Minimization of DFA – Done using Table filling algorithms.

~~Step~~: Identify

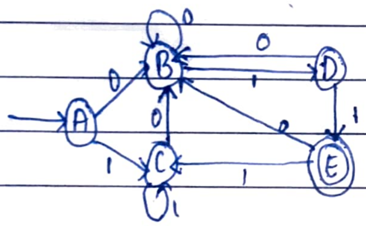
Two states p & q of DFA are called indistinguishable if
$\delta^*(p,w) \in F$ implies $\delta^*(q,w) \in F$ / $\delta^*(p,w) \notin F$ implies $\delta^*(q,w) \notin F$

Two states p & q of DFA are called distinguishable if
$\delta^*(p,w) \in F$ & $\delta^*(q,w) \notin F$

Steps:

1. Remove all inaccessible states.
   ~~Any path~~ state not part of some path is inaccessible.
2. Consider all pairs of states (p, q)
   If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat until no ~~sta~~ previously unmarked pairs are marked. ~~pairs are marked~~.

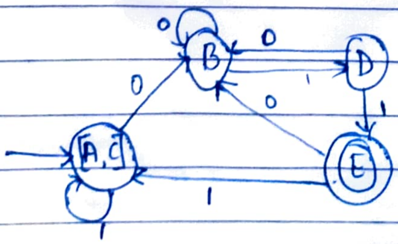
Ex:



States	0	1
→A	B	C
B	B	D
C	B	C
D	B	E
*E	B	C

Since all states reachable, so removing isn't done

B	X			
C	=	X		
D	X	X	X	
E	X	X	X	X
	A	B	C	D



Step 1: Remove state which does not

Step 1: Remove states which doesn't have any incoming connection

Step 2: In the table, mark all box with final and non final combination
i.e (E,A) (E,B) (E,C) (E,D)

Step 3: For remaining pairs, if any pair on inputs goes to one final and other non final then mark it.

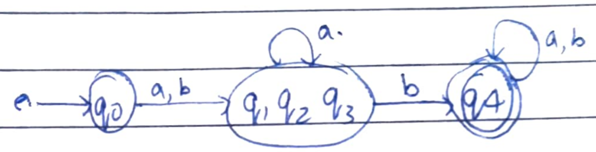
Ex: In pair (A,D) A on 1 → C
D on 1 → E
E is final state & C is not
so (A,D) is distinguishable.
marked with X

If any pair on applying input, it goes to other pair which determinable then if it is distinguishable then the current pair is also distinguishable
Ex: Pair (A,B) on 1 going to (C,D) pair where pair (C,D) is distinguishable.
∴ (A,B) is also distinguishable.

Ex:

	a	b
→q0	q1	q3
q1	q2	q4
q2	q1	q4
q3	q2	*q4
*q4	q4	q4

q1	X			
q2	X			
q3	X			
q4	X	X	X	X
	q0	q1	q2	q3



Ex:

	0	1
→q0	q1	q2
q1	q2	q3
q2	q2	q4
*q3	q3	q3
*q4	q4	q4

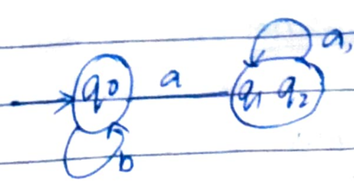
q1	x			
q2	x			
q3	X	x	x	
q4	x	x	X	
	q0	q1	q2	q3

Ex:

	a	b
→q0	q1	q0
*q1	q2	q1
*q2	q1	q2

q1	X	
q2	X	=
	q0	q1

(q1, q2) q0



Ex:

	0	1
→A	C	D
B	H	D
C	F	E
D	E	F
*E	A	E
F	F	B
(G	E	F)
H	F	E

G is inaccesible state so we can remove it

B						
C	x	x				
D	x	x				
*H		x				
F	x	x				
*E	X	X	X	X	X	X
	A	B	C	D	F	H

Regular Expression

more program syntax like

Regular expression → Regular language

Finite Automata

Operator

Union

~~Concatation~~ Concatenation

Kleene closure (* operator) = L^* is infinite set iff $|L| \geq 1$ and $L \neq \{\varepsilon\}$

Formulas

$a + \phi = a$ $\qquad (0+\epsilon)^* = 0^*$

$a + \epsilon = a + \epsilon$

$a \cdot \phi = \phi$

$a \cdot \epsilon = a$

Ex: Regular expression to accept a) one occurence of a.

b) One or more occurence of a

$a \cup aa^*$

c) Zero or more occurence of ab = $(ab)^*$

d) One or more occurence of ab = $ab(ab)^*$

e) Strings of a's & b's of any length = $(a+b)^*$

f) Strings of a's & b's of length four = $(a+b).(a+b).(a+b).(a+b)$

g) Alternate occurence of a's & b's = $(ab)^* + (ba)^* + a(ba)^* + b(ab)^*$ (or) $(a+\epsilon)(ba)^* + (b+\epsilon)(ab)^*$

h) strings of a's & b's of length four & less = $(a+b+\epsilon).(a+b+\epsilon).(a+b+\epsilon).(a+b+\epsilon)$

i) Strings of a's & b's of length 2 to 4 = $(a+b).(a+b).(a+b+\epsilon).(a+b+\epsilon)$

j) Strings of a's & b's with atleast 1 a's = $(a+b)^* a (a+b)^*$

k) strings of a's & b's exactly 2 a's = $\cancel{(a+b)^* a (a+b)^*}$ $b^* a b^* a b^*$

l) strings with 0's & 1's with even length =

m) Strings with odd no. of 1's =

n) Strings with 0's & 1's with length 6 or less =

o) Strings ~~with~~ of 0's & 1's ending with 1 & not containing consecutive zeroes

p) Strings of a's & b's beginning & ending with same character

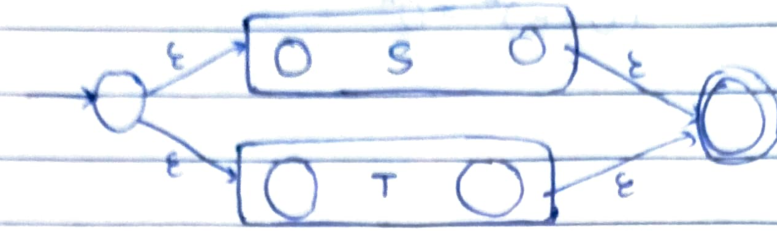
q) Strings of a's & b's with atleast one a and one b.

r) String with next to last symbol is 0

s) Strings of 0's & 1's not containing consecutive zero

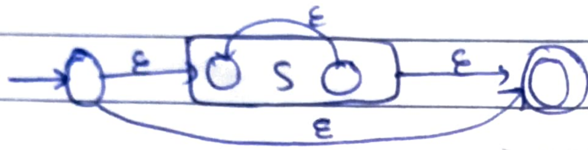
Conversion from RE to ε-NFA

$R = S + T$

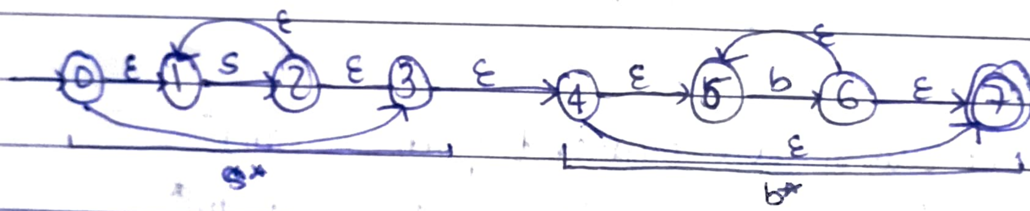


$R = ST$

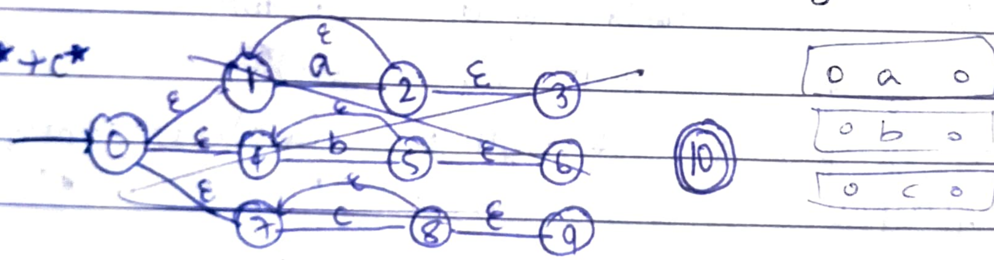
$R = S^*$



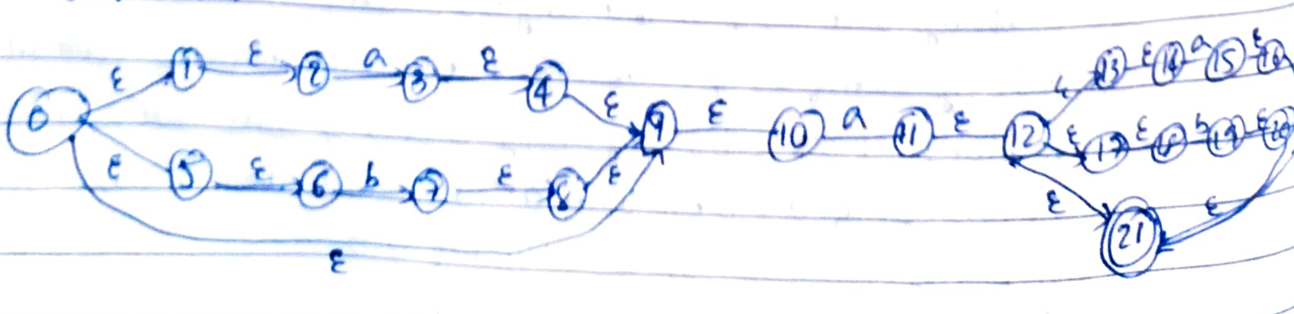
$R = S^* . b^*$



$R = a^* + b^* + c^*$

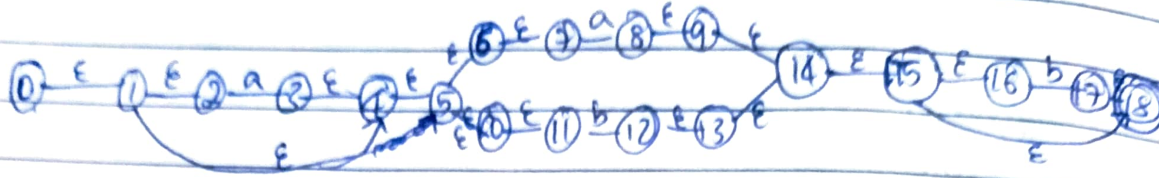


$R = (a+b)^* a (a+b)^*$

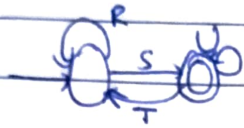


DATE: PAGE:

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



Conversion DFA to RE

Standard form:  $(R+SU^*T)^*SU^*$

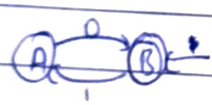
Ex: 

$R = \varepsilon \quad T = 1$
$S = 0 \quad U = \varepsilon$

$(\varepsilon + 0\varepsilon^* 1)^* \; 0\varepsilon^*$

$(\varepsilon + 01)^* \; \cancel{0\varepsilon^*}\; 0$

$(01)^* \, 0$

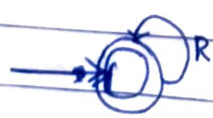
Ex: 

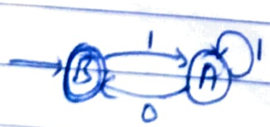
Eliminate the state ~~B~~ A since B itself is starting and final state.

R^* $\quad R = 10, \varepsilon$

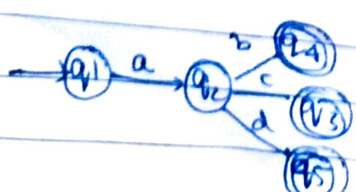
$(10+\varepsilon)^*$

$(10)^*$

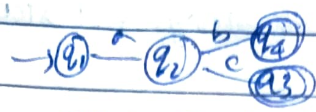
Standard form:  R^*

Ex:   $R^* = (11^*0+\varepsilon)^* = (11^*0)^*$

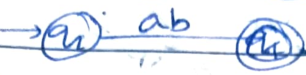
Ex:

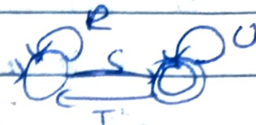


R1 = 

Eliminate q5 

Eliminate q3 

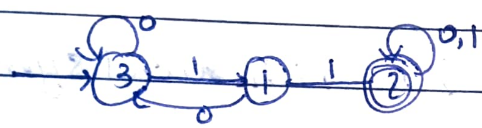
Eliminate q2 

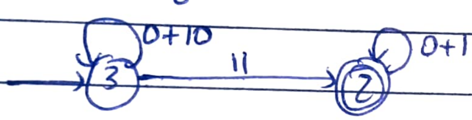


R = ε T = ∅
S = ab U = ε

R1 = $(R + SU^*T)^* SU^*$
$= \varepsilon + ((ab)^* \emptyset^*)\ ab.\varepsilon^*$
$= (\varepsilon + ab)^* ab$
$= ab\ ab$

Do the same for other final state
$R = R1 + R2 + R3 = ab + ac + ad$
$= a(b + c + d)$

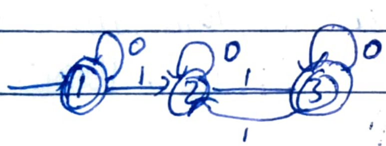
Ex: 

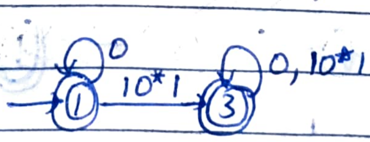


~~(0+10) + 11 (0+1)*~~
$(0+10)^*\ 11\ (0+1)^*$.

R = ε+0+10
S = 11
T = ∅
U = 0, 1, ε

$(R + SU^* T)^* SU^*$
$= [(\varepsilon + 0 + 10)^* + 11(0 + 1 + \varepsilon)^* \emptyset]^*\ 11(0+1+\varepsilon)^*$
$= (0+10)^*$

Ex: 

R1 =
Eliminate state 2 

~~R1~~ = Eliminate state 3