TOC

/ /

PS

These are my class notes.
Empty pages indicate I was absent
or highly confused/sleepy.
Read relevant topics from book for the same.

Languages → Regular, context free

Symbol - A symbol is an abstract entity we don't define formally just as 'point' is not defined in geometry. e.g. $\{0, 1, A, B, C\}$

Alphabet - Finite, non-empty, set of symbols.
→ 'Σ' denotes Alphabet
e.g. $\Sigma_1 = \{0, 1\}$

String - A Finite sequence of symbols chosen from the alphabet 'Σ'
e.g. "001100",
e.g. strings divisible by 3 over $\Sigma_1 = \{0, 11, 110, 1001 \ldots\}$

Length of string : $|\omega|$ where ω = "12345"

Concatenation : x = "110" y = "1111" xy = 110 1111 (x | y)

Power of an Alphabet ⇒ Let 'Σ' be an alphabet, a set of certain length 'k' from the alphabet is called power of an alphabet and is denoted by Σ^k

e.g. Set of all strings of length 0 → $\Sigma^0 = \{\in\}$
Set of all strings of length 1 → $\Sigma^1 = \{0, 1\}$
-11- 2 → $\Sigma^2 = \{00, 01, 10, 11\}$

$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \ldots \Sigma^n$
↳ summation *

$\Sigma^+ = \Sigma^* - \Sigma^0$ → positive
↳ everything excluding 0

... Σ^+ mainly used as it excludes 'ε'

Language L is said to be a language over the alphabet 'Σ' iff $L \subseteq \Sigma^*$.

e.g.
Let L be the language of all strings over the set $\{a, b\}$ which start with a or end with b.
↳ $\{a, b, aa, ab, bb, aaba, bab \ldots\}$

Not belonging to language → $\{baa, ba \ldots\}$

e.g.2) start with 'a' and end with 'b'
$L = \{ab, aab, abb, \ldots\}$

$L \neq \{\varepsilon, a, b, ba, baa, aba \ldots\}$

Empty Language:

- No strings in the language.
- denoted by ϕ
- Empty language is NOT EQUAL to $\in$.

$L = \{\in\} \neq \phi$

↑ Language with Empty string — is not equal — Empty Language

Concatenation of Languages

Let $L1$ and $L2$ be languages, Concatenation denoted by

$$L1L2 = \{uv \mid u \in L1, v \in L2\}$$

e.g.

$L1 = \{ab, b\}$

$L2 = \{aaa, abb, aaba\}$

$L1L2 = \{abaaa, ababb, abaaba, baaa, babb, baaba\}$

e.g.2)

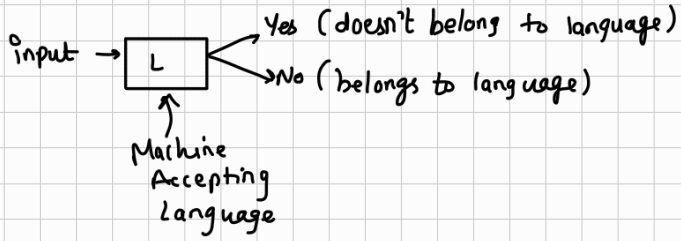
$L1 = a^*b$

$L1 = \{b, ab, a^2b, a^3b \cdots a^nb\}$

$L1^2 = L1L2 = a^*b\ a^*b$

$= \{bb, bab, ba^2b \cdots\}$

* In TOC we're majorly looking for decidability problems.

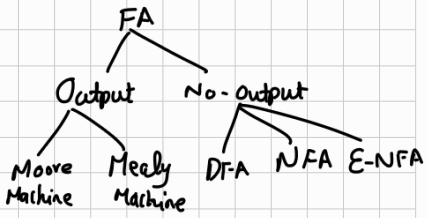


Regular → Finite Automata

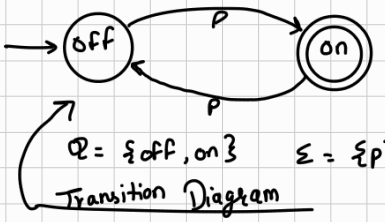
Context free language → Push down Automata

DFA

Deterministic Finite Automata



e.g. an basic on-off switch



$Q = \{off, on\}$ $\Sigma = \{p\}$

Transition Diagram

A deterministic finite automata (DFA) is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where,

- Q is the finite set of states.
- Σ is the alphabet
- δ is the transition function defined as $Q \times \Sigma \rightarrow Q$
- $q_0 \in Q$ and q_0 is the initial state (start state)
- $F \subseteq Q$ and F is a set of Final (accept) states

* In DFA it is not allowed to omit any transition.

It will take a state from 'Q' an alphabet from 'Σ' to transition to another state.

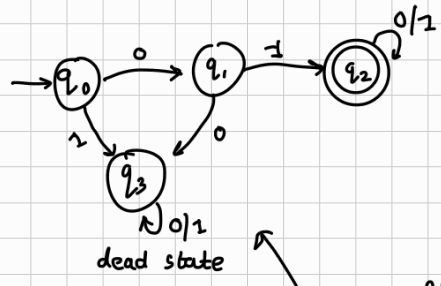
Transition Table

	p
→ off	on
* on	off

→ Start state

* Final state

Q.1) Construct an FA over {0,1} where every string starts with 01.



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

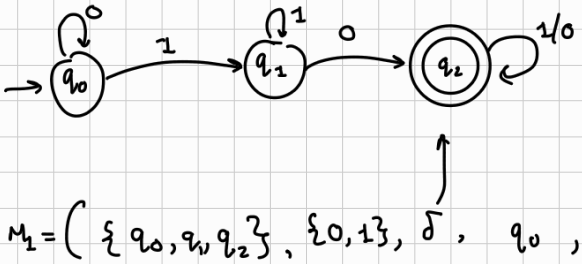
$Q = \{q_0, q_1, q_2, q_3\}$

$L = \{01(0+1)^*\}$

$\therefore$ delta is given by the transition diagram.

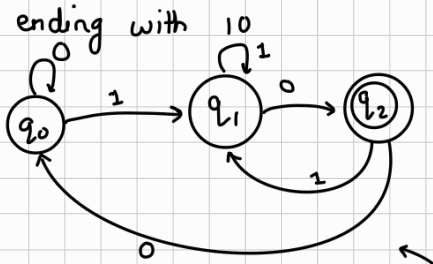
$M_1 = (\{q_0, q_1, q_2, q_3\}, \{0,1\}, \delta, q_0, \{q_2\})$

Q.2) Construct a FA over {0,1} which contain a substring 1,0



$M_1 = (\{q_0, q_1, q_2\}, \{0,1\}, \delta, q_0, \{q_2\})$

Q. -11- ending with 10



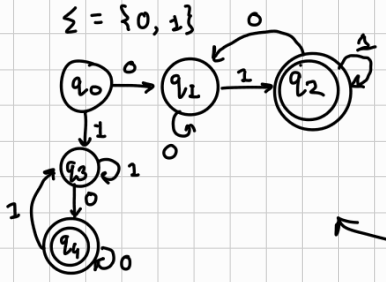
Transition Table

	0	1
q_0	q_0	q_1
q_1	q_2	q_1
q_2	q_0	q_1

$L = \{(0+1)^* 10\}$

$M_1 = (\{q_0, q_1, q_2\}, \{0,1\}, \delta, q_0, \{q_2\})$

Q. Start and end with a different symbol. {0,1}



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

$L = \{0(0+1)^*1, \ 1(0+1)^*0\}$

0011

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

$M = (\{q_0, q_1, q_2, q_3\}, \{0,1\}, \delta, q_0, \{q_2, q_4\})$

EXTENDED Transition function:

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

$\hat{\delta} : Q \times \Sigma^* \to Q$

↳ (Delta hat) $\hat{\delta}(q, w)$ is a state in the FA after reading the string w from the start state q.

$P = \hat{\delta}(q, w)$

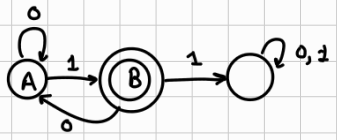
Formal def$_n$:

Basic: $\hat{\delta}(q, \varepsilon) = q$

Induction: $\hat{\delta}(q, wa) = \delta(\underbrace{\hat{\delta}(q, w)}_{P}, a)$

- A string w is accepted by a DFA $M = (Q, \Sigma, \delta, q_0, F)$ if $\hat{\delta}\{q_0, w\} = p$, where $p \in F$.
- Language accepted by Machine M, can be written as all string w such that $\hat{\delta}(q_0, w) \in F$

$L(M) = \{w \mid \hat{\delta}(q_0, w) \in F\}$



← Any string with NO consequtive 1's

$\hat{\delta}(A, 101) = \delta(\hat{\delta}(a, 10), 1)$

$= \delta(\delta(\hat{\delta}(a, 1), 0), 1)$

$= \delta(\delta(\delta(\hat{\delta}(a, \epsilon), 1), 0), 1)$

$= \delta(\delta(\delta(A, 1), 0), 1)$

$= \delta(\delta(B, 0), 1)$

$= \delta(A, 1)$

$= B$

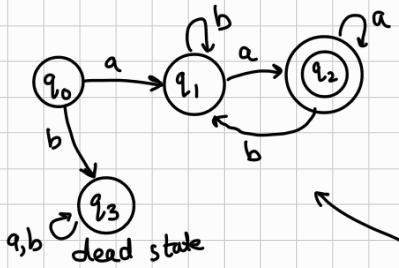
} Recursive call

REGULAR LANGUAGES

A Language L is regular if it is accepted by some DFA.

Q. Show that $L = \{ awa \mid w \in \{a,b\}^* \}$

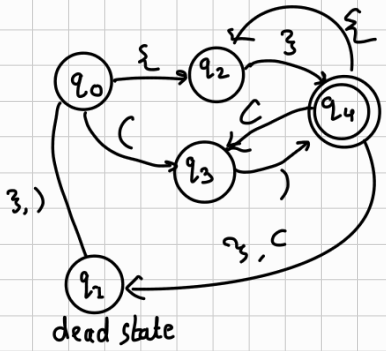
A language is regular if it is accepted by a DFA,



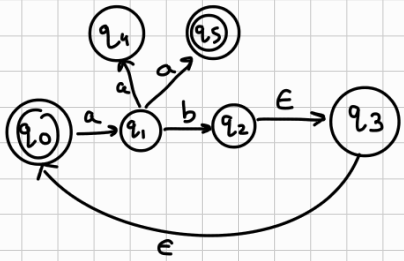
$M = (\{q_0, q_1, q_2, q_3\}, \{a,b\}, \delta, q_0, \{q_2\})$

∴ The Language $\{a,b\}$ is regular.

Q. $L = \{ w \mid w \in \{\{,(,\},)\}^* \text{ and } w \text{ is balanced} \}$



Consider following ε-NFA



* Always consider '$\hat{\delta}$' (Search the related stuff up.)

$$\hat{\delta}(q_0, \epsilon) = \epsilon^* q_0$$
$$= q_0$$

$$\hat{\delta}(q_0, a) = \epsilon\text{-closure}(q_1)$$

$$\hat{\delta}(q_0, ab) = \epsilon\text{-closure}(q_2)$$

$$\epsilon\text{-closure}(q_2) = \{q_2, q_3, q_0\} \subseteq F$$

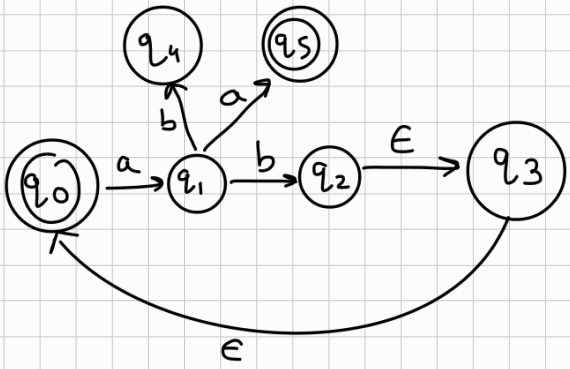
$\hat{\delta}(q_0, abaa)$

$\hat{\delta}(q_0, \epsilon) = \epsilon\text{-closure}(q_0)$
$= q_0$

$\hat{\delta}(q_0, a) = \epsilon\text{-closure}(q_1)$
$= q_1$

$\hat{\delta}(q_0, ab) = \epsilon\text{-closure}(q_2)$
$= \{q_2, q_3, q_0\}$

$\hat{\delta}(q_0, aba) = \epsilon\text{-closure}(q_1)$
$= q_1$



$\hat{\delta}(q_0, \epsilon) = q_0$

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

$\hat{\delta}(q_0, ab) = \epsilon\text{-closure}[\hat{\delta}(q_0, a), b)$
$= \epsilon\text{-clo}(\hat{\delta}(q_1, b))$
$= \epsilon\text{-clo}(q_2, q_4)$
$= \{q_4, q_2, q_3, q_0\}$

$\hat{\delta}(q_0, aba) = \epsilon\text{-clo}(\hat{\delta}(q_0, a), b), a)$
$= \epsilon\text{-clo}(\hat{\delta}(q_1, b), a)$
$= \epsilon\text{-clo}(\hat{\delta}(q_4, q_2, q_3, q_0), a)$
$= = \{q_1\}$

$\hat{\delta}(q_2, a) = \epsilon\text{-clo}(\delta(\hat{\delta}(q_2, \epsilon), a)$
$= \epsilon\text{-clos}(\delta((q_2, q_3, q_0), a))$
$= \epsilon\text{-clos}(q_1)$

$= q_1$

$\hat{\delta}(q_3, a) = \epsilon\text{-clo}(\delta(\hat{\delta}(q_3, \epsilon), a)$

$= \epsilon\text{-clo}(\delta(q_3, q_0), a)$

$= \epsilon\text{-clo}(q_1)$

$= \{q_1\}$

$\hat{\delta}(q_4, a) = \epsilon\text{-clo}(\delta(\hat{\delta}(q_4, \epsilon), a)$

$= \epsilon\text{-clo}(\delta(q_4, a))$

$= \{\epsilon\}$

abaa is accepted for 1 thread.

$\hat{\delta}(q_0, abab)$

$\hat{\delta}(q_0, \epsilon) = \epsilon\text{-clo}\{q_0\} = \{q_0\}$

$\hat{\delta}(q_0, a) = \epsilon\text{-clo}\{\delta(q_0, a)\} = \epsilon\text{-clo}\{q_1\} = \{q_2\}$

$\hat{\delta}(q_0, ab) = \epsilon\text{-clo}(\delta(\hat{\delta}(q_0, a), b) = \epsilon\text{-clo}\{\}$

$\hat{\delta}(q_1, b) = \epsilon\text{-clo}(\hat{\delta}(q_1, b))$

$= \epsilon\text{-clo}(q_4, q_2)$

$= \{q_4, q_2, q_3, q_0\}$

$q_0 \in F$ i.e. $q_0 \subseteq F$

$\therefore$ abab is accepted.

δ is on single symbol

$\hat{\delta}$ is on string.

Given an ϵ-NFA $= (Q, \Sigma, \delta, q_0, F)$, construct an equivalent DFA,

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

$E_{NFA} = (Q', \Sigma, \delta, q_0', F')$

$Q' = 2^Q$

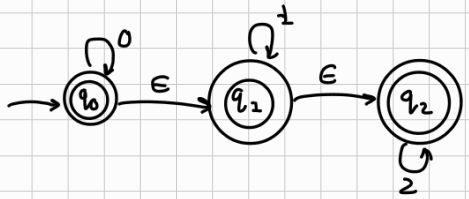
1st major diff $q_0' = \epsilon$-closure (q_0)

F' is set of states of Q' containing a member of F

Let $S \in Q'$ be $\{p_1, p_2 \ldots p_k\}$

a) Compute $\bigcup_{i=1}^{k} \hat{\delta}(p_i, a) = \{r_1, r_2 \ldots r_m\}$

b) $\hat{\delta}(S, a) \bigcup_{j=1}^{m} \epsilon\text{-clo}(r_j)$



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

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

$\epsilon\text{-clo}(q_2) = \{q_2\}$

	0	1	2
$[q_0, q_1, q_2]$	$q_0q_1q_2$	q_1, q_2	q_2
$[q_1 q_2]$	ϕ	q_1q_2	q_2
q_2	ϕ	ϕ	q_2

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

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

$= \{q_0\}$

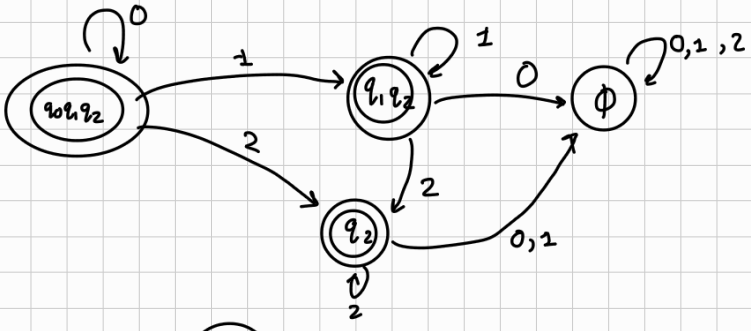
$\epsilon\text{-clo}(q_0) = q_0q_1q_2$

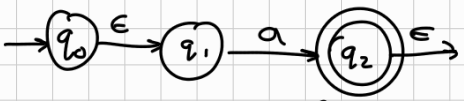
$\hat{\delta}(q_0, 1) = \delta(\hat{\delta}(q_0, \epsilon), 1)$

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

$= q_1$

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





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

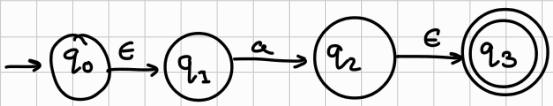
$= \epsilon\text{-clo}(\hat{\delta}(q_0))$

$= \epsilon\text{-clo}(\delta(\hat{\delta}(q_0, \epsilon), a)$

$= \delta(q_0q_1, a)$

$= e\text{-clo}(\phi q_2$

$= q_2 \subseteq F$



ϵ-clo$(q_0) = \{q_0, q_2\}$

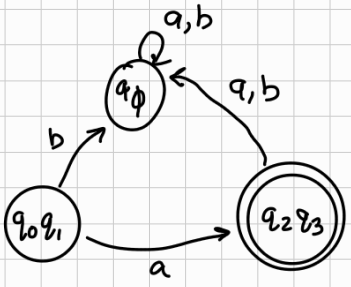
ϵ-clo$(q_1) = \{q_2\}$

ϵ-clo$(q_2) = \{q_2, q_3\}$

ϵ-clo$(q_3) = \{q_3\}$

	a	b
q_0q_1	q_2q_3	ϕ
q_2q_3	ϕ	ϕ

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



$\delta(\{q_0, q_2\}, a) = \epsilon\text{-clo}[\hat{\delta}(q_0, q_1], a)$

$= \delta(\hat{\delta}(q_0, \epsilon), a)$

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

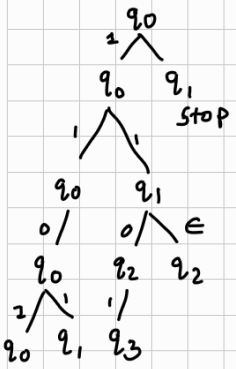
$= \phi \cup q_2$

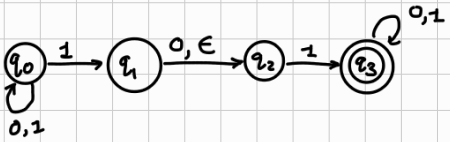
$= q_2$

ϵ-clo$(q_2) = \{q_2, q_3\}$

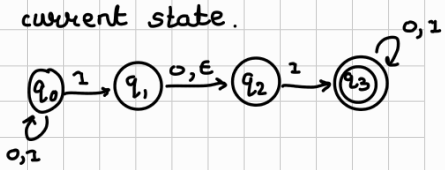
Q. Design a NFA to accept all strings containing 101 or 11 as substring.

Thread Tree

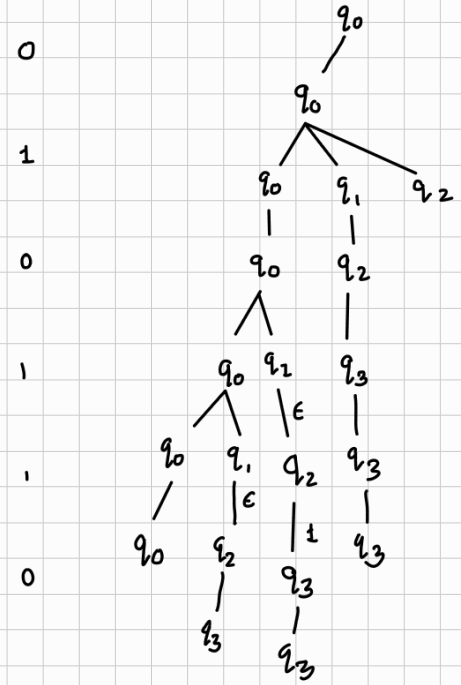




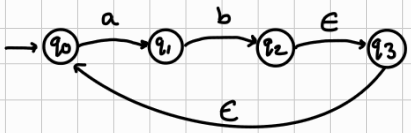
* When a Transition on 'ϵ' is encountered without reading any input, a finite control overtakes multiple inputs. One following each of the epsilon transitions and one state the current state.



Input string - 010110

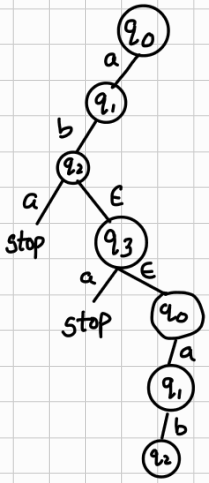


*



here, $(ab)^n \mid n>0$ i.e. $n \geq 1$

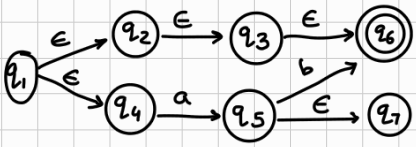
input string: abab



q_2 is the final state.

* ϵ-Closure

- It is a set of all states reachable by following transition function from the given state which are labelled by ϵ.



ϵ-closure of $q_1 = \{q_1, q_2, q_3, q_4, q_6\}$

-11- $q_2 = \{q_2, q_3, q_6\}$

-11- $q_3 = \{q_3, q_6\}$

-11- $q_4 = \{q_4\}$

$q_5 = \{q_5, q_7\}$

$q_6 = \{q_6\}$

$q_7 = \{q_7\}$

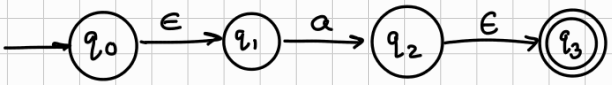
* $\hat{\delta}$ of ϵ-NFA

basis: $\hat{\delta}(q_1, \epsilon) = \epsilon\text{-closure}(q_1)$

induction: let $\omega = xa$ $\hat{\delta}(q, x) = \{p_1, p_2 \ldots p_n\}$

let $\bigcup_{i=1}^{k}$ $\delta(p_i, a) = \{y_1, y_2 \ldots y_m\}$

Then $\hat{\delta}(q, \omega) = \bigcup_{j=1}^{m} \epsilon\text{-closure}(y_j)$



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

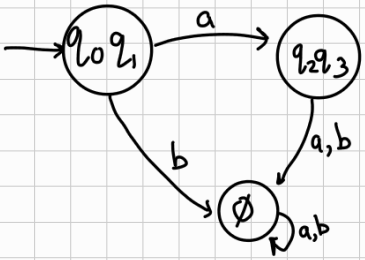
ϵ-closure $(q_0) = \{q_0, q_1\}$

ϵ-closure $(q_1) = \{q_1\}$

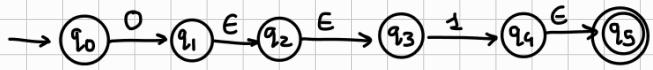
ϵ-closure $(q_2) = \{q_2, q_3\}$

ϵ-closure $(q_3) = \{q_3\}$

	a	b
$[q_0]$	$[q_2q_3]$	ϕ
$[q_1]$	$[q_2q_3]$	ϕ
$[q_2]$	ϕ	ϕ
$[q_3]$	ϕ	ϕ
→ $[q_0,q_1]$	$[q_2q_3]$	ϕ
* $[q_2q_3]$	ϕ	ϕ



Q.



	0	1
→ $[q_0]$	$[q_1q_2q_3]$	ϕ
$[q_1]$	ϕ	$[q_4q_5]$
$[q_2]$	ϕ	$[q_4q_5]$
$[q_3]$	ϕ	$[q_4q_5]$
$[q_4]$	ϕ	ϕ
$[q_5]$	ϕ	ϕ
$[q_1q_2q_3]$	ϕ	$[q_4q_5]$
* $[q_4q_5]$	ϕ	ϕ

ϵ-closure

$q_0 \rightarrow q_0$

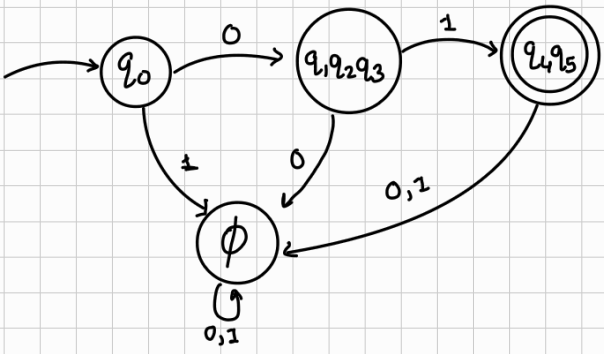
$q_1 \rightarrow q_1 q_2 q_3$

$q_2 \rightarrow q_2 q_3$

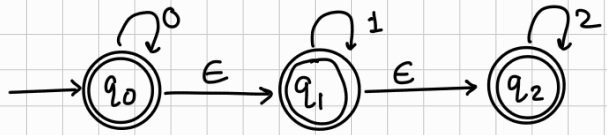
$q_3 \rightarrow q_3$

$q_4 \rightarrow q_4 q_5$

$q_5 \rightarrow q_5$



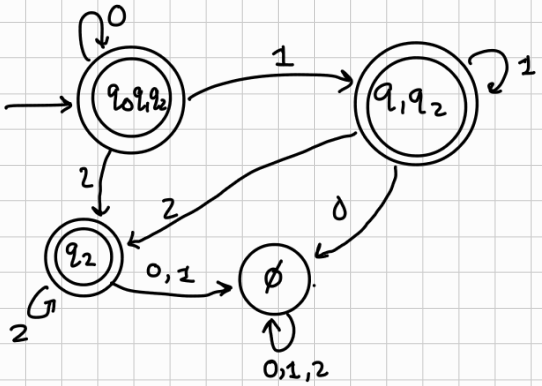
Q.)



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

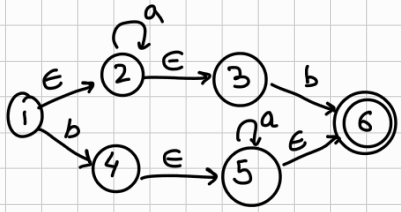
	0	1	2
$[q_0]$	$q_0q_1q_2$	q_1q_2	q_2
$[q_1]$	$\emptyset$	q_1q_2	q_2
$[q_2]$	$\emptyset$	$\emptyset$	q_2
$\rightarrow *q_0q_1q_2$	$q_0q_1q_2$	q_1q_2	q_2
$*q_1q_2$	$\emptyset$	q_1q_2	q_2

state	ϵ-closure
q_0	$q_0q_1q_2$
q_1	q_1q_2
q_2	q_2



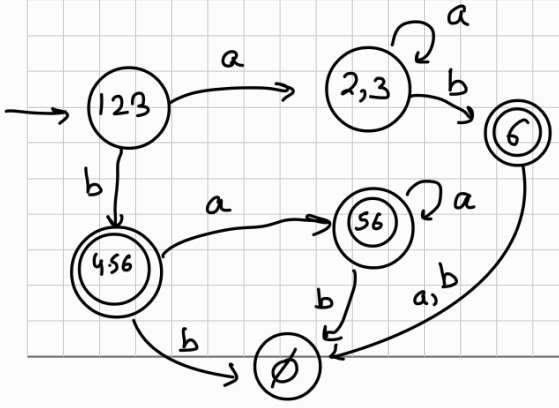
$L = \{\omega \mid \omega = 0^n 1^m 2^k, n, m, k \geq 0\}$

Q)



	a	b
$\rightarrow$ 1	2,3	4,5,6
2	2,3	6
3	$\emptyset$	6
4	5,6	$\emptyset$
5	5,6	$\emptyset$
$*$6	$\emptyset$	$\emptyset$
$\rightarrow$ [1,2,3]	[2,3]	[4,5,6]
$*$ [4,5,6]	[5,6]	$\emptyset$
$*$ [5,6]	[5,6]	$\emptyset$
[2,3]	[2,3]	6

state	ϵ-closure
1	1,2,3
2	2,3
3	3
4	4,5,6
5	5,6
6	6



DFA Minimisation

Equivalent States

- $P \equiv q$ iff $\forall \omega \in \Sigma^* \ \hat{\delta}(p, \omega) \in F$ and $\hat{\delta}(q, \omega) \in F$
- $P \not\equiv q$ iff $\forall \omega \in \Sigma^* \ \hat{\delta}(p, \omega) \notin F$ and $\hat{\delta}(q, \omega) \notin F$

Two states are not equivalent or distinct if

- $P \equiv q$ iff $\forall \omega \in \Sigma^* \ \hat{\delta}(p, \omega) \in F$ and $\hat{\delta}(q, \omega) \notin F$
- $P \not\equiv q$ iff $\forall \omega \in \Sigma^* \ \hat{\delta}(p, \omega) \notin F$ and $\hat{\delta}(q, \omega \in F$

Steps:

① Delete all unreachable states

② Create a lower triangular table:

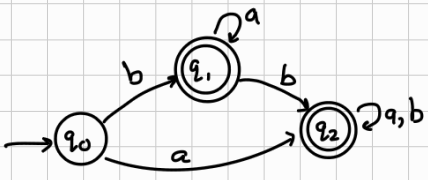
For each pair of P, Q if P is final and Q is not final or vice versa, mark pq

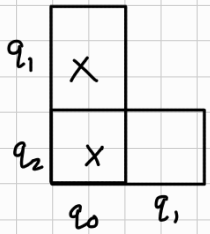
③ Loop until no change in iteration, for each unmarked pair PQ and an input symbol A. Let,

$$r = \delta(p, a)$$
$$s = \delta(q, a)$$

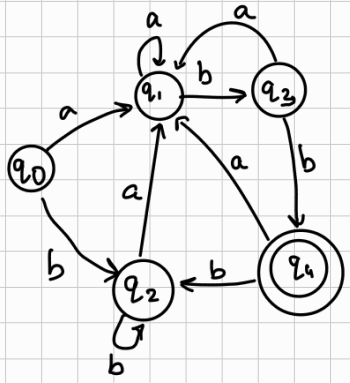
if (r, s) is already marked, mark (p, q).

④





Q)



q_1	X			
q_2		X		
q_3	X	X	X	
q_4	X	X	X	X
	q_0	q_1	q_2	q_3

$X \rightarrow q_4$ final others not final

$(q_0 q_3) \xrightarrow{a} (q_1 q_1)$
$\xrightarrow{b} (q_2 q_4)$ ∴ $q_2 q_4$ is maked mark $q_0 q_3$

$(q_0 q_2) \xrightarrow{a} (q_1 q_1)$
$\xrightarrow{b} (q_2 q_2)$

$(q_0 q_1) \xrightarrow{a} (q_1 q_1)$
$\xrightarrow{b} (q_2 q_3)$

$(q_1 q_3) \xrightarrow{a} (q_1 q_1)$
$\xrightarrow{b} (q_3 q_4)$ ✓

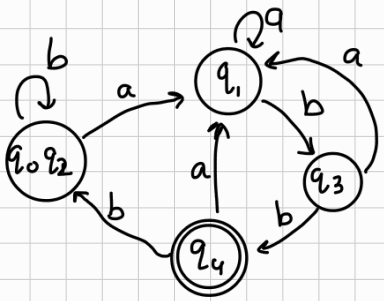
$(q_1 q_2) \xrightarrow{a} (q_1 q_1)$
$\xrightarrow{b} (q_3 q_2)$

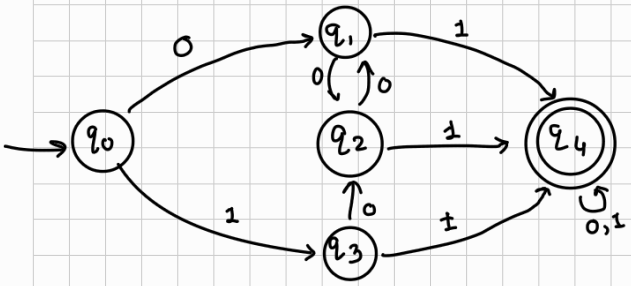
$(q_2 q_3) \xrightarrow{a} (q_1 q_1)$
$\xrightarrow{b} (q_2 q_4)$ ✓

2nd Iteration:

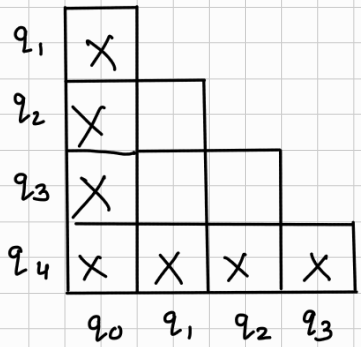
$(q_0 q_1) \xrightarrow{b} (q_2 q_3)$ ✓ Mark

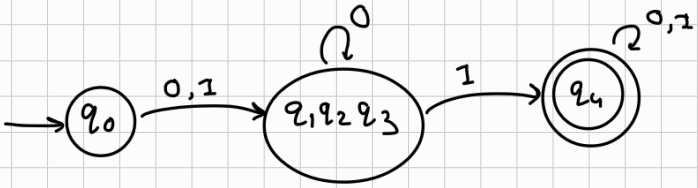
$(q_1 q_2) \xrightarrow{b} (q_3 q_2)$





① All reachable



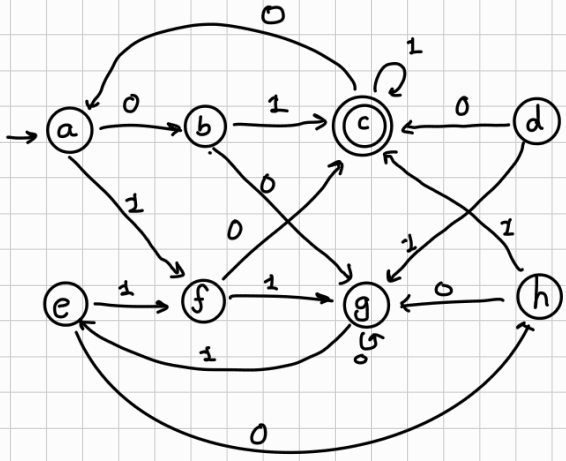


	0	1	
q_0q_3	q_1q_2	q_3q_4	✓
q_0q_2	q_1q_1	q_3q_4	
q_0q_1	q_1q_2	q_3q_4	
q_1q_3	q_2q_2	q_4q_4	
q_2q_1	q_1q_2	q_4q_4	
q_2q_3	q_1q_2	q_4q_4	

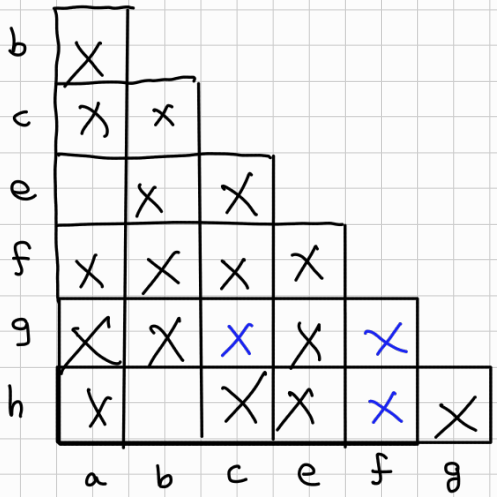
Iteration (II)

q_1q_2	q_1q_2	q_4q_4
q_1q_3	q_2q_2	q_4q_4
q_2q_3	q_1q_2	q_4q_4

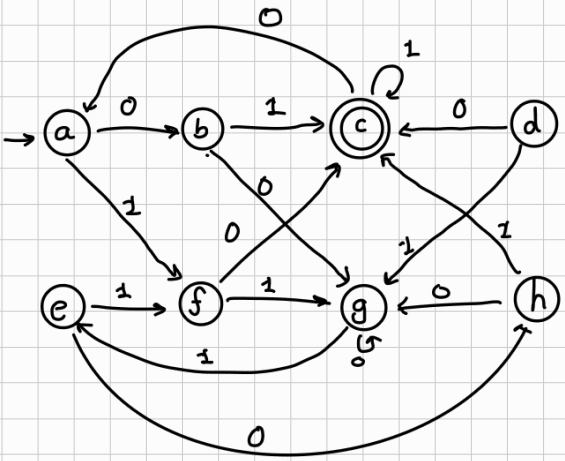
3]



d is unreachable!



state	0	1	✓/✗
ah	bg	fc	
ag	bg	fe	
af	bc	fg	✓
ae	bh	ff	
ac	ba	fc	
ab	bg	fc	
bh	gg	cc	
bg	gg	ce	
bf	gc	cg	
be	gh	cf	
ch	ag	cc	
cg	ag	ce	
cf	ac	cg	✓
ce	ab	cf	
eh	hg	fc	✓
eg	hg	fe	
ef	hc	fg	
fh	cg	gc	
fg	cg	ge	



	0	1
→ q	b	f
b	g	c
* c	a	c
e	h	f
f	c	g
g	g	e
h	g	c

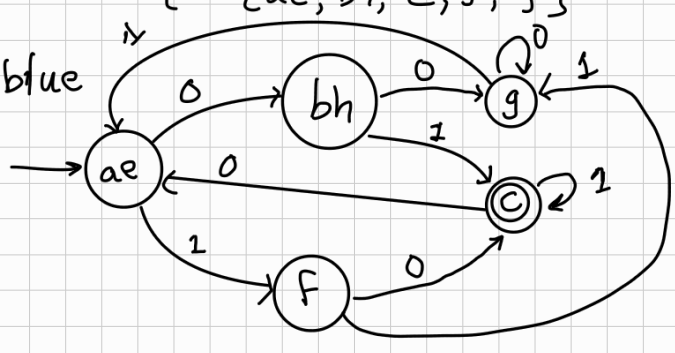
	0	1		next iter
ag	bg	fe	x	✓ (bg)
ae	bh	ff	x	x
bh	gg	cc	x	x
~~bg~~	gg	ce	✓	
bf	gc	cg	✓	
ch	ag	cc	x	✓
~~eg~~	hg	fe	✓	
ef	hc	fg	✓	

ae	bh	ff	x
bh	gg	cc	x

Iteration 2:

State	0	1	
ah	bg	fc	✓
gg	bg	fe	
ae	bh	ff	
ab	bg	fc	✓
bh	gg	cc	x
bg	gg	ce	x
bf	gc	cg	x
be	gh	cf	✓
ch	ag	cc	x
cg	ag	ce	✓
ce	ah	cf	✓
eg	hg	fe	x
ef	hc	fg	x
fh	cg	gc	✓
fg	cg	ge	✓
gh	gg	ec	✓

∴ Q = {ae, bh, c, f, g}



/ /

Closure property of Regular Language:

Closure property of a language class says that given languages in a class an operation produces another language in the same class.

if $L_1 \cup L_2 \in R.L.$ languages are closed under union.

e.g. $L_1 = \{ab\}$
$L_2 = \{ba\}$
} Regular langs

$L_3 = L_1 \cup L_2$

$L_3 = \{ab, ba\}$ .. it is possible to construct DFA.

$\therefore L_1, L_2$ closed under union.

Operations to check:

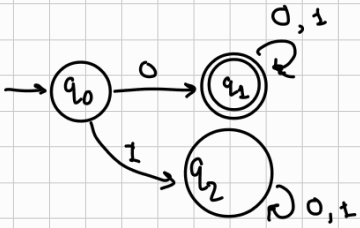
- Union
- Intersection
- Set difference
- Complement
- Concatenation

1) Union:

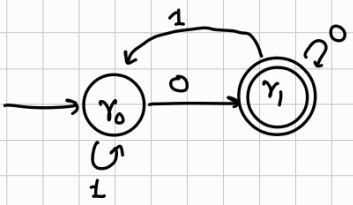
$L_1 = \{0x \mid x \in \{0,1\}^*\}$

$L_2 = \{x0 \mid x \in \{0,1\}^*\}$

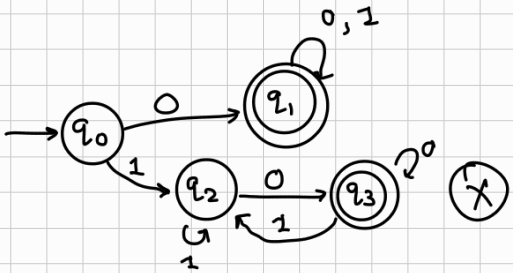
$L_1 \cup L_2 = \{0x, x0 \mid x \in \{0,1\}^*\}$



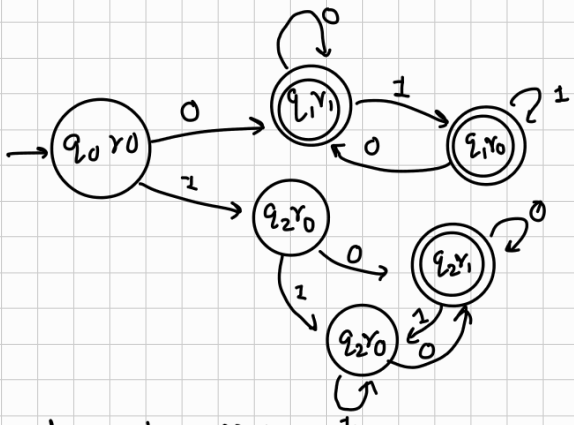
M_1 for L_1



M_2 For L_2



M_3 for $L_1 \cup L_2$



Lemma: Class of Regular languages is closed under Union.

Let $M_1 = (Q_1, \Sigma, \delta_1, q_0, F_1\}$

$M_2 = (R, \Sigma, \delta_2, r_0, F_2\}$

be the two DFA's for L_1 and L_2.

Product Construction:

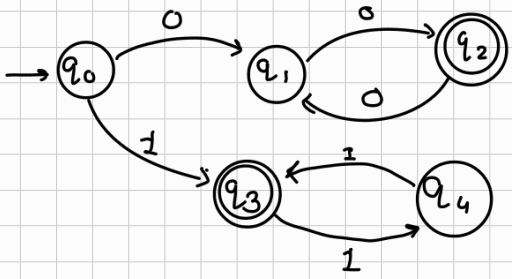
Create a DFA,

$M = (Q \times R, \Sigma, \delta, (q_0, r_0), F)$

$\delta((q_0, r_0), a) = (\delta_1$

L_1 L_2

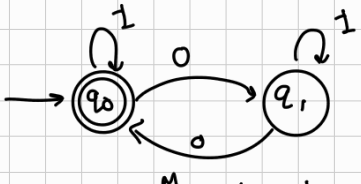
Q. DFA to accept even number of 0's OR odd number of 1's



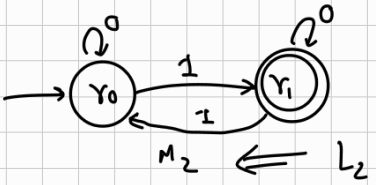
101110

L_1 = Even 0's

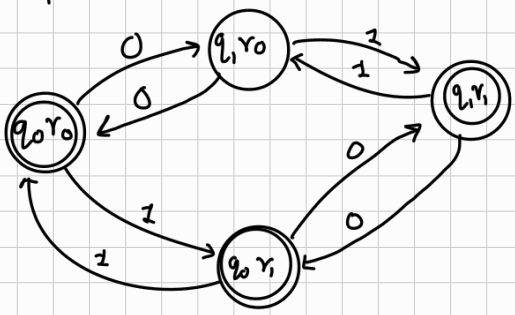
L_2 = Odd 1's



$M_1 \Leftarrow L_1$



$M_2 \Leftarrow L_2$



$M_1 \cup M_2 \Leftarrow L_1 \cup L_2$

2) Regular Language is closed under intersection!

$L_1 \cap L_2 = (L_1^c \cup L_2^c)^c$

C → Complement

let DFA for $M_{L_1} = (Q, \Sigma, \delta_1, q_0, F_1)$

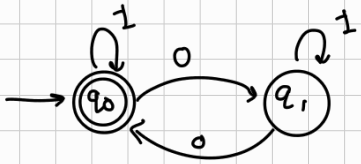
DFA for $ML_2 = (R, \Sigma, \delta_2, r_0, F_2)$

$M_{L_2 \cap L_2} = (Q \times R, \Sigma, \delta, (q_0 r_0), F)$

✗ both final states should be present.

1) EVEN Zeros

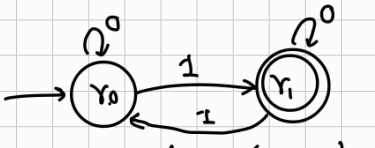
L_1 = Even 0's



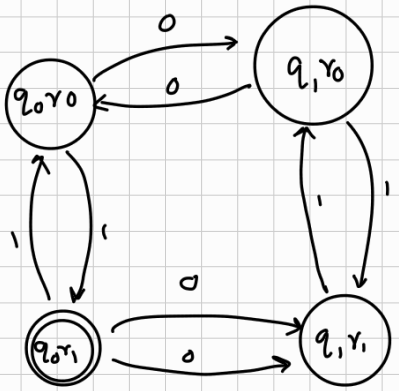
$M_1 \Leftarrow L_1$

2) Odd Ones

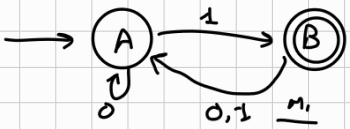
L_2 = Odd 1's



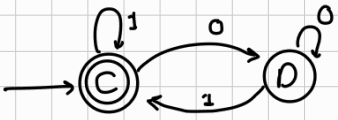
$M_2 \Leftarrow L_2$



Q) Construct product DFA for intersection

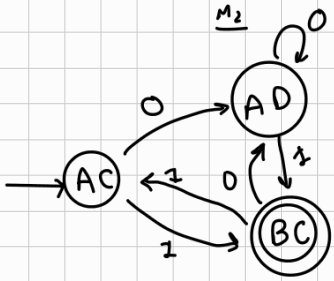


L_1 = odd ones



L_2 = start & end 1

M_2



	0	1
A	A	B
B	A	A
C	D	C
D	D	C

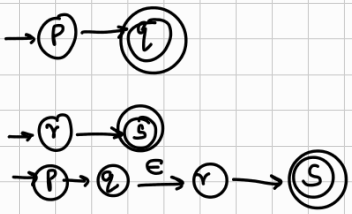
$M_3 = M_1 \cap M_2$

Concatenation:

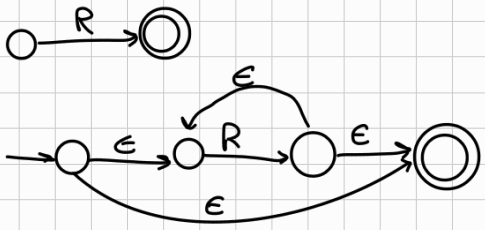
$L_1 = \{01, 10\}$

$L_2 = \{110\}$

$L_1L_2 = \{01110, 10110\}$



If R belongs to regular lang, R* (R closure) also is regular.



Complement of a language L with respect to Alphabet Σ, is $\Sigma^* - L$

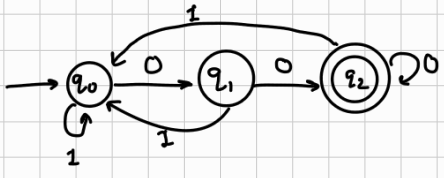
e.g. $\Sigma = \{0,1\}$ $L_1 = \{01, 10\}$

$L_1^C = \{0,1\}^* - L_2$

To construct compliment DFA, convert final states to non-final state and every non final states into a final state.

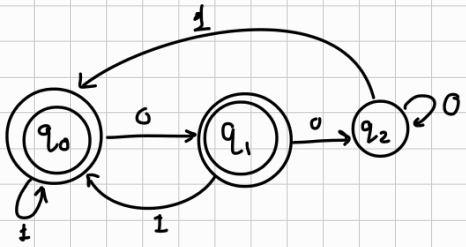
e.g.

Construct DFA for all strings ending with 00 in $\{0,1\}^*$



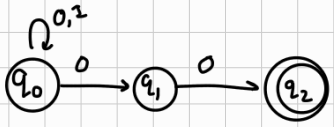
M_1

$L^C \Rightarrow M_1^C$:)



NFA to accept strings over {0,1} ending with (0,0).

NFA ⇒



NFA^C would be wrong

Set difference.

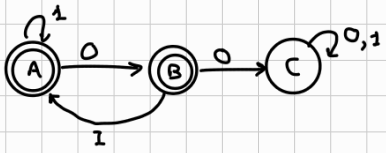
$L-M = L \cap M^C$

∵ ∩ and Complement closed in regular languages, L-M will be closed.

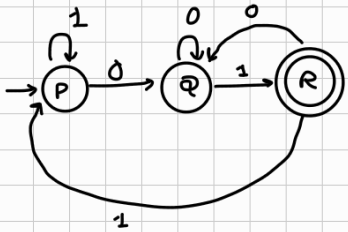
e.x) $L_1 = \{x \mid 00$ is not a substring of $x\}$

$L_2 = \{x \mid x$ ends with $01\}$

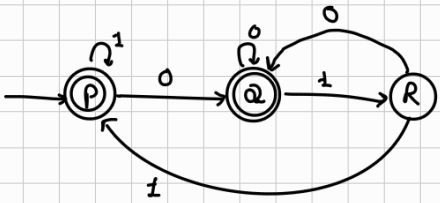
Construct a minimum FA accepting $L_1 - L_2$



M_1



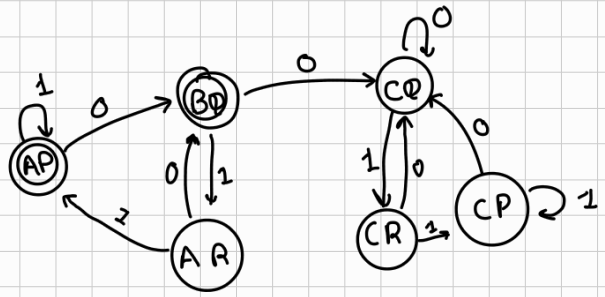
M_2



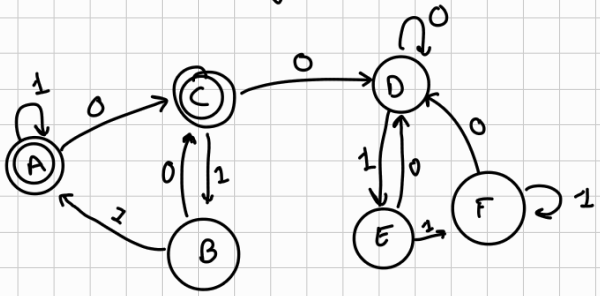
M_2^C

	0	1
→*A	B	A
*B	C	A
C	C	C

	0	1
→*P	Q	P
*Q	Q	R
R	Q	P

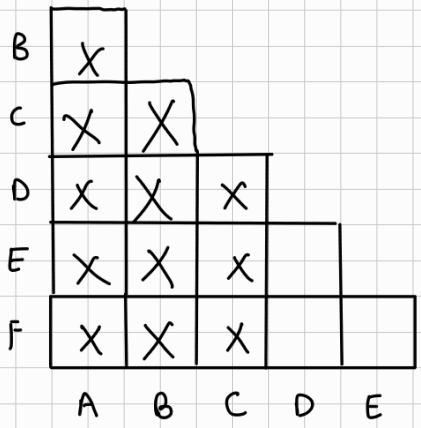


Renaming states



	0	1
A	C	A
B	C	A
C	D	B
D	D	E
E	D	F
F	D	F

	0	1	
A F	C D	A F	
E	C D	A D	
D	C D	A D	
B	C C	A A	
B F	C D	A F	✓
E	C D	A F	✓
D	C D	A E	✓
B C	C D	A B	✓
C F	D D	B F	
E	D D	B F	
D	D D	B E	
D F	D D	E F	X
D E	D D	E F	X
E F	D D	F F	X

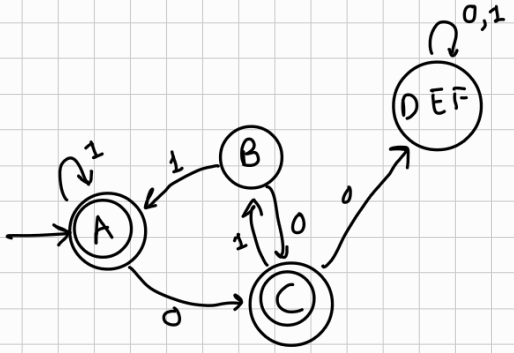


Iteration 2 :

AC	DD	AB	✓
DF	DD	EF	X
DE	DD	EF	X
EF	DD	FF	Y

Iteration 3

X



HW NFA to accept either ab or bba as substring

↓

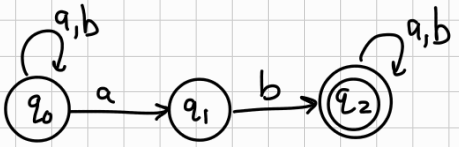
Convert to DFA

↓

Minimise DFA.

$L_1 = \{(a+b)^* ab (a+b)^*\}$

$L_2 = \{(a+b)^* bba (a+b)^*\}$

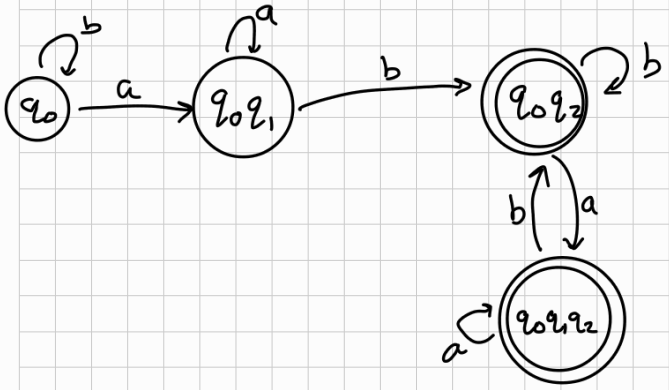


M_1

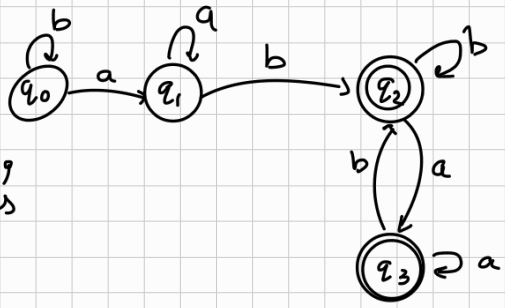
⇓

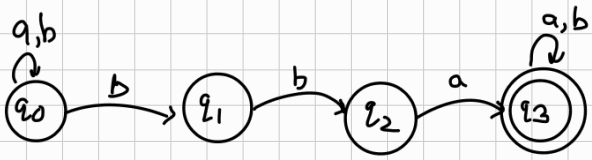
	a	b
q_0	q_0q_1	q_0
q_1	ϕ	q_2
q_2	q_2	q_2

	a	b
q_0	q_0q_1	q_0
→ q_0q_1	q_0q_1	q_0q_2
q_0q_2	$q_0q_1q_2$	q_0q_2
$q_0q_1q_2$	$q_0q_1q_2$	q_0q_2



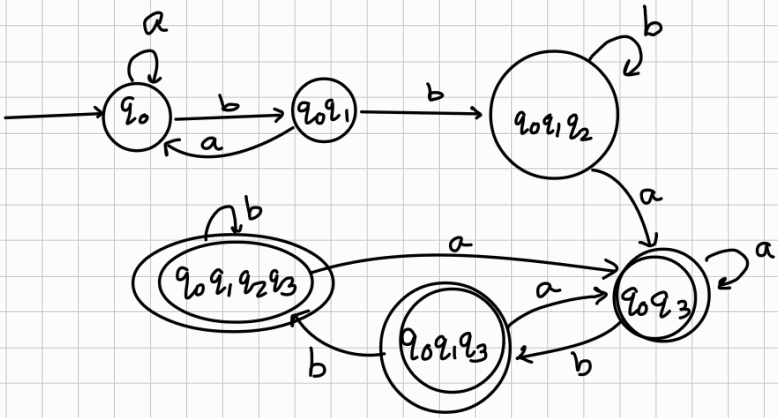
⇒ Renaming states

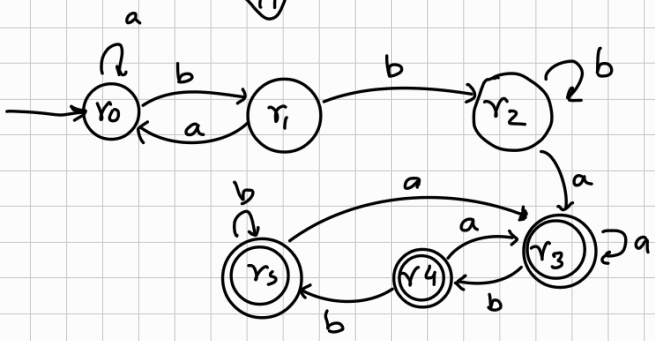


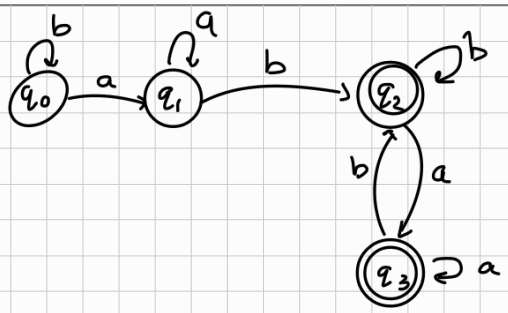


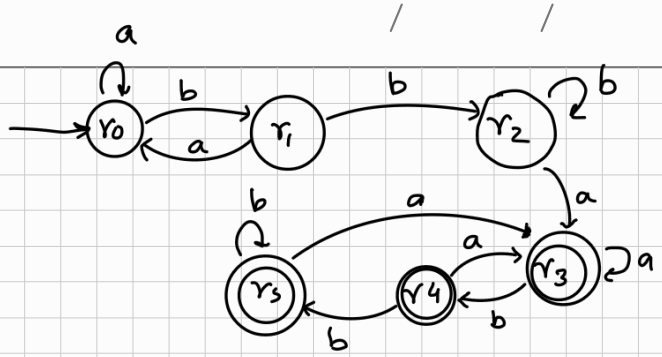
M_2

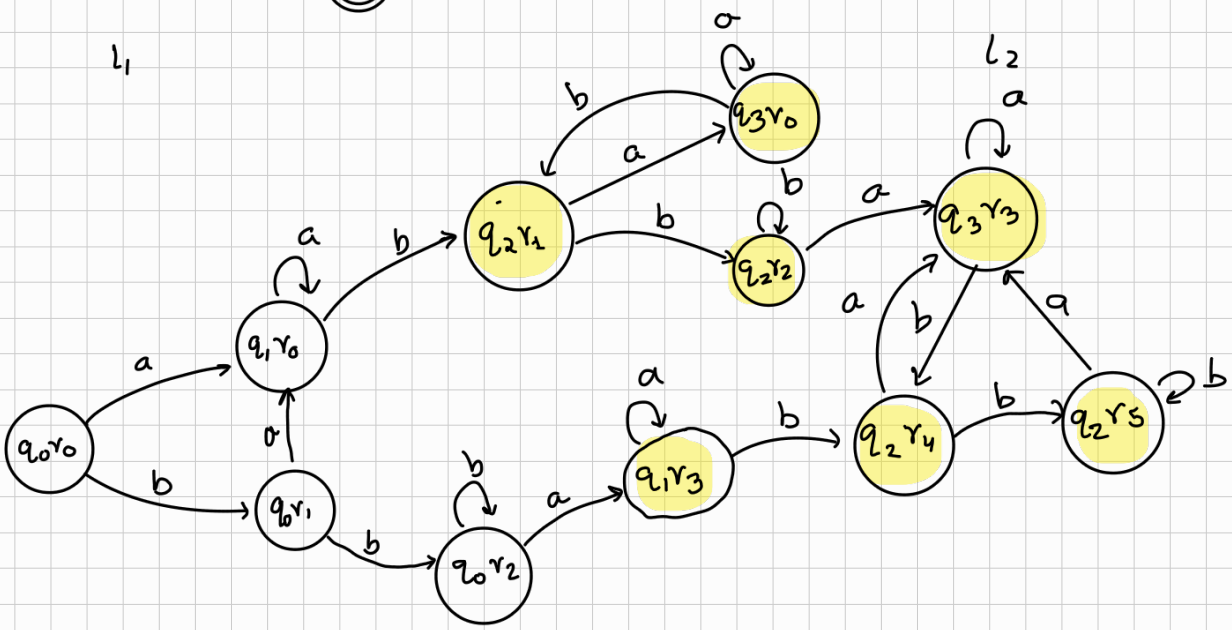
	a	b
→ q_0	q_0	q_0q_1
q_1	ϕ	q_2
q_2	q_3	ϕ
q_3	q_3	q_3
q_0q_1	q_0	$q_0q_1q_2$
$q_0q_1q_2$	q_0q_3	$q_0q_1q_2$
q_0q_3	q_0q_3	$q_0q_1q_3$
$q_0q_1q_3$	q_0q_3	$q_0q_1q_2q_3$
$q_0q_1q_2q_3$	q_0q_3	$q_0q_1q_2q_3$

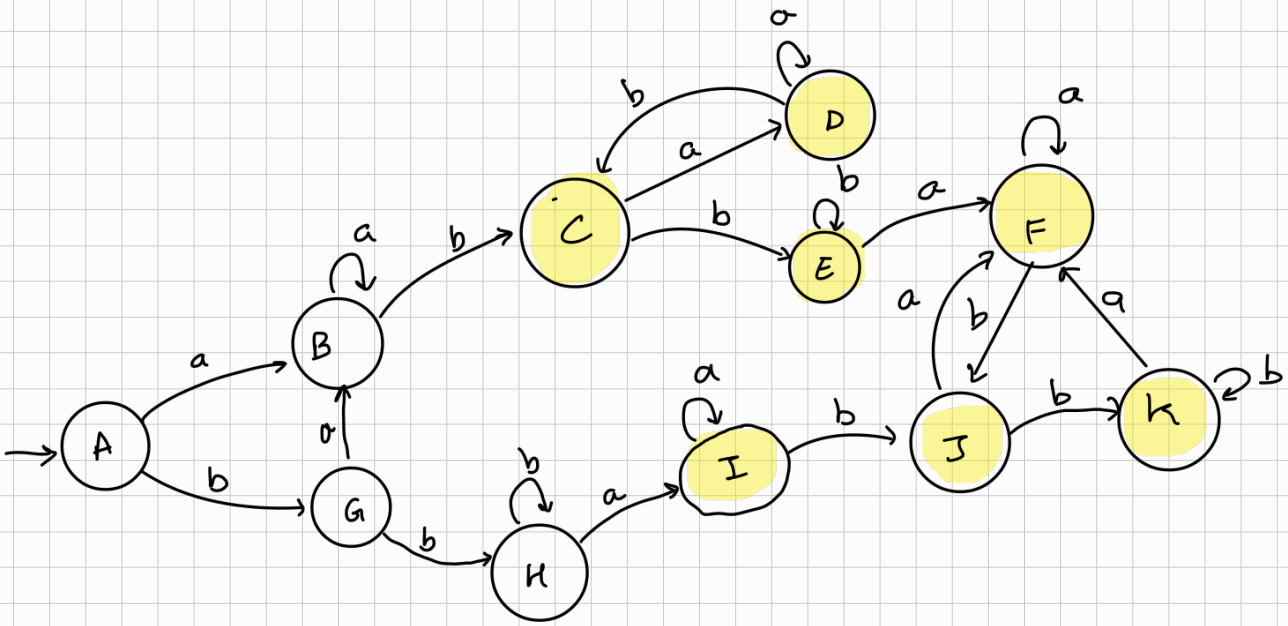


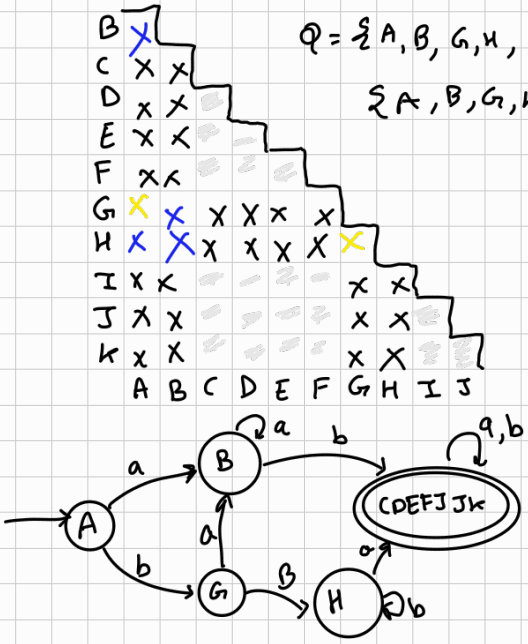



q0
q1
q2
q3
a
b


r0
r1
r2
r3
r4
r5
a
b


l_1
l_2
q_0r_0
q_0r_1
q_0r_2
q_1r_0
q_1r_3
q_2r_1
q_2r_2
q_2r_4
q_2r_5
q_3r_0
q_3r_3
a
b


A
B
C
D
E
F
G
H
I
J
K
a
b



$Q = \{A, B, G, H, CDEFIJK\}$

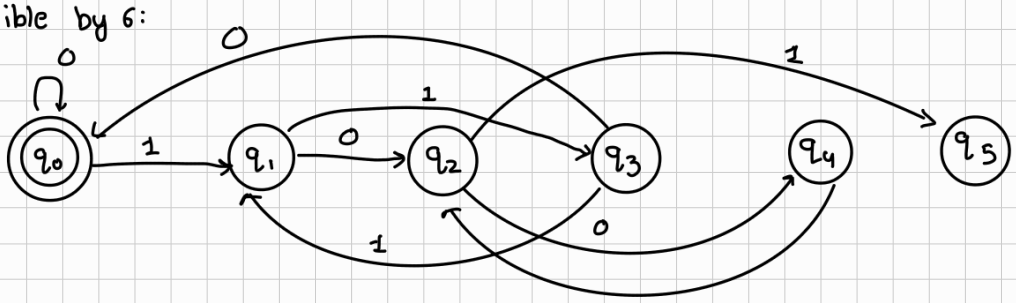
$\{A, B, G, H, \emptyset\}$

		a	b
→	A	B	G
	B	B	C
*	C	D	E
*	D	D	C
*	E	F	E
*	F	F	I
	G	B	H
	H	I	H
*	I	I	J
*	J	F	K
*	K	F	K

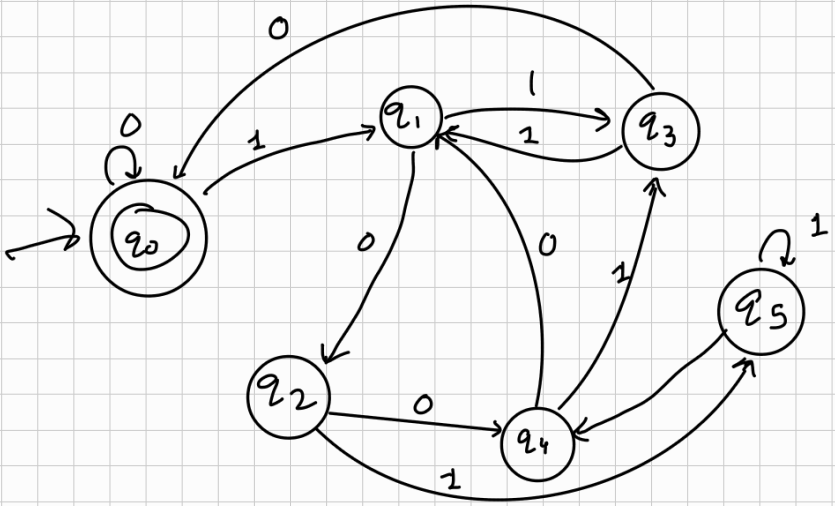
GH = BI HH

		0	1	
A	H	B I ✓	G H	✓
A	G	B B	G H	X
A	B	B B	G C ✓	✓
B	H	B I	C H ✓	✓
B	G	B B	C H	✓
C	K	D F	E K	X
C	J	D F	E K	X
C	I	D I	E J	X
C	F	D F	E J	X
C	E	D F	E E	X
C	D	D D	E C	X
D	K			
D	J			
D	I			X
D	F			
D	E			
E	K			
E	J			X
E	I			
E	F			
F	K			
F	J			X
F	I			
G	H			
I	K	BI ✓	HH	✓
I	J			
J	K			

Divisible by 6:



100|0 8
1 3



Pumping Lemma:

There exists a constant $n \geq 0$ s.t. if you take a string 'w' from language 'L' such that:

$$|w| \geq n ,$$

You divide string w in 3 parts such that:

$$|y| > 0$$
$$|xy| \leq n$$
$$i \geq 0 , \quad xy^i z \in L$$

Step:

① State Pumping Lemma
② Assume language is regular
③ Proof by contradiction.

Q) $L = \{a^i b^j c^k \mid i+j = k , i \geq 0 , j \geq 0, k \geq 0\}$

Assume L is regular.

let,

$w = a^m b^{n-m} c^n$

$x = a^m$
$y = b^{n-m}$
$z = c^n$

$|y| > 0$

$|xy| \leq n$

for $i = 0$, $xz = a^m c^n \notin L$

$i = 2$ $xy^2z = a^m b^{2(n-m)} c^n \notin L$

$\therefore$ Contradiction, language is not regular.

$w = a^n b^n c^{2n}$ $\quad w = xyz$

$x = a^{n-k}$
$y = a^k$
$z = b^n c^{2n}$

$i = 3$

$xy^3z = a^{n-k} a^{3k} b^n c^{2n}$

$n + 2k + n > 2n$

Q. $L = \{0^i 1^j \mid i \geq j\}$

Assume L is regular

let $w = 0^n 1^n$ $\quad \because i \geq j$ $\quad i = j$ is allowed

$x = 0^{n-k}$ where $k > 0$.
$y = 0^k$
$z = 1^n$

$i = 0$, $xy^0z = 0^{n-k} 1^n \notin L$ as 0^i must be $\geq j$

$L = \{0^p \mid \text{where } p \text{ is a prime number}\}$

$\omega = 0^n \in L$ let n be a prime number.

$|\omega| = |0^n| = n$

$|xy^i z| = n + (i-1)y$

$|xyz|$ since we used i once, we add the remaining i's as $(i-1)y$.

let $i = n+1$

$= n + (n+1-1)y$

$= n + ny$

$= n(1+y)$ ... which is a composite number

∴ by Contradiction, language is irregular.

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

$n \leq |\omega|$

$2a + b \geq n$

let n be the pumping constant.

$\omega = 0^a 1^b 2^a$

$x = 0^{a-k}$

$y = 0^k$

$z = 1^b 2^a$

for $i = 0$,

$xy^0 z = 0^{a-k} 1^b 2^a$

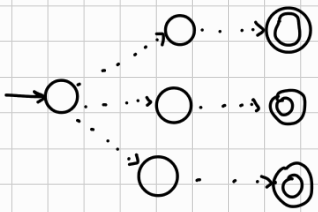
Reversal:

$L = \{0, 01, 100\}$

$L^R = \{0, 10, 001\}$

Steps:

Suppose a DFA like

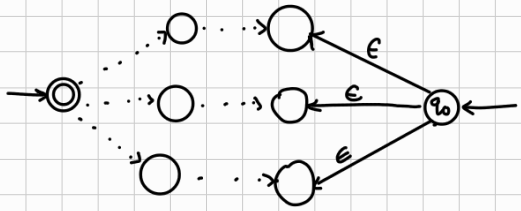


- Create ϵ-NFA.

  Step 1: Add a new start state

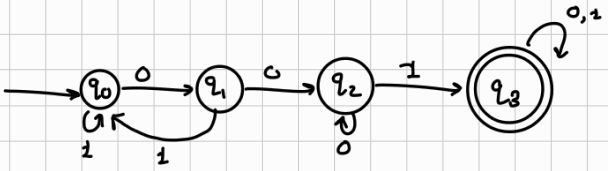
  Step 2: Make a ϵ-move from new start state to all previous final states. Make final states non-final (ϵ-transitions from new start to all final).

  Step 3: change direction of all arrows (reverse all arrows).

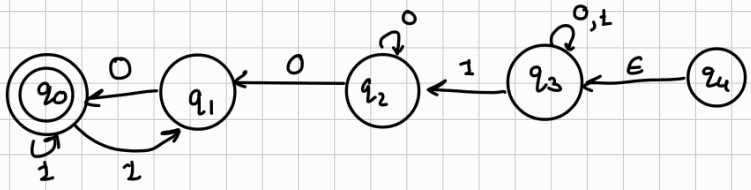


Step 4: change old start state to final state.

$L = \{w \mid$ substring $001\}$



$L^R =$



Homomorphism :

It is an operation $h: \Sigma \to T^*$ where each symbol in Σ is substituded by a string from T.

$\Sigma = \{0,1\}$
$T = \{a,b\}$
$h(0) = ab$
$h(1) = \epsilon$

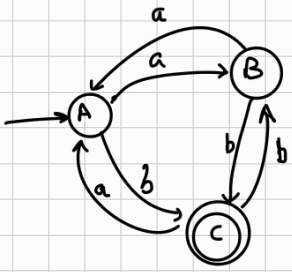
$\omega = 10110$
$h(\omega) = \epsilon ab \epsilon \epsilon ab$
$= abab$

Replace every edge with symbol $a \in \Sigma$ by a path labelled $h(a)$

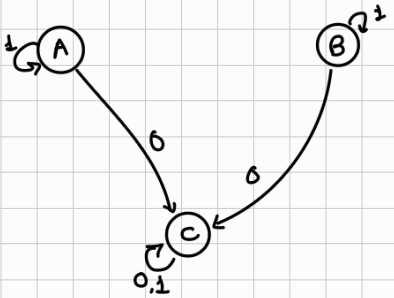
Inverse Homomorphism :

Let h be a homomorphism on $h: \Sigma \to T^*$, let M be a language over T.

The $h^{-1}(M) = \{\omega \mid h(\omega) = M\}$

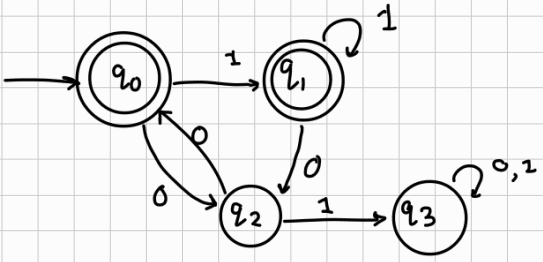


$h(0) = ab$
$h(1) = \epsilon$

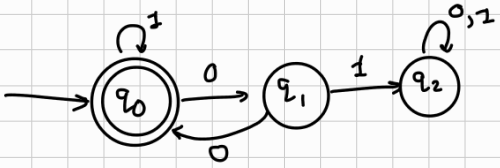


1 is ϵ,
you can mark everywhere

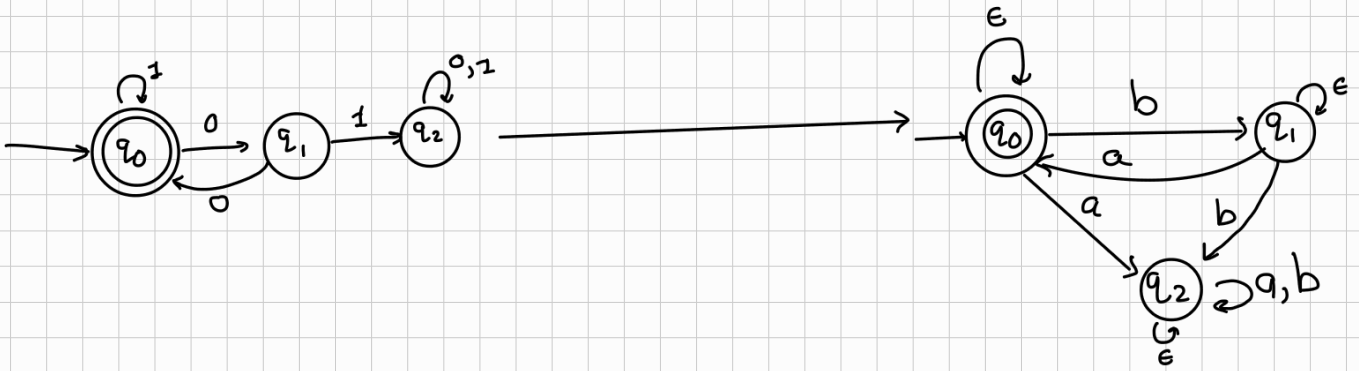
$L = (00+1)^* = \{00, 1, 0000, 001, 100, 11 \ldots\}$
$h(a) = 01$
$h(b) = 10$

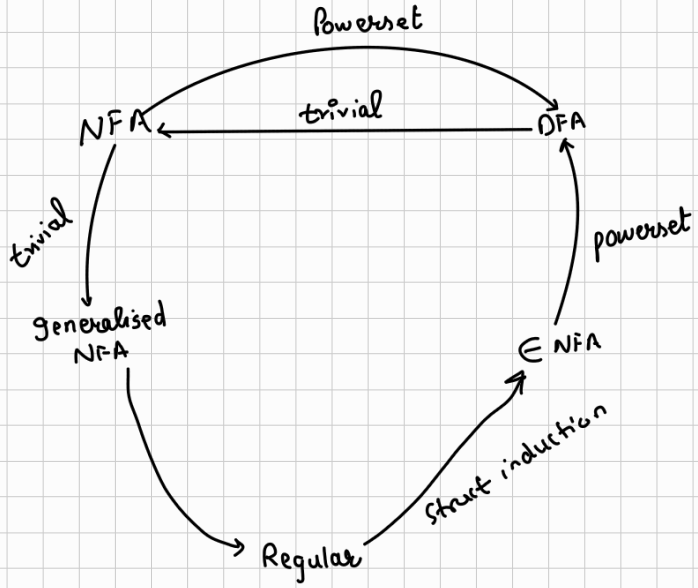


optimal



$h(a) = 01$
$h(b) = 10$





Another way of describing regular language by using additional symbols such as *, +, ..

Defn: 'R' is a regular expression with following conditions:

1. if $a \in \epsilon$ (epsilon), a is a regular expression, having the language $L = \{a\}$
2. ϵ is a regular expression with language $L = \{\epsilon\}$
3. ϕ is a regular expression with language $L = \phi$ i.e. $L = \{\}$
4. $R_1 + R_2$ -11- where '+' represents Union.
5. $R_1 \cdot R_2$ or $R_1 R_2$ -11- where '.' represents concatenation.
6. R^* -11- where '*' represents closure
7. (R)

Q.) find regular expression over $\Sigma = \{a,b\}$ with strings of exactly length 2 or more

$$R_1 = (a+b)(a+b)\ (a+b)^*$$

2.) Language of even length.

$$R_2 = ((a+b)(a+b))^*$$

3) Containing substring 'ab'

$$R_3 = (a+b)^*\ ab\ (a+b)^*$$

4) string of length atmost 2

$$R_4 = (a+b+\epsilon)\ (a+b+\epsilon)$$

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

$L = \{w \mid$ without substring $0,0\}$

$$R_5 = (1+01)^*\ (0+\epsilon)$$

Rules:

$R\emptyset = \emptyset$

$R\epsilon = R$

$R+\emptyset = R$

$\{a\} \cup \{\} = \{a\}$

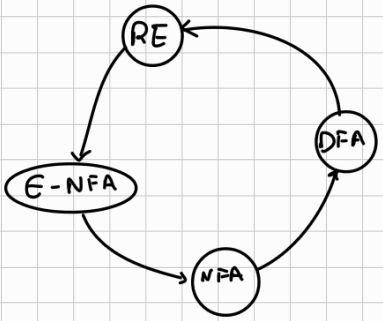
Check whether two regular expressions are same or not:

① $(01)^*0$ $\quad L_1 = \{0, 010, 01010 \ldots\}$

② $0(10)^*$ $\quad L_2 = \{0, 010, 01010 \ldots\}$

$L_1 = L_2$

$\therefore$ same



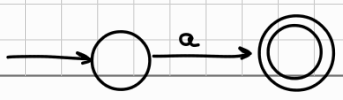
1) RE to ϵ-NFA

basis

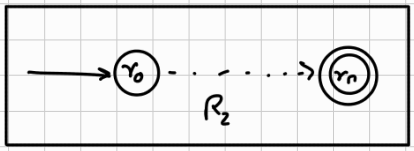
$L = \emptyset$

$L = \{\epsilon\}$

$L = \{a\}$

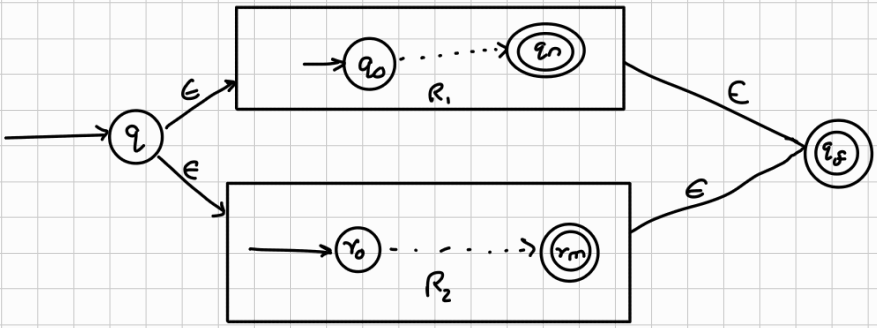


R_1 - Regular expression 1

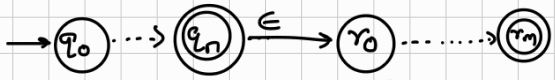


R_2 - Regular expression 2.

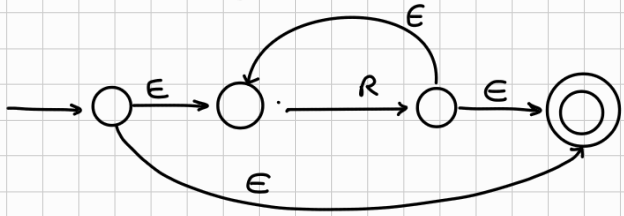
→ $R_1 + R_2$



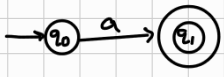
$R_1 \cdot R_2$ (Concatenation):

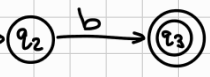


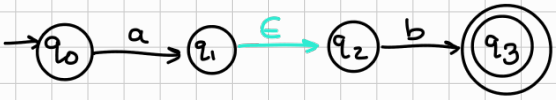
R^*

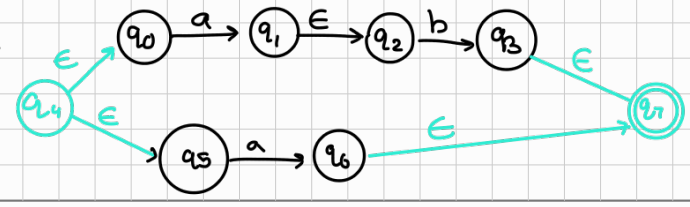


Q) Convert $(ab+a)^*$

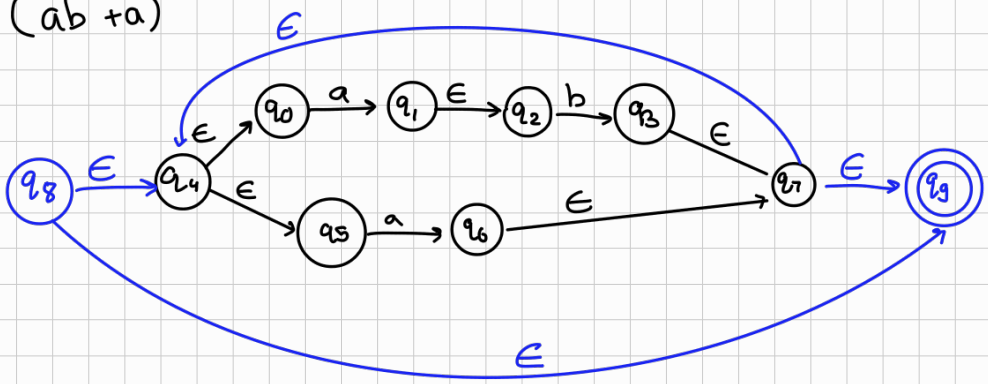
a. 

b 

ab 

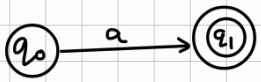
ab + a 

$(ab + a)^*$

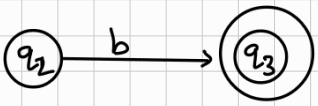


find regex for $a + (bc)^*$

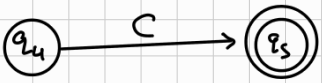
a



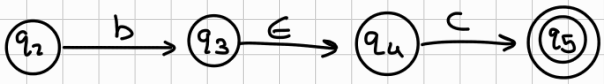
b



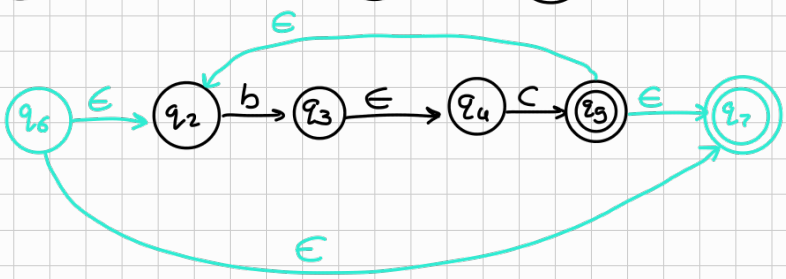
c



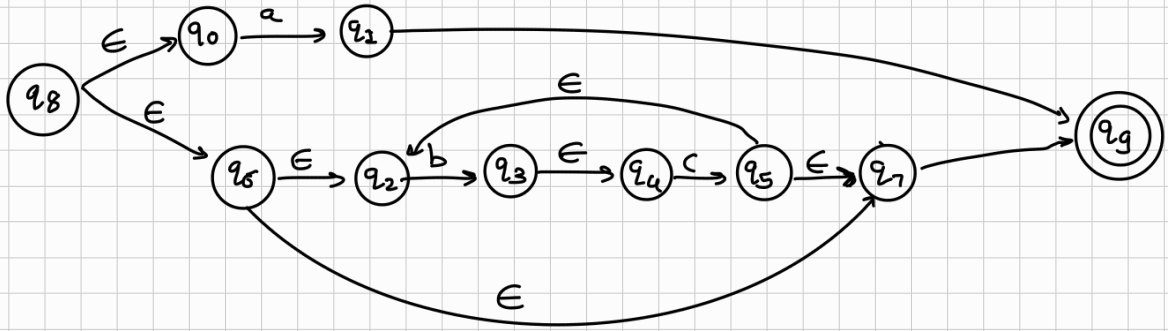
bc



$(bc)^*$



$a + (bc)^*$



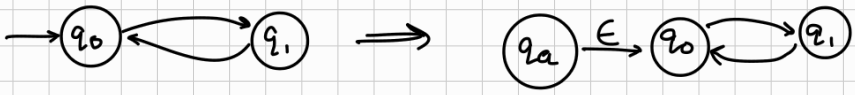
DFA to Regular Expression:

1) State Elimination method

2) Arden's theorem.

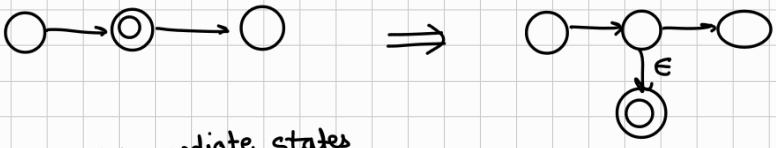
1) Initial state of DFA shouldn't have any incoming edge, if there exist any incoming edge, then create new initial state without having any edge to it.

e.g.



② There must exist only one final state, if there are multiple final states, convert final states to non final states and make ϵ-transition from all prev. final states to new final state.

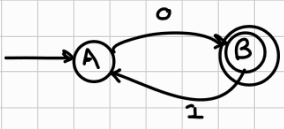
③ Final state of DFA must not have any outgoing edge, if there is any outgoing edge create a new final state



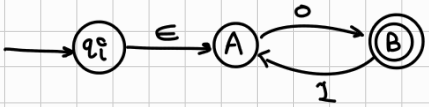
④ Eliminate all the ^ intermediate states one by one in any order, at the end there will be only initial and final state remaining.

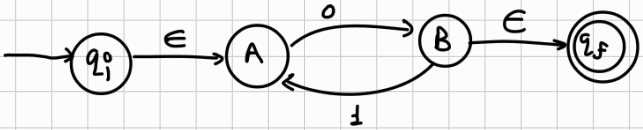
- Cost on the edge will be the required regular expression.

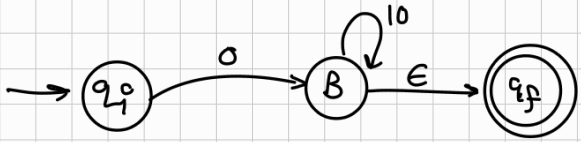
Find the regular expression

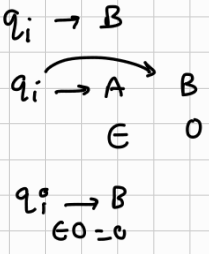


R = 0(10)*



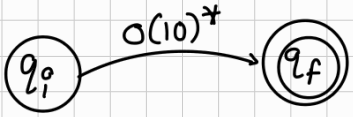




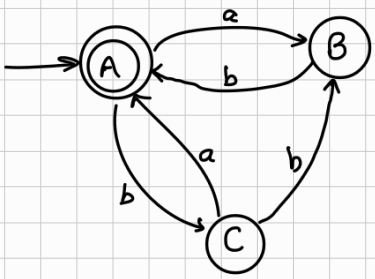


$q_0 \rightarrow q_f$ / /

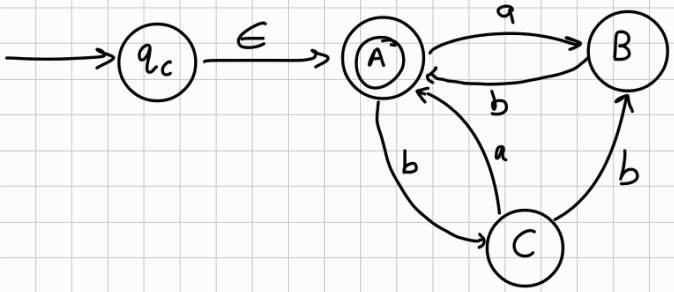
$q_i \rightarrow B \rightarrow q_f$
$0 \quad 10 \quad \epsilon$



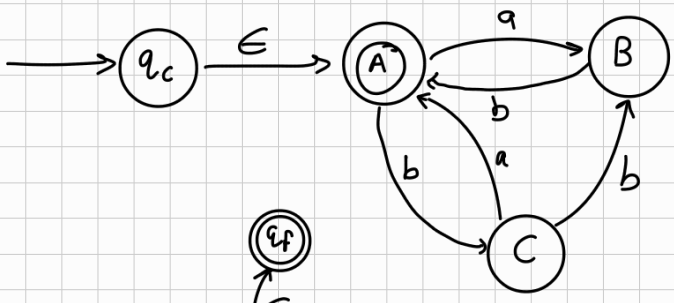
Regex → $0(10)^*$



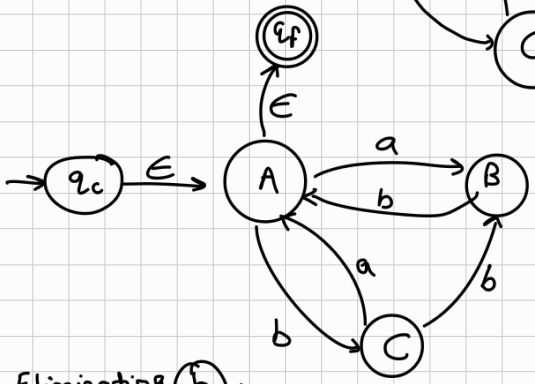
step 1:



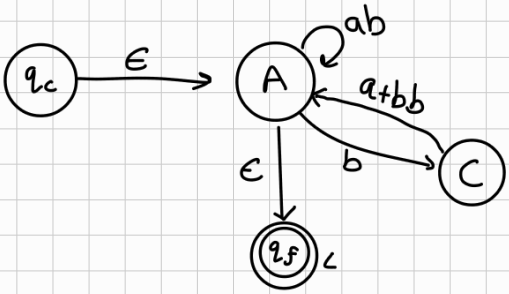
(2)



(3)

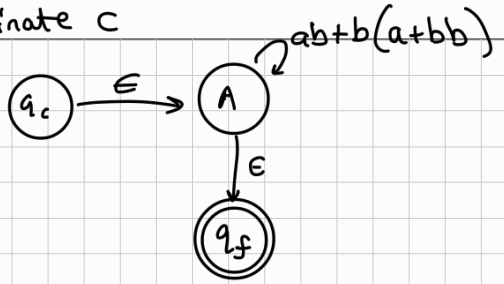


(4) i. Eliminating (b):

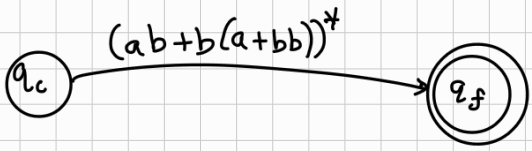


$q_c \xrightarrow{\epsilon} A \xrightarrow{\epsilon} F$

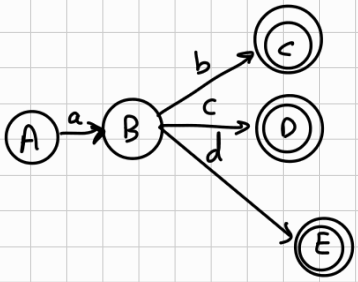
ii) Eliminate C



iii) Eliminate A



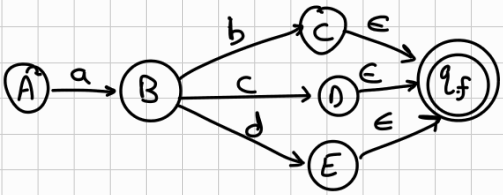
Q)



Step 1:

No change

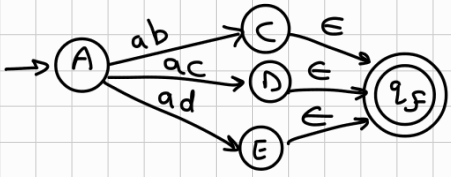
Step 2:



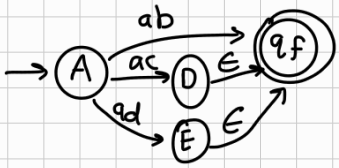
Step 3:

No change

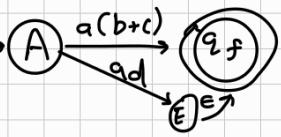
Step 4: Removing (B)



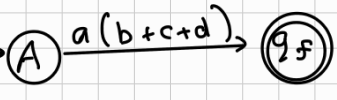
R... (C)

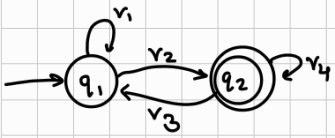


Remove (D)

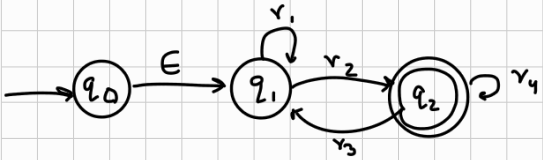


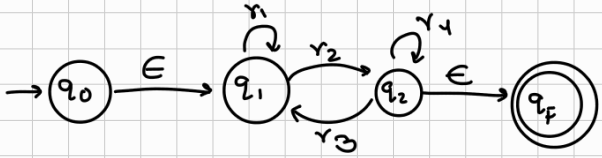
Remove (E)





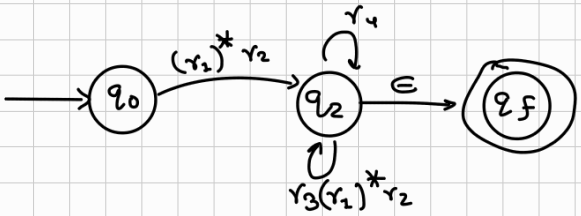
// Solve again by eliminating q_2 first:

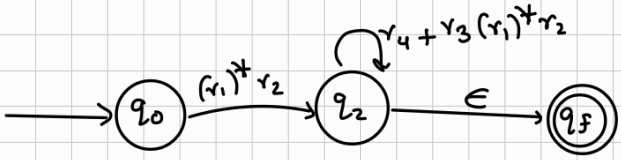


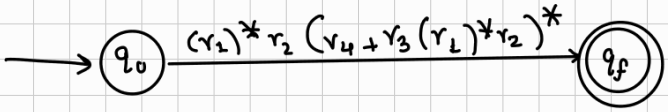


No outgoing for final state, not even on self!

Step 4:
Eliminate q_1







$\therefore R.E = r_1^* r_2 (r_4 + r_3 r_1^* r_2)^*$

---

① $R + \phi \Rightarrow \{R\} \cup \{\} = \{R\}$

② $R\epsilon \Rightarrow R\epsilon = R$

③ $R + \epsilon \neq R$, only $= R$ if R is ϵ

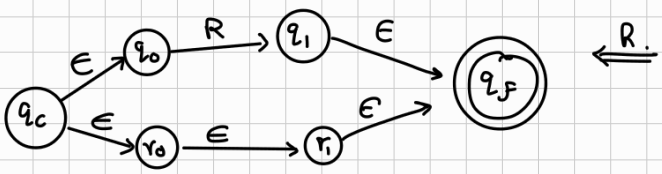
④ $R.\phi = \phi$

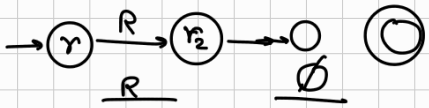
⑤ $\epsilon^* = \epsilon$

⑥ $\phi = \epsilon$

⑦ $(R^*)^* = R^*$

⑧ $(\epsilon + R)^* = R^*$





(9) $r+s = s+r$

(10) $rs(t) = r(st)$

(11) $(r+s)^* = (r^*+s^*)^*$

(12) $(r+s)t = rt+st$

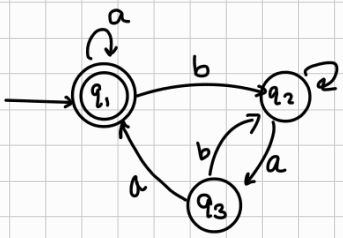
Arden's

Let P and Q be two regular expressions and P does not include ϵ. Then if $R = Q+RP$, then $R = QP^*$

$R = Q+RP$
$R = (QP)^*$

$R = Q+RP$
$R = Q+(Q+RP)P$
$R = Q+QP+RP^2$
$R = Q+QR+(Q+RP)P^2$
$R = Q+QR+QP^2+RP^3 \ldots.$

Transition diagram must not have ϵ transitions, it must have



Create equations for all states, for initial state add ϵ.

$q_1 = q_1a + q_3a + \epsilon$ ⟶ $q_1 = q_1a + q_2aa + \epsilon$

$q_2 = q_1b + q_3b + q_2b$

$q_3 = q_2a$

$q_2 = q_1b + q_2b + (q_2a)b$

$q_2 = q_1b + q_2(b+ab)$ ... form $R = Q+RP$, $R = QP^*$

(q_2: R, q_1: Q, q_2: R, $(b+ab)$: P)

$q_2 = q_1b(b+ab)^*$

$q_1 = q_1 a + q_3 a + \epsilon$

$= q_1 a + q_2 aa' + \epsilon$

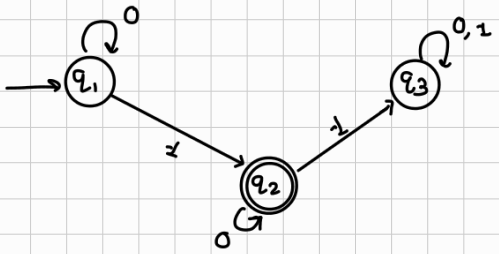
$= q_1 a + (q_1 b(b+ab)^*) aa + \epsilon$

$= q_1 a + q_1 b(b+ab)^* aa + \epsilon$

$q_1 = q_1 (a + b(b+ab)^* aa) + \epsilon$

$\underset{R}{q_1} = \underset{Q}{\epsilon} + \underset{R}{q_1}\underbrace{(a+b(b+ab)^* aa)}_{P}$

$q_1 = \epsilon (a + b(b+ab)^* aa)$



$q_1 = q_1 0 + \epsilon \Rightarrow$ form $R = Q + RP^* \Rightarrow q_1 = \epsilon + q_1 0 \Rightarrow \epsilon 0^* \Rightarrow 0^*$

$q_2 = q_1 1 + q_2 0$

$q_3 = q_3 0 + q_2 1 + q_3 1$

$\underset{R}{q_2} = \underbrace{0^* 1}_{Q} + \underset{R}{q_2}\underset{P}{0}$

$q_2 = QP^* \Rightarrow \boxed{0^* 1 0^*}$

Simplify: $(1+00^*1) + (1+00^*1)\underbrace{(0+10^*1)^*}_{R^*}\underbrace{(0+10^*1)}_{R}$ $\qquad R^* R = R^+$

$= (1+00^*1)(\epsilon + (0+10^*1)^*(0+10^*1))$

$= (1+00^*1)(\epsilon + (0+10^*1)^+)$

$= (1+00^*1)(0+10^*1)^*$

$= (\epsilon + 00^*)1(0+10^*1)^*$

$= 0^*1(0+10^*1)^*$

Simplify: $0^* + 0^*1(\epsilon + 00^*1)000^*$

$0^*(\epsilon + 1(\epsilon + 00^*1)000^*)$

$0^*(\epsilon + 1(\epsilon + 0^+1)00^+)$

$0^*(\epsilon + 1(0^*1)00^+)$

$0^*(\epsilon + 10^*100^+)$

$0^*(10^*100^*)$

$0^*10^*100^*$

(X)

$0^* + 0^*1(\epsilon + 00^*1)^*000^*$

$= 0^* + 0^*1(00^*1)^*000^*$

$\epsilon + 00^* + 0^*1(00^*1)^*000^*$ $(\epsilon + a)^* = a^*$

$\epsilon + 00^* + 0^*10(0^*10)^*00^*$

$r(st) = (rs)t$

$= \epsilon + (\underbrace{\epsilon}_{\epsilon} + \underbrace{0^*10}_{R}\underbrace{(0^*10)^*}_{R^*})00^*$

$\epsilon + R^+R^* \rightarrow \epsilon + R^*$

$= \epsilon + (0^*10)^*)00^*$

/ /

Context Free Languages:

- FA to accept regular languages.