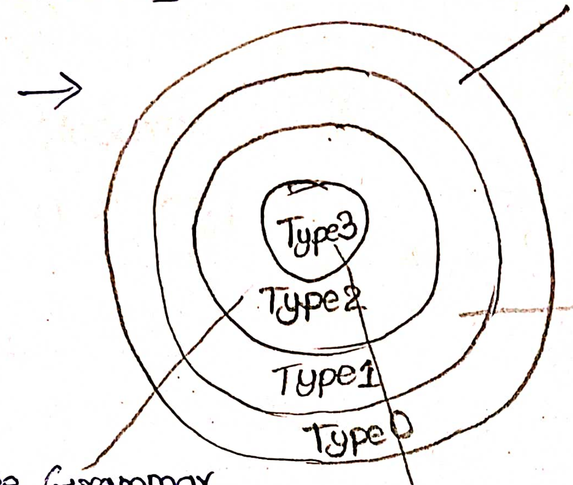
Chomsky Hierarchy of Formal Languages →



Unrestricted Grammar. Recursively enumarable language. Turing machine

(Context Sensitive) CSG, CSL, LBA (Linear bound Automata) (Least Popular)

Context Free Grammar, Context Free Language, Push down Automata (Common) Most Popular

Regular Grammar
Regular Language
Finite automata
(Most restricted)

Formal Languages

Symbols

Some characters which has no meaning by itself.

Alphabet (Σ)

A finite non empty set of Symbols.

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

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

$\Sigma = \{P,Q\}$

String (W or S)

Sequence of symbols taken from a defined alphabet

* $|w|$ Cardinality ⇒ Length of a String
* ε ⇒ Null String

Language (L) ⇒ Set of words

Operations on String

1) Concatenation

$S_1 = Stu \quad S_2 = dent$

$S_1 \cdot S_2 = Student$

2) Kleen Closure / Kleen Star

⇒ Applied on set of characters

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

$\Sigma^* = \cup \Sigma^i$ where $i \geq 0$ $\quad i \rightarrow$ Length of Strings.

$\Sigma^0 = \{\varepsilon\}$

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

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

$\Sigma^3 = \{000, \ldots. 111\}$

→ Always infinite set

⇒ Applied on set of strings

$S = \{c, dd\}$

$S^* = \cup S^i, i \geq 0$

$S^0 = \varepsilon$

$S^1 = \{c\}$

$S^2 = \{cc, dd\}$

$S^3 = \{ccc, cdd, ddc\}$

$\vdots$

3) Positive closure

$\Sigma^* = \cup \Sigma^i, \ i \geq 1$

Language - L

L(M)

$L = \{ \omega \mid \text{some rules on } \omega \}$

$L_1 = \emptyset$ Empty Language

$L_2 = \{\varepsilon\}$ Language with empty String

Write a language with $\omega = \{a, b\}$ such that all a precedes b.

$L_3 = \{\omega \in \{a, b\}^*\}$

$L_4 = \{\omega \in \{a, b\}^* \mid \text{all a's precedes b's}\}$

$= \{a, b, ab, aab, aaab, aabb, aaabbb \ldots\}$

$L_5 = \{\underbrace{\omega = ya \mid y \in \{a, b\}^*}\}$

Set of all Strings ends with a.

Σ = Natural numbers

$L_6 = \{x \neq y, \text{square}(x) = y\} = \{2 \neq 4, 6 \neq 36, \ldots\}$

Operations on L

1) $L_1 \cup L_2 = \{\omega, \omega \in L_1 \text{ or } \omega \in L_2\}$

2) $L_1 \cap L_2 = \{\omega, \omega \in L_1 \text{ and } \omega \in L_2\}$

3) Compliment

$\bar{L_1} / L_1' = \Sigma^x - L_1$

4) $L_1^R = \{\omega^R \mid \omega \in L_1\}$

5) Klean closure

6) Positive Closure

$\{a, aa, ab\}$ $L = \{ab, aab, abb, aa, aaa, aba\}$

$L = \{b, a\}$

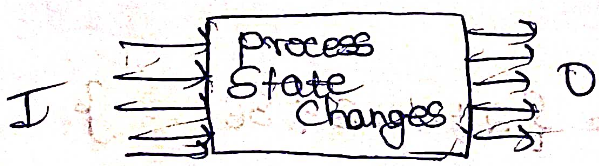
Type 3 (Regular Language)

3 representations

- Finite Automata
- Regular Expression
- Regular Language/Grammar

Finite Automata (FA)

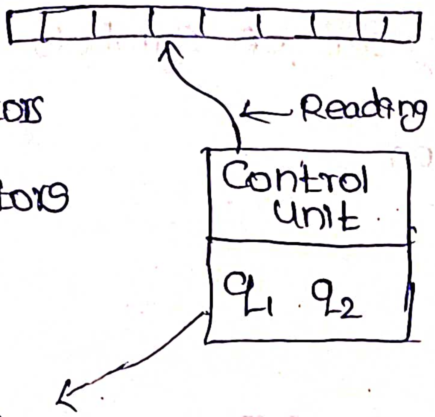
Abstract Model



* Self Propelled machine
* Finite no. of States

Components

* Input tape
* Pattern Recognizors
* Language acceptors



* Consists of finite no. of States
* Set of rules for State transition.

Next State = f (current State, Current i/p Symbol)

Special State - final State

DFA — Deterministic Finite Automata

NFA — Non-deterministic Finite Automata

DFA

* No choices
* What to do - well defined
* For any i/p symbol and current State

There is a unique next State.

Formal Def:

5 tuple DFA $= \{Q, \Sigma, \delta, q_0, F)$

Q = Non-empty finite set of States

Σ = Non-empty finite set of Symbols / Alphabets

δ = Transition function

$$\delta(q, a) = q'$$

$$\delta : Q \times \Sigma \to Q$$

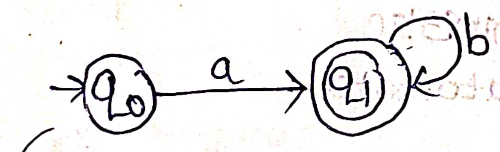
q_0 = Initial State $\quad q_0 \in Q$

F = Final State $\quad F \subseteq Q$

Notations

* Transition Diagram
* Transition Table

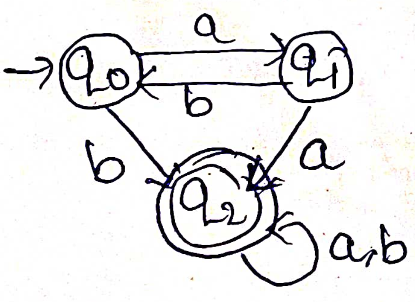
Transition Diagram



initial State

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

$$\delta(q_1, b) = q_1$$



Transition table

Rows - States

Columns - i/p symbols

Cell value - δ

Initial State - $\rightarrow$

Final State - $*$

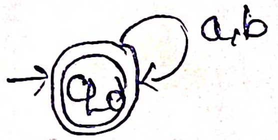
Language of finite Automata

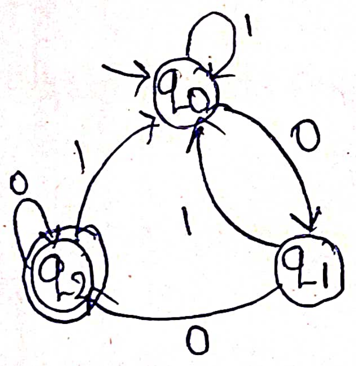
$$L = \{ \omega \mid \omega \in \Sigma^*, \delta(q_0, \omega) \in F \}$$

	a	b
$\rightarrow q_0$	q_1	q_1
q_1	q_2	q_0
$* q_2$	q_2	q_2

Q) Design a DFA to accept set of all strings over a,b

$$L = \{ \omega \mid \omega \in \Sigma^*, \Sigma = \{a,b\} \}$$





	1	0
$\rightarrow q_0$	q_0	q_1
q_1	q_0	q_2
$* q_2$	q_0	q_2

Testing of String

eg: 1100

$\delta(q_0, 1100) = \delta(\delta(q_0, 1), 100)$

$= \delta(q_0, 100) = \delta(\delta(q_0, 1), 00) = \delta(q_0, 00)$

$= \delta(\delta(q_0, 0), 0) = \delta(q_1, 0) = q_2 \in F$

accepted string.

Trace for 1010

$\delta(q_0, 1010) = \delta(\delta(q_0, 1), 010) = \delta(q_0, 010)$

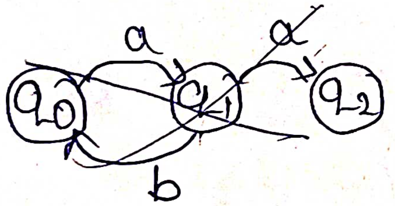
$= \delta(\delta(q_0, 0), 10) = \delta(q_1, 10) = \delta(\delta(q_1, 1), 0)$

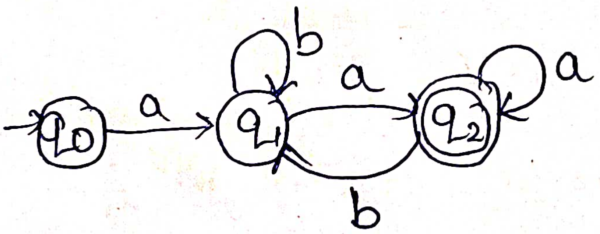
$= \delta(q_0, 0) = q_1 \notin F$

String not acceptable.

Q) Design a DFA to accept set of all strings over a and b

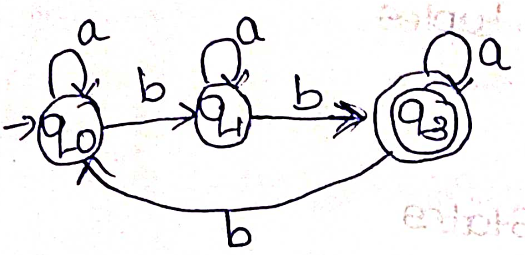
$L = \{awa, w \in \Sigma^*\}$





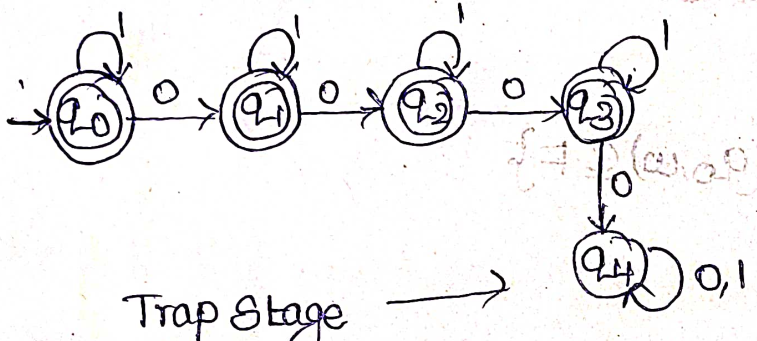
Design a DFA to accept the language $L = \{ \#b(\omega) \bmod 3 > 1, \omega \in (a,b)^* \}$

$\#b$ - no.of b's

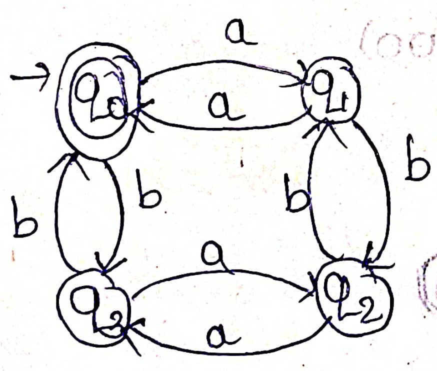


Design DFA to accept set of strings over 0 and 1 with atmost 3 0's

No.of 0's - 0,1,2,3



Design a DFA to accept set of all strings over a and b with even no.of a's and b's



Non Deterministic FA

Formal Def:

An NFA can be formally defined as 5 tuples

$NFA = \{Q, \Sigma, \delta, q_0, F\}$

Q = Non-empty finite set of states

Σ = Non-empty finite set of symbols

q_0 = Initial State

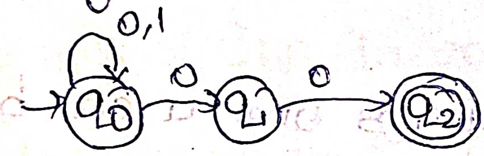
F = Set of final States

$\delta : Q \times \Sigma \rightarrow 2^Q$ (Set of all subsets of Q)

Language of DFA

$L = \{ \omega \mid \omega \in \Sigma^*, \delta(q_0, \omega) \subseteq F \}$

Ending with 00



1100

$\delta(q_0, 1100) = \delta(\delta(q_0,1), 100) = \delta(q_0, 100)$

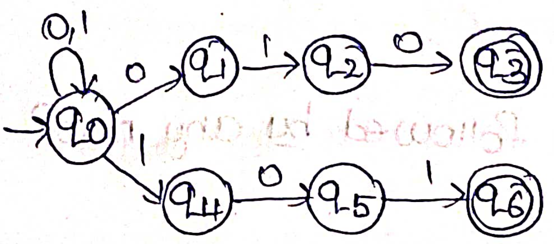
$= \delta(\delta(q_0,1), 00) = \delta(q_0, 00)$

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

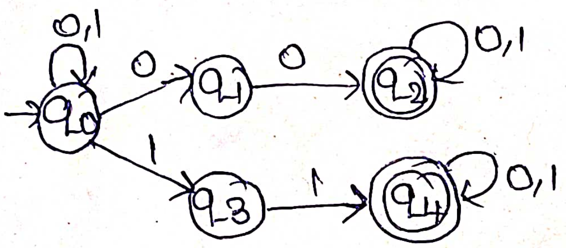
$= \delta(q_0, 0) \cup \delta(q_1, 0)$

$= \{q_0, q_1\} \cup \{q_2\} = \{q_0, q_1, q_2\}$ $\quad \underline{\underline{q_2 \in F}}$

(3) Design an NFA set of all strings on 0 and 1 that should end with either 0,1,0 or 1,0,1



(4) Design an NFA to accept set of all strings are 0 and 1, with 2 consecutive 0's or 2 consecutive 1's



Trace 1001

$\delta(q_0, 1001)$

$= \delta(\delta(q_0,1), 001) = \delta(\{q_0,q_3\}, 001)$

$= \delta(\delta(\{q_0,q_3\},0), 01)$

$= \delta(q_0, 001) \cup \delta(q_3, 001)$

$= \delta(\delta(q_0,0), 01) \cup \delta(\delta(q_3,0), 01)$

$= \delta(\{q_0,q_1\}, 01) \cup \delta\phi$

$= \delta(q_0, 01) \cup \delta(q_1, 01)$

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

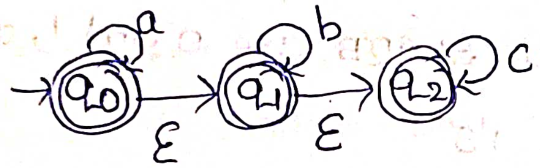
$= \delta(\{q_0,q_1\}, 1) \cup \delta(q_2, 1)$

$= \delta(q_0,1) \cup \delta(q_1,1) \cup \delta(q_2,1) = \{q_0, q_8\} \cup q_2$

$= \{q_0, q_8, q_2\} \wedge F = q_2$

Design NFA to accept

$\Sigma = \{a,b,c\}$ with any no. of a's followed by any no. of b's followed by any no. of c's



NFA with ε transition

Formal Def:

A NFA with ε can be formally defined as a 5 tuple.

$NFA = \{Q, \Sigma, \delta, q_0, F\}$

Q = Non-empty finite set of states

Σ = Non-empty finite set of symbols

q_0 = Initial state

F = Final states.

$\delta : Q \times \{\Sigma \cup \varepsilon\} \rightarrow 2^Q$

	a	b	c	ε
q_0	q_0	ϕ	ϕ	q_1
q_1	ϕ	q_1	ϕ	q_2
q_2	ϕ	ϕ	q_2	ϕ

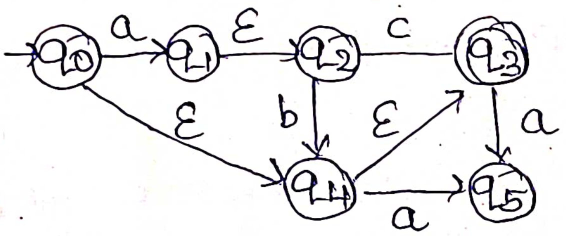
ϵ-closure of state Q is set of all states that can be reached from Q without reading any input symbols.

$$\epsilon\text{-closure}(q_0) = \{q_0, q_1, q_2\}$$

ϵ-closure will never be an empty set. The state itself is in it.

ϵ-closure $(q_1) = \{q_1, q_2\}$

ϵ-closure $(q_2) = \{q_2\}$



ϵ-closure $(q_0) = \{q_0, q_4, q_3\}$

ϵ-closure $(q_4) = \{q_4, q_3\}$

ϵ-closure $(q_1) = \{q_1, q_2, q_3\}$

ϵ-closure $(q_5) = \{q_5\}$

ϵ-closure $(q_2) = \{q_2, q_3\}$

" $(q_3) = \{q_3\}$

Tracing

$$\delta(q_0, ab) = \epsilon\text{-closure}\Big(\bigcup_{P \in \delta(q_0, a)} \delta(P, b)\Big)$$

ϵ-closure $(q_0) = \delta(q_0, t) = \{q_0, q_1, q_2\}$

$\delta(q_0, a) \cup \delta(q_1, a) \cup \delta(q_2, a)$

$\{q_1\} \cup \phi \cup \phi$

ϵ-closure $(q_0) = \{q_0, q_1, q_2\}$

$\delta(q_0,b) \cup \delta(q_1,b) \cup \delta(q_2,b)$

$\phi \cup q_1 \cup \phi$

$\varepsilon\text{-closure}(q_1) = \{q_1,q_2\} \cap F \neq \phi$

Accepted

$\delta(q_0,ab)$

$\varepsilon\text{-closure}(q_0) = \{q_0,q_4,q_3\}$

$\delta(q_0,a) \cup \delta(q_4,a) \cup \delta(q_3,a)$

$q_1 \cup q_5 \cup q_5$

$q_1 \cup q_5$

$\varepsilon\text{-closure}\{q_1 \cup q_5) = \{q_1,q_2,q_3\} \cup \{q_5\}$

$= \{q_1,q_2,q_3,q_5\}$

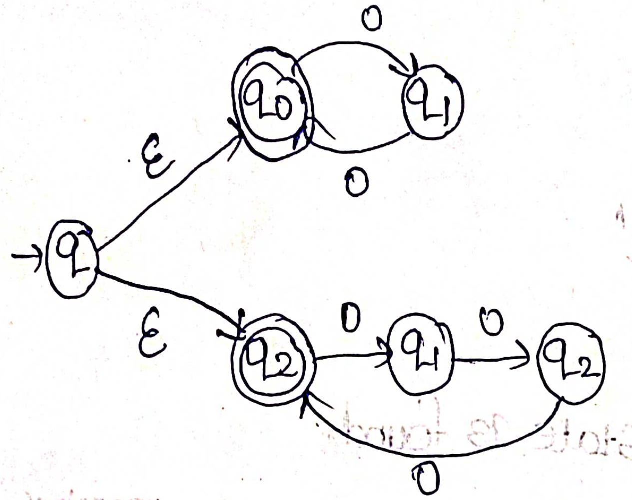
$\delta(q_1,b) \cup \delta(q_2,b) \cup \delta(q_3,b) \cup \delta(q_3,b) \cup \delta(q_5,b)$

$\phi \cup q_4 \cup \phi \cup \phi$

q_4

$\varepsilon\text{-closure}(q_4) = \{q_4,q_3\} \cap F \neq \phi$

Q) Design a NFA for the $L = \{ 0^k, k \text{ is a multiple of } 2 \text{ or } 3 \}$

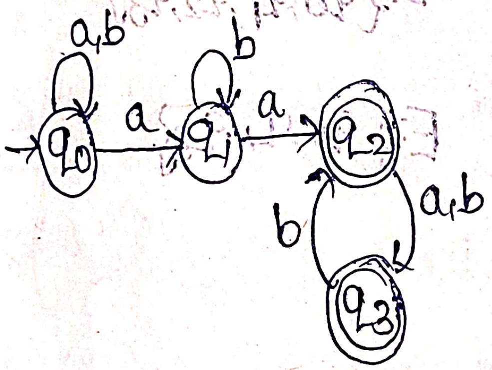


Conversion of NFA to DFA

Subset construction

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

$DFA = (Q', \Sigma, \delta', q_0', F')$



Step1:

Assign q_0' as q_0

Initialize $Q' = \{q_0\}$

Step2:

Find $\delta(q_0, a)$, $\forall a \in \Sigma \rightarrow B$

~~Q' U B~~

$Q' = Q' \cup B$

Step3

Find $\delta(q, a)$ $\forall$ new state in Q'

Update Q'.

4) Repeat the steps until no new state is found.

~~Step5~~ Step5: Set of states which contains atleast one member of final states of given NFA.

$\delta(q_0, a) = \{q_0, q_1\}$ New state

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

$\delta(\{q_0, q_1\}, a) = \{q_0, q_1, q_2\}$ new state

~~$\delta(\{q_0, q_1\}, b) = \{q_0, q_1\}$~~

$\delta(\{q_0, q_1, q_2\}, a) = \{q_0, q_1, q_2, q_3\}$ new

$\delta(\{q_0, q_1, q_2\}, b) = \{q_0, q_1, q_3\}$ new

$\delta(\{q_0, q_1, q_2, q_3\}, a) = \{q_0, q_1, q_2, q_3\}$

$\delta(\{q_0, q_1, q_2, q_3\}, b) = \{q_0, q_1, q_3, q_2\}$

$\delta(\{q_0, q_1, q_3\}, a) = \{q_0, q_1, q_2\}$

$\delta(\{q_0, q_1, q_3\}, b) = \{q_0, q_1, q_2\}$

Q'

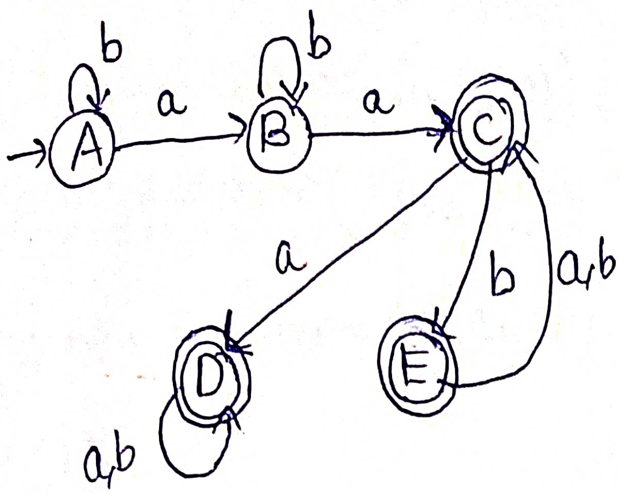
A: $\{q_0\}$

B: $\{q_0, q_1\}$

C: $\{q_0, q_1, q_2\}$

D: $\{q_0, q_1, q_2, q_3\}$

E: $\{q_0, q_1, q_3\}$



*) convert NFA to DFA

	a	b
$\rightarrow q_0$	$\{q_0,q_1\}$	$\{q_2\}$
q_1	$\{q_0\}$	$\{q_1\}$
$* q_2$	ϕ	$\{q_0,q_2\}$

$q_0' = q_0$

$\delta(q_0,a) = \{q_0,q_1\}$ new

$\delta(q_0,b) = \{q_2\}$ new

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

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

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

$\delta(q_2,b) = \{q_0,q_2\}$ new

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

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

Q'

$\{q_0\}$ A

$\{q_0,q_1\}$ B

$\{q_2\}$ C

$\{q_1,q_2\}$ E

$\{q_0,q_2\}$ D

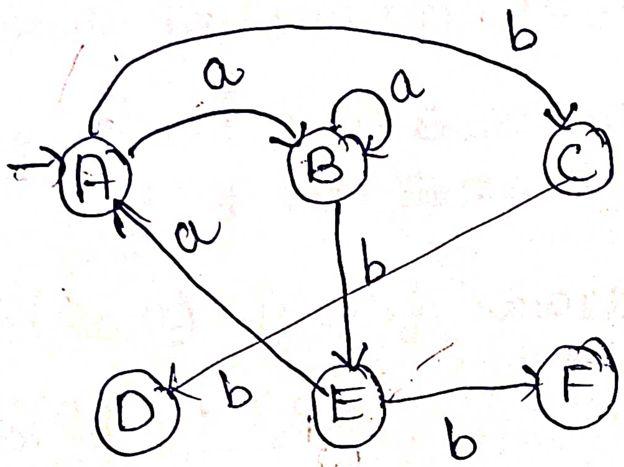
$\{q_0,q_1,q_2\}$ F

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

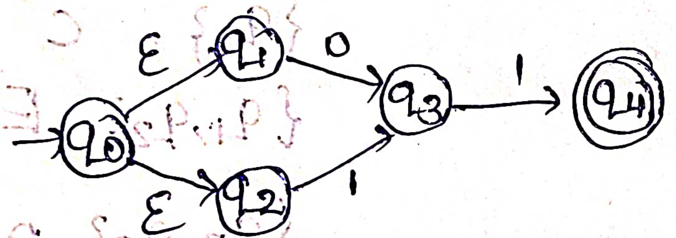
$\delta(\{q_0,q_2\},b) = \{q_2,q_0\}$

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

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



NFA ε to DFA



Q'

$\{q_0,q_1,q_2\}$ – A

$q_0 = \varepsilon\text{-closure}(q_0) = \{q_0,q_1,q_2\}$

$\delta(A,0) = \varepsilon\text{-closure}(\delta(q_0,0) \cup \delta(q_1,0) \cup \delta(q_2,0)) = q_3$ new

$\delta(A,1) = \varepsilon\text{-closure}(\delta(q_0,1) \cup \delta(q_1,1) \cup \delta(q_2,1))$

$= q_3$

Q1

$\{q_0, q_1, q_2\}$ - A

$\{q_3\}$ - B

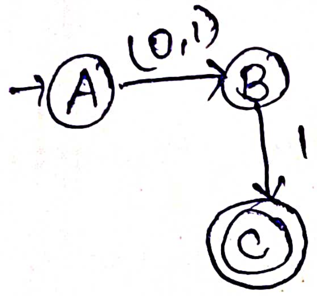
$\{q_4\}$ - C

$\delta(B,0) = \varepsilon\text{-closure}(\delta(q_3,0)) = \phi$

$\delta(B,1) = \varepsilon\text{-closure}(\delta(q_3,1)) = q_4$ new

$\delta(C,0) = \varepsilon\text{-closure}(\delta(q_4,0)) = \phi$

$\delta(C,1) = \phi$



Regular Grammar

Formal Grammar can be defined as 4 tuples

$G = (V, T, P, S)$

V = ~~Finite~~ non-empty finite set of non-terminals (NT)

T = Non-empty finite set of Terminals (T)

P = Non-empty finite set of Production rules

LHS → RHS

S = Start symbol $S \in V$

Regular Grammar can be defined as four tuples

$G = (V, T, P, S)$

V = Non-terminals

T = Terminals

P = LHS → Single NT

RHS → Nothing (ε) or Single T or Single T followed by Single NT or Sing NT Single T

$A \to \varepsilon$ | $A \to \varepsilon | a | aB$ (Right linear Reg. grammar)

$A \to a$

$A \to aB$ | $A \to \varepsilon | a | Ba$ (Left linear reg. grammar)

$A \to Ba$

S = Start symbol $S \in V$

Derivation

$u \to v$

$u \overset{*}{\Rightarrow} v$

$A \to a | aB$

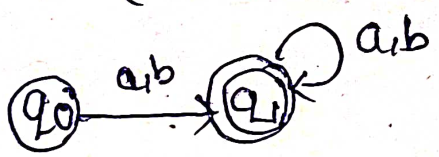
$B \to a$

$A \to a$ | $A \to aB \to aa$

$A \overset{*}{\Rightarrow} aa$

Q) Regular grammar for generating any strings of a and b without ε

$L = (a+b)^+$



abba

$S \to aS$ [$S \to aS$]

$\to abS$ [$S \to bS$]

$\to abbS$ [$S \to bS$]

$\to abba$ [$S \to a$]

$S \to a | aS | b | bS$

ab

$S \to aS$

$\to ab$

$G = (V, T, P, S)$

$V = \{S\}$

$T = \{a, b\}$

$P =$

4) set of strings begin with Y over X and Y. (Regular grammar)

~~$S \to y \mid yS \mid yT$~~

~~$T \to xS \mid$~~

$S \to yA$

$A \to yA \mid xA \mid \epsilon$

5) Length of string should even.

$S \to aA \mid bA \mid \epsilon$

$A \to aS \mid bS$

6) Generative
All words a and b, such that word start and end with a.

P: $S \to aA \mid a$

$A \to aA \mid bA \mid aB \mid bB$

$B \to aB \mid a$

$G = (\{S, A, B\}, \{a, b\}, P, S\}$

Equivalence of ~~NFA and~~ Regular Grammar & FA

Convertion of Regular Grammar to FA

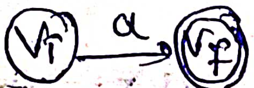
Assume that it is Right Linear Grammar

1) Create State for each NT, State ≈ Start symbol → initial State

2) $V_i \to aV_j$, $a \in T$, $\delta(V_i, a) = V_j$

$V_i \xrightarrow{a} V_j$

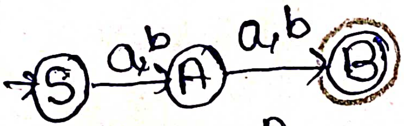
3) $V_i \to a$, $\delta(V_i, a) = V_f$; $V_f \in F$



4) $V_i \to \varepsilon$

✲) $S \to aA \mid bA$

$A \to a \mid b$



$S \to aA$

$S \to bA$

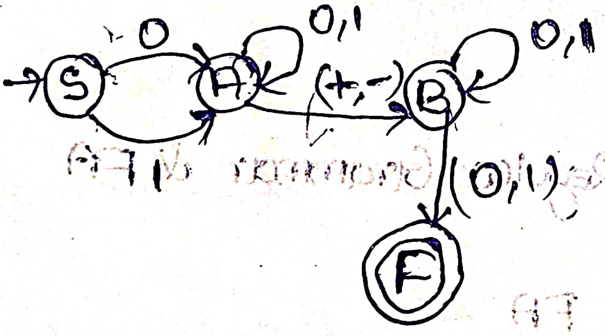
$A \to a$

$A \to b$

✲) $S \to 0A \mid 1A$

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

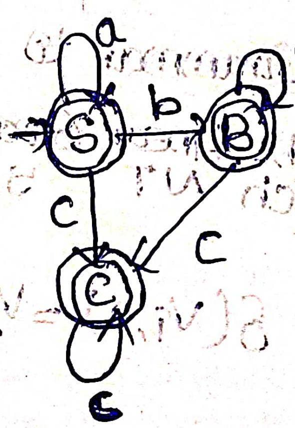
$B \to 0B \mid 1B \mid 0 \mid 1$



✲) $S \to aS \mid bB \mid cC \mid \varepsilon$

$B \to bB \mid cC \mid \varepsilon$

$C \to cC \mid \varepsilon$



In case of Left Linear

→ Follow procedures for RL

→ Reverse →

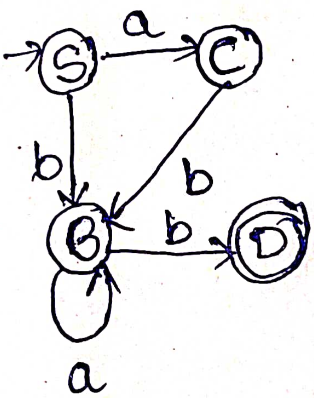
→ swap initial & F

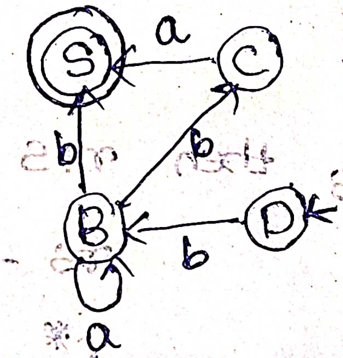
→ convert to DFA

☆) $S \to Ca \mid Bb$

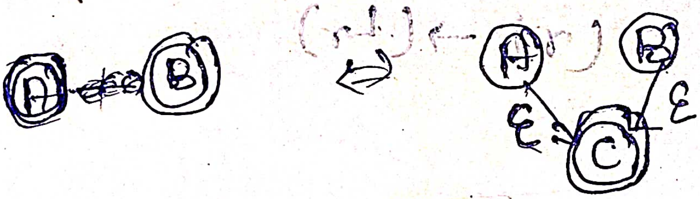
$C \to Bb$

$B \to Ba \mid b$

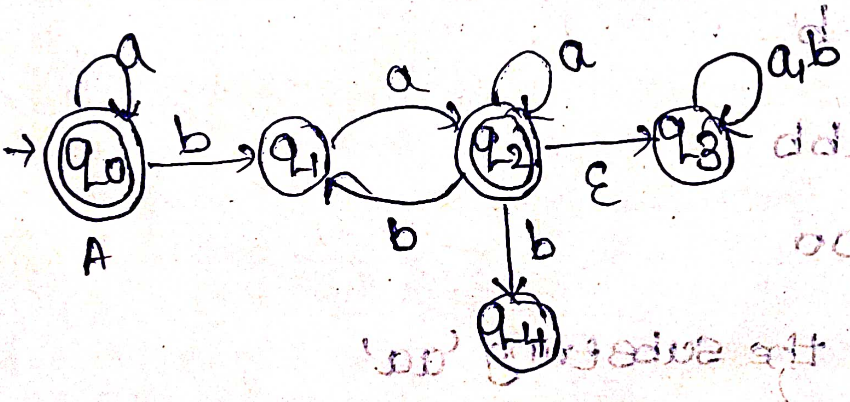




Incase of multiple final states



FA to RG



P: $A \rightarrow aA \mid bB \mid \epsilon$

$B \rightarrow aC$

$C \rightarrow aC \mid bB \mid \epsilon$