Q22. Construct a minimal DFA over {a,b} that accepts all strings having

(i) At Least 2 a's and At Least 3 b's

(ii) At Least 2 a's and At Most 3 b's

(iii) At Most 2 a's and At Most 3 b's

(iv) Exact 2 a's and Exact 3 b's

Q1: The no. of DFA that can be drawn over $\sum$ ={a, b} with 2 states q_0 and q_1 having q_0 as the initial state is ?

Q2: How many will accept $\sum^*$

Q3: How many will accept Φ

Q4: How many DFA neither accept Φ nor $\sum^*$.

Q5: The no. of DFA over {0,1} with two states is ?

Q6: The no. of DFA over {0,1} with three states is ?

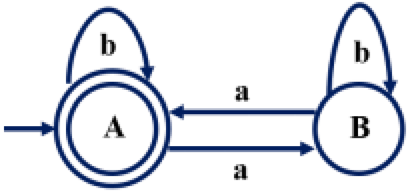
Formula count no. of DFA:

Operation on Finite Automata:

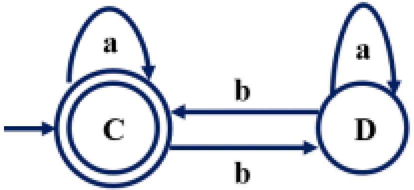
1. Union
2. Cross Product
3. Subtraction

(Theory of Computation)

Example: M_1 =



M_2 =



Find: $M_1 \times M_2$ = ?

$M_1 + M_2$ = ?

$M_1 - M_2$ = ?

NFA Designing: Simple Design as compared to DFA

Q1. Construct NFA accepting a set of strings over {a, b} in which each string of the language start with abb.

Q2. Construct NFA accepting a set of strings over {a, b} in which each string of the language ends with ‘abb’

Q3. Construction of NFA accepting a set of strings over {a, b} in which each string of the language containing 'abb' as the substring.

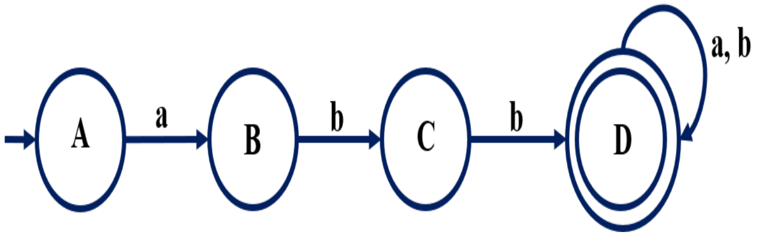
Q4. Design a NFA for 2nd symbol from LHS is a, over {a, b}

Q5. Design NFA for 2^{nd} symbol from RHS is a, over {a, b}

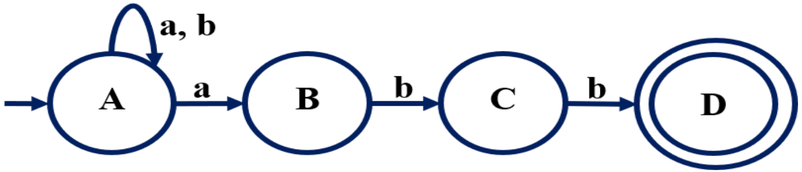
Q6. Design NFA for 3rd symbol from RHS is a, over {a, b}

Conversion from NFA to DFA:

Q1. Convert the following NFA to DFA



Q2. Convert the following NFA to DFA



Q3. Design NFA for 3rd symbol from RHS is ‘a’, over $\Sigma = \{a, b\}$ and convert to DFA

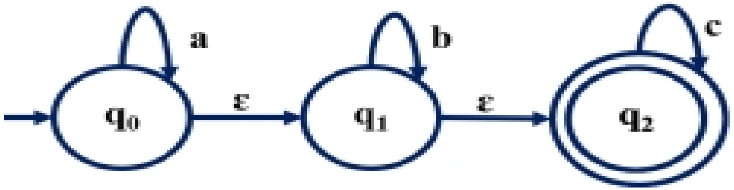
Note: If n^{th} symbol from RHS is a, over $\Sigma = \{a, b\}$. Then number of states in the corresponding DFA = 2^n

ε - NFA: NFA with ε - moves

Conversion ε - NFA to NFA (or) Removal of ε - Move:

1-Find out all the ε - transitions from each state from Q. That will be called as ε - closur(q_i), where $q_i \in Q$

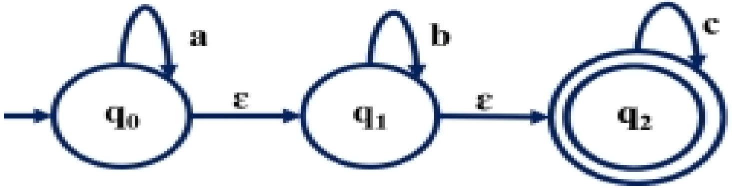
ε - closure (q_i): Set of all those states of the automata (NFA with ε - transition) which can be reached from q_i on a path labeled by ε i.e., without consuming any input symbol.



ε - closure (q_0) = {q_0, q_1, q_2}
ε - closure (q_1) = {q_1, q_2}
ε - closure (q_2) = {q_2}

2-Then δ' transition can be obtained. The δ' transition means a ε - closure on δ moves

$$\delta'(q_i, x) = \varepsilon\text{-closure}\,[\delta(\varepsilon\text{-closure}(q_i), x)]$$



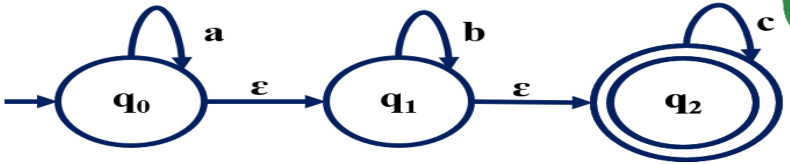
ε - closure $(q_0) = \{q_0, q_1, q_2\}$
ε - closure $(q_1) = \{q_1, q_2\}$
ε - closure $(q_2) = \{q_2\}$

3. Repeat step 2 for each input symbol and each state of given NFA

4. Final State:

$$F' = \begin{cases} F \cup \{q\}, & \text{if } \varepsilon\text{ - closure }(q) \text{ contains a state of } F \\ F & \text{otherwise} \end{cases}$$

Example: Convert the following NFA with ε to NFA without ε and DFA



Minimization of DFA: ε-NFA → NFA → DFA → Minimize DFA

DFA minimization stands for converting a given DFA to its equivalent DFA with minimum number of states for minimum DFA.

1. Reduce Unreachable State
2. Reduce Equivalent State

Equivalent States: p and q are equivalent ($p \approx q$) state

iff $\delta(p, x) \in F$ and $\delta(q, x) \in F$

(or)

$\delta(p, x) \notin F$ and $\delta(q, x) \notin F$

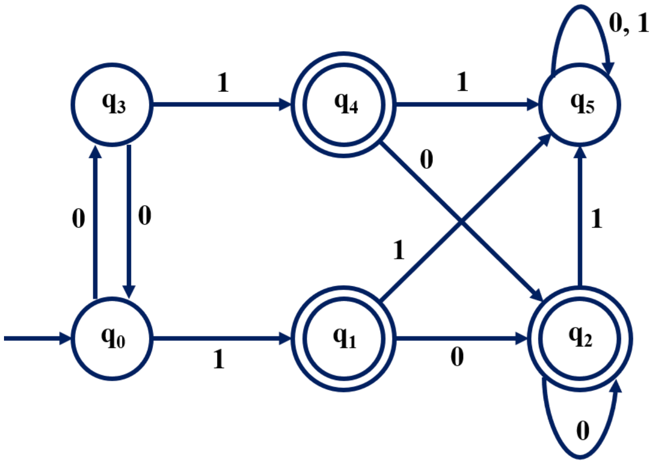
Two Method for Minimization of DFA:

1. Set Partition Method

2. Myhill Nerode Theorem

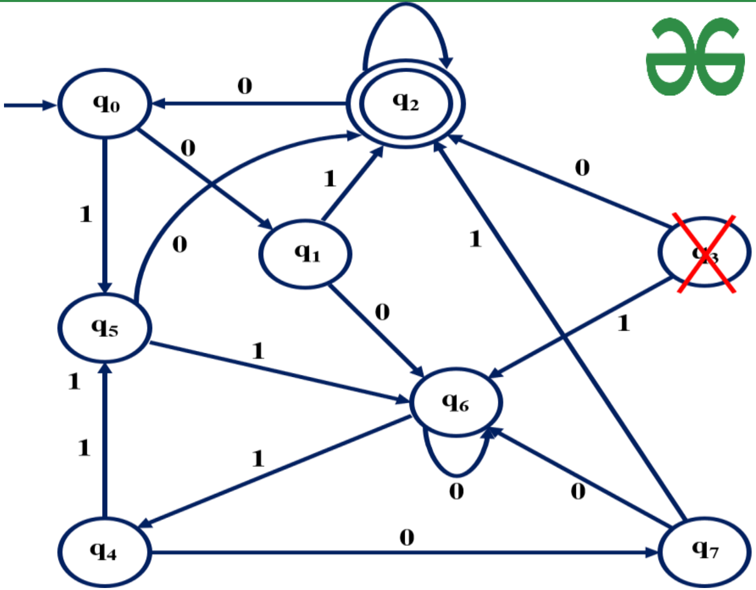
Set Partition Method:

Example: Minimize the following DFA



Example: Minimize the following DFA

Q/∑	0	1
$\rightarrow q_0$	q_1	q_5
q_1	q_6	q_2
$^*q_2$	q_0	q_2
q_4	q_7	q_5
q_5	q_2	q_6
q_6	q_6	q_4
q_7	q_6	q_2



(Theory of Computation)

Finite Automata with output:

Machine

Moore Machine	**Mealy Machine**
"Output depends only present state"	"Output depends on the present state and input"
Represented by 6 tuples	Represented by 6 tuples
$= (Q, \Sigma, \delta, q_0, \Delta, \lambda)$, where	$= (Q, \Sigma, \delta, q_0, \Delta, \lambda)$, where
Q : Finite set of states	Q : Finite set of states
Σ : Input alphabet	Σ : Input alphabet
δ : transition function $\delta : Q \times \Sigma \rightarrow Q$	δ : transition function $\delta : Q \times \Sigma \rightarrow Q$
q_0: initial state $q_0 \in Q$	q_0: initial state $q_0 \in Q$
Δ : Finite set of output	Δ : Finite set of output
λ : Output function $\boxed{\lambda : Q \rightarrow \Delta}$	λ : Output function $\boxed{\lambda : Q \times \Sigma \rightarrow \Delta}$

Representation of Moore and Mealy Machine:

Design Moore and Mealy Machine:

Q1: Design a mealy and moore m/c over {0,1} that produces output A if the no. of 1's in the input string is even otherwise produce output B

Q2: Construct a mealy and moore m/c that takes set of all strings over {0, 1} and produce 'A' as O/P if input ends with '10' or produce 'B' as O/P if input ends with '11' otherwise produces 'C'

Q1: Design mealy m/c for

(i) one's complement of binary no.

(ii) two's complement of binary no. (input read from LSB to MSB)

Q2: Design a mealy m/c which reads the input from $(0 + 1)^*$ and produces the following outputs.

(i) if input ends in 101, output is A

(ii) if input ends in 110, output is B

(iii) for other inputs, output is C

Conversion Moore to Mealy & Mealy to Moore m/c: