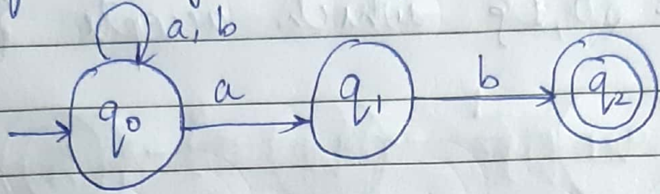
31. decimal ~~[illegible]~~ divisible by 3.

$r = 10 \quad i = 0, 1, 2 \quad d = \{0,1,2,3,4,5,6,7,8,9\} \quad K = 3.$

remainder digits.

NFA – More number of transition function (or states) for a given input alphabet.

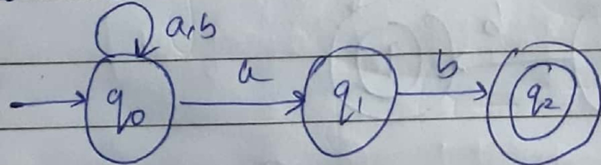
for eg:-



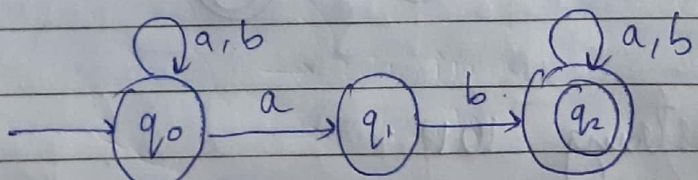
Note:- We don't have to find transition state for every input alphabet over a particular state.

Examples.

1. Construct a ~~D~~NFA with input alphabet $\Sigma = \{a, b\}$ that ends with a, b $L = \{aab, ab, bab, bbab, aaab \dots\}$.

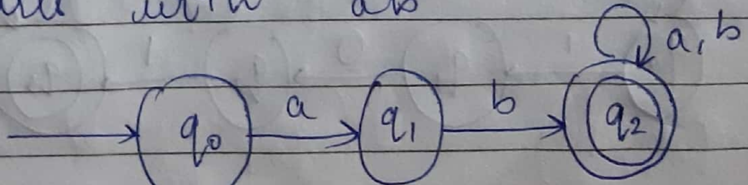


2. CNFA ~~ends~~ has substring 'ab'.



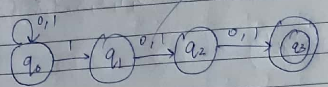
δ	a	b
q_0	$\{q_0, q_1\}$	q_0
q_1	ϕ	q_2
q_2	q_2	q_2

3. CNFA starts with 'ab'

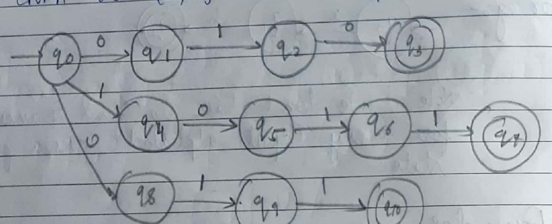


δ	a	b
q_0	q_1	ϕ
q_1	ϕ	q_2
$*q_2$	q_2	q_2

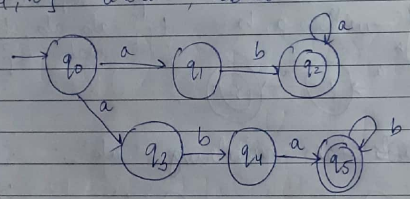
4. CNFA over $\{0,1\}$ such that 3^{rd} symbol from right end should be 1.



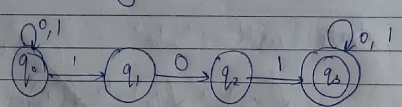
5. CNFA over $\{0,1\}$ which accepts $\{0101, 1011, 011\}$



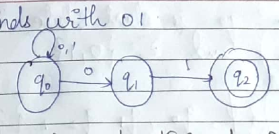
6. $\{a,b\}$ aba^n, $abab^n$



7. Having substring 101.



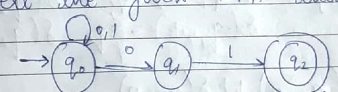
8. Ends with 01



Conversion of NFA to DFA.

1. Subset construction method
2. Lazy evaluation method.

* Convert the given NFA into its equivalent DFA using subset construction method.



1st Step: In the given NFA, the start state and input alphabet for NFA remains same in DFA.

2nd step. NFA $\rightarrow$ n states

DFA $\rightarrow$ 2^n states

3rd step Power set $\{\phi, q_0, q_1, q_2, \{q_0,q_1\}, \{q_1,q_2\}, \{q_0,q_2\}, \{q_0,q_1,q_2\}\}$.

4th step → The final state of NFA if contained in the subset of the power set, then that particular subset becomes the final state in DFA.

5th step → (If the set contains more states)

$$\delta(\{q_0,q_1\},0) = \delta(q_0,0) \cup \delta(q_1,0)$$
$$= \{q_0,q_1\} \cup \phi$$
$$= \{q_0,q_1\}$$

δ	0	1
q_0	$\{q_0, q_1\}$	q_0
q_1	ϕ	q_2
$*q_2$	ϕ	ϕ

1. q_0 is start state in DFA and $\Sigma = \{0,1\}$

2. $\{\phi, q_0, q_1, q_2, \{q_0, q_1\}, \{q_1, q_2\} \{q_0, q_2\} \{q_0, q_1, q_2\}\}$

3. Transition states :-

$\delta(q_0, 0) = \{q_0, q_1\}$ $\quad \delta(q_1, 0) = \phi$ $\quad \delta(q_2, 0) = \phi$

$\delta(q_0, 1) = q_0$ $\quad \delta(q_1, 1) = \cancel{\phi} q_2$ $\quad \delta(q_2, 1) = \phi$

$\delta(\{q_0, q_1\}, 0) = \delta(q_0, 0) \cup \delta(q_1, 0)$

$= \{q_0, q_1\} \cup \phi$

$= \{q_0, q_1\}$

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

$= q_0 \cup \cancel{\phi}\, q_2$

$= \{q_0, q_2\}$

$\delta(\{q_0, q_2\}, 0) = \delta(q_0, 0) \cup \delta(q_2, 0)$

$= \{q_0, q_1\} \cup \phi = \{q_0, q_1\}$

$\delta(\{q_0, q_2\}, 1) = \delta(q_0, 1) \cup \delta(q_2, 1)$

$= q_0 \cup \phi = q_0$

$\delta(\{q_1, q_2\}, 0) = \delta(q_1, 0) \cup \delta(q_2, 0)$

$= \phi \cup \phi = \phi$

$\delta(\{q_1, q_2\}, 1) = \delta(q_1, 1) \cup \delta(q_2, 1)$

$= q_2 \cancel{\phi} \cup \phi = q_2$

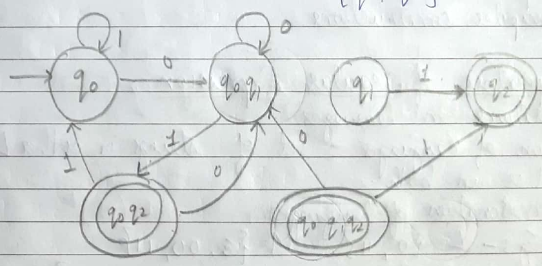
$\delta(\{q_0, q_1, q_2\}, 0) = \delta(q_0, 0) \cup \delta(q_1, 0)\ \delta(q_2, 0)$

$= \{q_0, q_1\} \cup \phi \cup \phi = \{q_0, q_1\}$

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

$= q_0 \cup \phi \cup q_2$

$= \{q_0, q_2\}$



Lazy Evaluation Method

→ Transition states :-

$\delta(q_0, 0) = \{q_0, q_1\}$

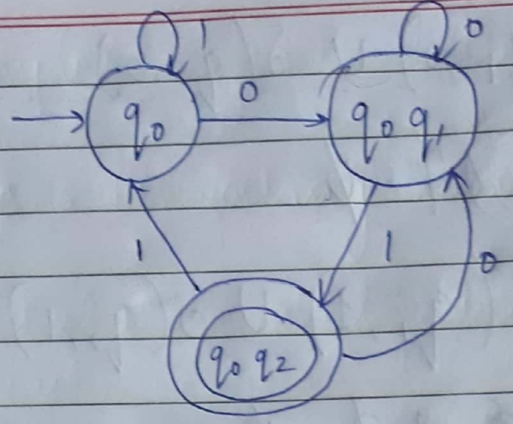
$\delta(q_0, 1) = q_0$

$\delta(\{q_0, q_1\}, 0) = \{q_0, q_1\}$

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

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

$\delta(\{q_0, q_2\}, 1) = q_0$



UNIT – II

* <u>Regular Expressions</u> → Algebraic-like description to denote Regular Language.

* <u>Applications of Regular Expressions.</u>
- UNIX.
- 'grep' command.
- Compiler constructions.

* <u>Operators of regular Expression.</u>
→ Operations on Languages.
→ Union of Languages – Set of strings in either L, or in M or in both.
→ Concatenation of languages.
  Eg :- $L = \{0, 11\}$ $M = \{\varepsilon, 00, 11\}$
  $L \cdot M = \{0, 11, 000, 011, 1100, 1111\}$
→ The closure (star or Kleene closure) of a language
  $L = \{0, 1\}$ $L^* = \{$set of all strings of any length$\}$
  $L^0 = \{\varepsilon\}$
  $L^1 = \{0, 1\}$
  $L^2 = \{00, 01, 10, 11\}$

$L = \{0, 11\}$ $L^2 = \{011, 110, 1111, 00\}$
$L^* = L^0 \cup L^1 \cup L^2 \cup L^3 \ldots$

Scanned with CamScanner

classmate
Date
Page

* Building regular expression

Basis

Regular Exp.	Language
ϵ	$\{\epsilon\}$ → The language contains empty string
ϕ	ϕ → The language is empty
a	$\{a\}$

Induction
⇒ a) If E & F are regular expression then E+F is also a regular expression denoting the union of L(E) and L(F).

$$L(E+F) = L(E) \cup L(F)$$

b) If E & F are regular expressions then E . F or EF is also regular expression denoting the contatenation of L(E) and L(F).

$$L(E.F) = L(E).L(F)$$

c) If E is a regular expression, then E^* is also a regular expression denoting the closure of L(E).

$$L(E^*) = (L(E))^*$$

d) If E is a regular exp. then (E) is also a regular expression denoting the same language as of.

$$L((E)) = L(E)$$

* Precendence of Operators → $*$, $\cdot$, $+$.

Q. When a string can be called a regular expression?

18. Any number a's followed by any number b's & followed by any number c's.

R.E = $a^* b^* c^*$

19. Atleast one 'a' followed by atleast one 'b' and atleast one 'c'.

$a a^* b b^* c c^*$

20. WARE that ~~all~~ either ends with a or bb

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

21. RE that should not end with aa.

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

22. R.E whose length is multiple of 3 / divisible by 3 /

$L = \{ w \mid |w| \bmod 3 = 0,\ w \in \{a, b\}^* \}$.

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

23. Number of a's divisible 3 ~~and~~ followed by no of b's divisible by 4.

$(aaa)^* (bbbb)^*$.

24. 10th symbol from the end is a.

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

25. WARE which has atleast 3 consecutive zero's

$(0+1)^* 000 (0+1)^*$

26. WARE that begin with ab and ends with ba.

R.E = $ab(a+b)^* ba$.

27. a's and b's whose length is either even or multiple of 3.

R.E = $(((a+b)(a+b))^* + ((a+b)(a+b)(a+b))^*)^*$

28. WARE for $L = \{ a^{2n} b^{2m} \mid n >= 0, m >= 0 \}$

$(aa)^* (bb)^*$.

29. $L = \{ a^n b^m \mid n >= 4, m <= 3 \}$

$aaaaa^* (\varepsilon + b + bb + bbb)$

30. RE for the string of a's & b's such a's are divisible by 3

R.E = $(b^* a b^* a b^* a)^*$

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

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

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

~~(a+b)(a+b)~~

32. WARE such that every block of four consecutive symbol contains atleast two a's

$((a+b) a (a+b) a + (a+b)(a+b) aa +$

$aa (a+b)(a+b) + a(a+b)(a+b) a +$

$(a+b) aa (a+b) + a(a+b) a(a+b))$ (Same thing $)^*$

33. $L = \{a^n b^m \mid m >= 1,\ n >= 1,\ nm >= 3\}$

$3\times1 \qquad 1\times3 \qquad 2\times2$

$(aa\,a)a^* b b^* + a a^* (bbb) b^* + (aa) a^* (bb) b^*$

34. R.E for the strings of a's & b's containing not more than 3 a's

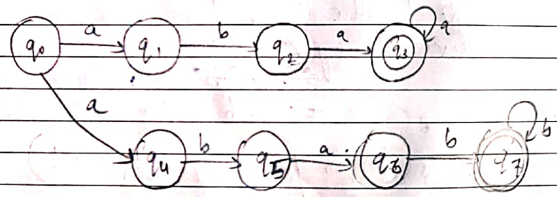
35. $L = \{ vuv \mid u, v \in \{a, b\}^* \text{ and } |v| = 2 \}$

Epsilon NFA.

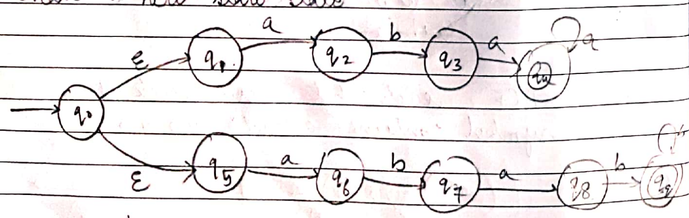
Eg Construct NFA for aba^n or $abab^n$.

aba^n

$abab^n$



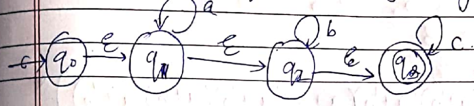
Create a new start state.



↓

Epsilon NFA

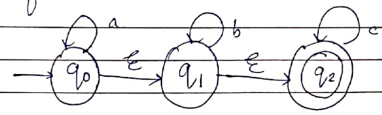
1. Construct an E-NFA for any no of a's followed by any b's followed by any c's.



* Epsilon closure. (See SS).

* Conversion of E-NFA to DFA.

Example -



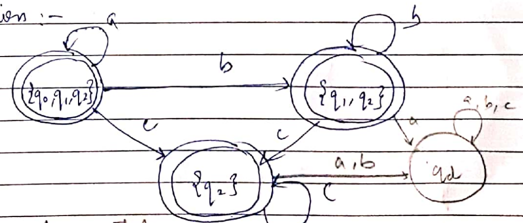
δ_{NFA}	ε	a	b	c
q_0	q_1	q_0	null	null
q_1	q_2	null	q_1	null
* q_2	null	null	null	q_2

δ_{DFA}	a	b	c
* $\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	new state $\{q_1, q_2\}$	new state $\{q_2\}$
new state * $\{q_1, q_2\}$	null	$\{q_1, q_2\}$	$\{q_2\}$
* $\{q_2\}$	null	null	$\{q_2\}$

EClose $(q_0) = \{q_0, q_1, q_2\}$
EClose $(q_1) = \{q_1, q_2\}$
EClose $(q_2) = \{q_2\}$

Start state of DFA is EClose of start state of NFA

DFA construction :-



nulls are trap-state