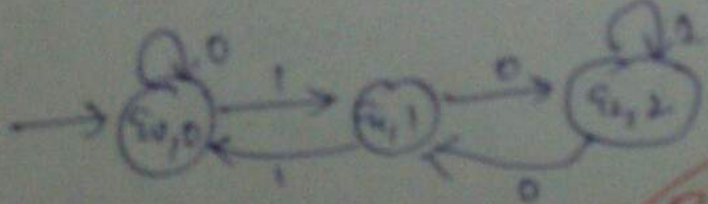
$\lambda(\overset{\epsilon}{a}baaab) = 001001$

→ Construct moore m/c to find it's complement of binary number



$\lambda(\epsilon 0100) = \overset{\epsilon}{1}1011$

→ modulus 3 → module 4



3952

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Mealy m/c

FA where the output is associated with the transition is called as mealy machine

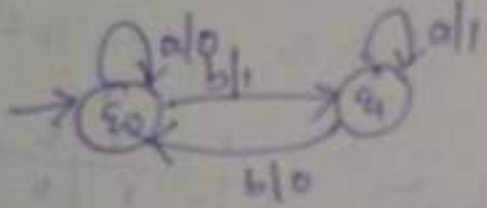
(or)

6-tuple $M = (Q, \Sigma, \delta, \Delta, \lambda, q_0)$

$\delta = Q \times \Sigma \to Q$

Δ = o/p alphabet

$\lambda = Q \times \Sigma \to \Delta$ is an o/p function



Representation of Mealy m/c

- Transition diagram
- transition Table.

δ	a	λ	b	λ
$\to q_0$	q_0	0	q_1	1
q_1	q_1	1	q_0	0

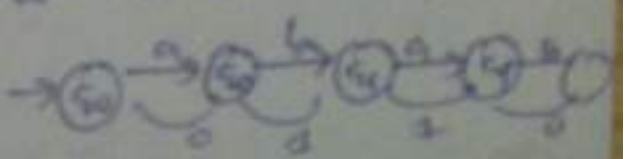
$\delta(q_0, a) = q_0$ $\quad \lambda(q_0, a) = 0$

$\delta(q_0, b) = q_1$ $\quad \lambda(q_0, b) = 1$

$\delta(q_1, a) = q_1$ $\quad \lambda(q_1, a) = 1$

$\delta(q_1, b) = q_0$ $\quad \lambda(q_1, b) = 0$

w= abab



$\lambda(abab) = 0110$

Generated by Smart Document Scanner https://bit.ly/SmartScanner

→ o/p depends on transition i.e i/p + state

→ mealy m/c doesnot respond for empty string

$\because \lambda(\epsilon) = \epsilon$

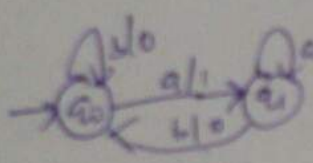
→ Length of i/p = Length of the output.

problems on mealy m/c

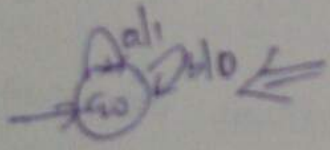
$\Sigma = \{a, b\}$

→ mealy m/c takes less no of states than moore & DFA

(i) count the no of a's in the given i/p string

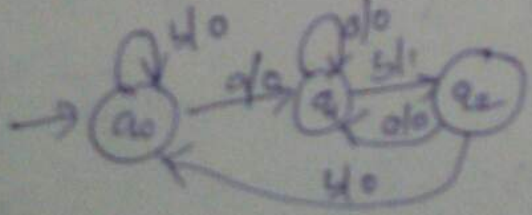


	a	λ	b	λ
→ q_0	q_1	1	b	0
q_1	q_1	1	b	0

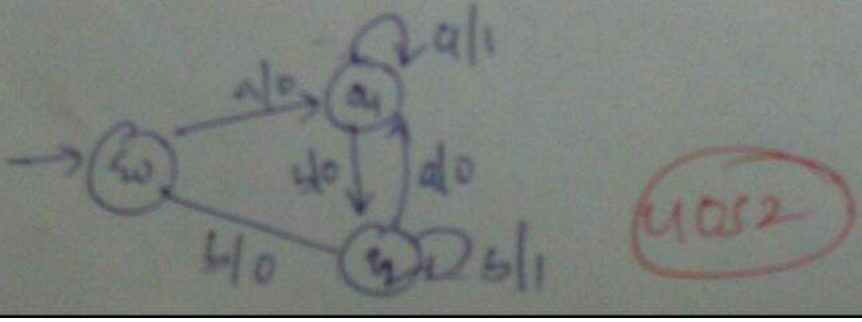


$q_0 = q_1$ are equal.

(ii) count no of occurances of ab



(iii) i/p ends with aa or bb

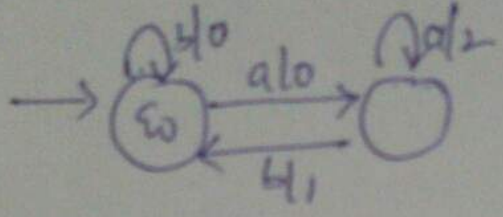


4052

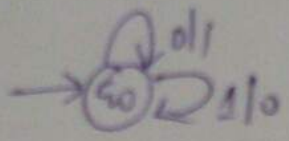
(iii) 1 – ends with ab

2 – aa

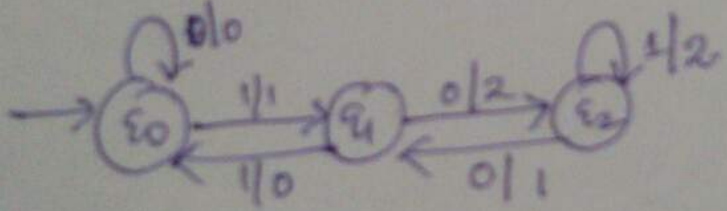
0 – otherwise



iv) complement of binary number



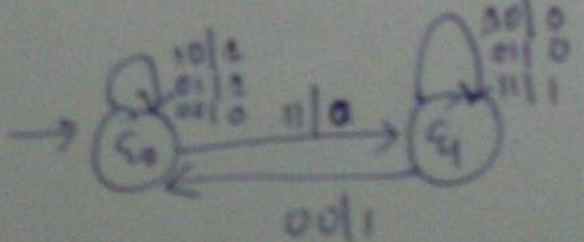
→ modulo 3



→ sum of 2 binary numbers

i/p: 00 01 10 11

carry: 0 1



4151

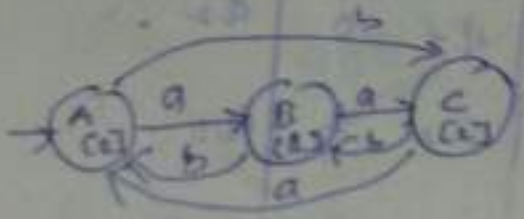
Generated by Smart Document Scanner https://bit.ly/SmartScanner

Conversion of moore to mealy

1.40

moore :

δ	a	b	λ
A	B	C	0
B	C	A	1
C	A	B	2

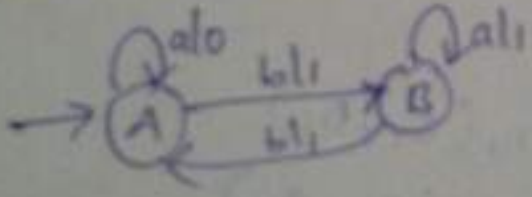


mealy :

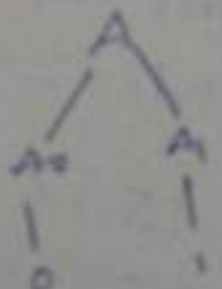
	a	λ	b	λ
→ A	B	1	C	2
B	C	2	A	0
C	A	0	B	1

→ no change in no of states when moore → mealy

Conversion of mealy to moore m/c



δ	a	λ	b	λ
A	A	0	B	1
B	B	1	A	1



B
|
a

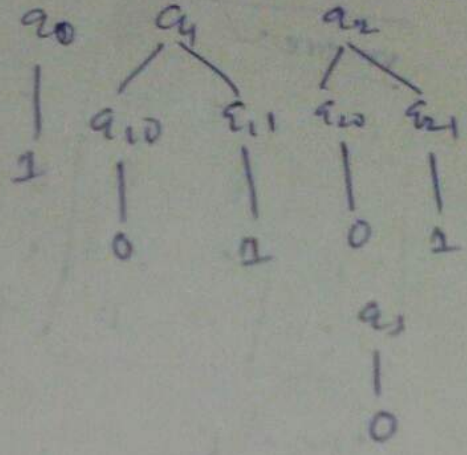
δ	a	b	λ
→ A0	A0	B	0
A1	A0	B	1
B	B	A1	1

U2S1

Generated by Smart Document Scanner https://bit.ly/SmartScanner

⇒

δ	a	1	b	λ
→ q_0	q_3	0	q_2	0
q_1	q_2	1	q_0	1
q_2	q_1	0	q_3	0
q_3	q_0	1	q_1	1



	a	b	1
→ q_0	q_3	q_{20}	0
q_{10}	q_{21}	q_0	0
q_{11}	q_{21}	q_0	1
q_{20}	q_{10}	q_3	0
q_{21}	q_{10}	q_3	1
q_3	q_0	q_{11}	0

→ more m/c → $m_1(Q_1, \Delta_1)$

mealy m/c → $M_2(Q_2, \Delta_2)$

$|Q_1| \leq |Q_2| \times |\Delta_2|$

4 4 × 2

$|Q| \times \Delta$

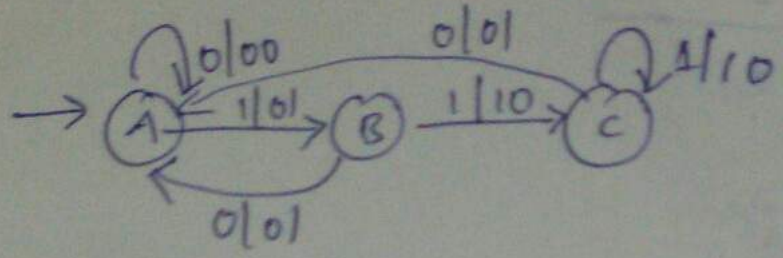
4 × 2 → Long process

minimize → 6 is also simple

4252

Generated by Smart Document Scanner https://bit.ly/SmartScanner

→ Consider the FSM



GATE 2002

a) o/p - 01 & i/p contain - 11

b) o/p - 00 & i/p 10

c) output is sum of present and previous bits

d) none

→ Consider the finite SM

P.S	X = 1	X = 0
A	D, 0	B, 0
B	B, 1	C, 1
C	B, 0	D, 1
D	B, 1	C, 0

If the initial ~~state~~ is unknown then the ~~shortest~~ i/p sequence do reach the final state is

a) 01 b) 10 c) 101 d) 110

U351

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

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

then $L_1 \cap L_2$, $L_1 \cup L_2$ is

a) RL , RL b) RL, NRL c) NRL, RL d) NRL NRL

(2) if L is any RL accepted by FA without dead states will m states and prefix of L is accepted by FA with n states then

a) $m \leq n$ b) $m \geq n$ c) $m = n$ d) cannot say

if dead state is then then $m = n$

(3) if L is any RL without dead states will m states then no of states in NFA to accept prefix of L is

a) 1 b) m+1 c) m+2 d) m^2

— will dead state

(4) no of strings of length ≤ 3 generated by the RE $(0+10)^*0(0+1)^*$ a) 8 b) 7 c) 6 d) none

$L=0 \rightarrow 0$

$L=1 \rightarrow 1$ {0}

$L=2 \rightarrow 2$ {00, 01}

$L=3 \rightarrow 5$ {000, 001, 010, 01, 100}

Generated by Smart Document Scanner https://bit.ly/SmartScanner

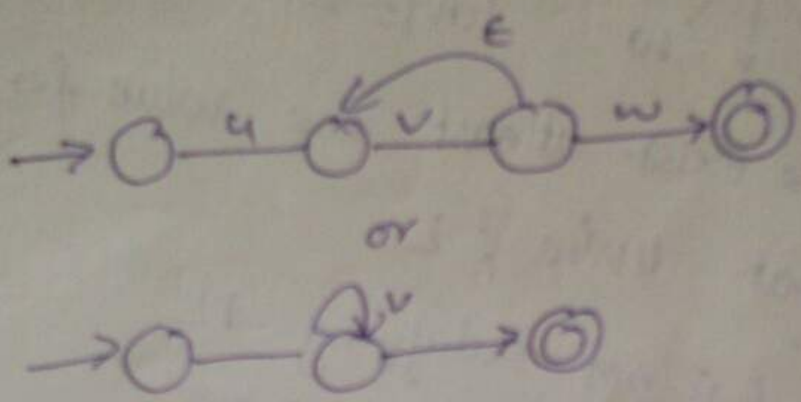
Pumping Lemma

1.43

If L is a RL (infenite) and $z \in L$ such that $|z| \geq n$ then $uv^i w \in L \ \forall \ i \geq 0$ where

$z = uvw$, $|uv| \leq |z|$ $|v| \neq 0$

n = no of states in DFA



→ pumping Lemma is used to prove that some of its tanguages are non Regular

→ every RL statisfies PL

→ The language which doesn't statisfy PL is not RL

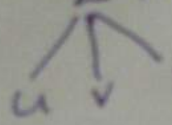
→ . . . statisfy PL may or may not RL.

USSI

Process of PL

Given that L is a NRL

1. Assume that L is regular
2. Choose $z \in L$ $|z| \geq n$
3. Split z into 3 parts



$|uv| \leq z$ & $|v| \neq 0$

4. If there exist atleast one value for 'i' such that $uv^iw \notin L$

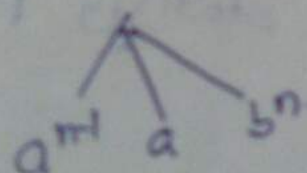
L is not satisfying PL

which is a contradiction

5. L is NRL

Ex: Prove that $L = \{a^n b^n \mid n \geq 1$ is a NRL$\}$.

$z = a^n b^n$ $k \geq 1$ (k → no of states)



$uv^1w = a^{n-1}a^1b^n$ if $i = 2$

$= a^{n+1}b^n \notin L$

$\therefore$ L is a NRL

4652

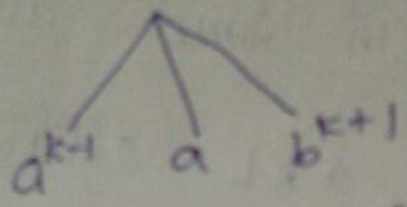
Generated by Smart Document Scanner https://bit.ly/SmartScanner

⇒ Prove that $L = \{a^m b^n \mid m<n\}$ is a NRL

$L = \{a^m b^n \mid m<n\}$

Let $z \in L$

$z = a^k b^{k+1}$ $\quad |z| = |2k+1| > n$



$uv^i w = a^{k-1} a^i b^{k+1}$ $\quad i = 2$

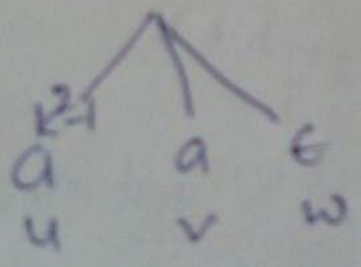
$= a^{k+1} b^{k+1} \in L$

$\therefore L$ is a non RL.

⇒ $L = \{a^{n^2} \mid n \geq 0\}$

Let $z \in L$

$z = a^{k^2}$



$uv^i w = a^{k^2-1} a^1$ $\quad i = 2$

$= a^{k^2+1} \notin L$

$\therefore L$ is NRL.

4751

Generated by Smart Document Scanner https://bit.ly/SmartScanner

Weak form of PL

If L is any language defined over the alphabet with only one symbol such that the length of strings of L are in some arithmatic progression then L is a RL otherwise it is NRL

Ex ⇒ $L = \{1^{2n} \mid n \geq 0\}$ $\Sigma = \{1\}$ — RL

0, 2, 4, 6, 8, 10 ... in AP

⇒ $L\{0^{3n+1} \mid n \geq 1\}$ $\Sigma = \{0\}$ — RL

4, 7, 10, 13 ... in AP

⇒ $L = \{a^{n^2} \mid n \geq 0\}$ — NRL

0, 1, 4, 9, 16 ... Not in AP

⇒ $L = \{a^{n+1} \mid n \geq 0\}$ — NRL

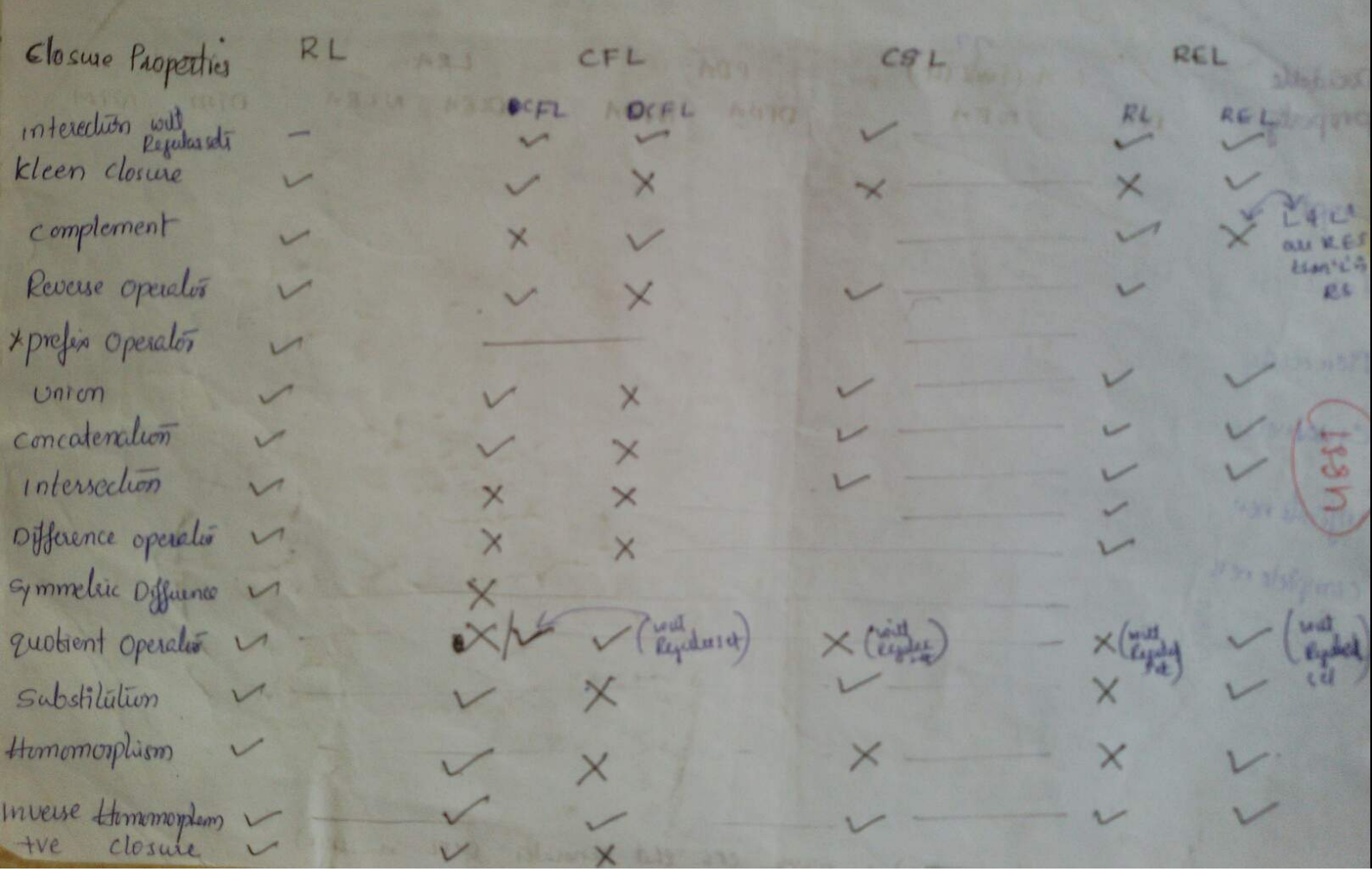
⇒ $L = \{a^{2n} \mid n \geq 0\}$ — NRL

⇒ $L = \{a^{3n+2} \mid n \geq 0\}$ — NRL

⇒ $L = \{a^{n!} \mid n \geq 0\}$ — NRL

$L = \{a^{n^{1/2}} \mid n \geq 0\}$ — NRL

Generated by Smart Document Scanner https://bit.ly/SmartScanner



Generated by Smart Document Scanner https://bit.ly/SmartScanner

Decidable Properties

	finite set	CFL	CSL	Recursive set	RE
1. Membership	✓	✓	✓	✓	✗
2. Emptiness	✓	✓	✗	✗	✗
3. finiteness	✓	✓	✗	✗	✗
4. Infiniteness	✓	✓	✗	✗	✗
5. Completeness L = Σ^*	✓	✗	✗	✗	✗
6. Equivalence	✓	✗	✗	✗	✗

4951

Generated by Smart Document Scanner https://bit.ly/SmartScanner

→ every Regular Language is finite and every RRL is infinite

Generated by Smart Document Scanner https://bit.ly/SmartScanner