An NFA can have zero, one or more than one move from a given state on a given input symbol. An NFA can also have NULL moves (moves without input symbol). On the other hand, DFA has one and only one move from a given state on a given input symbol.

Conversion from NFA to DFA
Suppose there is an NFA N < Q, ∑, q0, δ, F > which recognizes a language L. Then the DFA D < Q', ∑, q0, δ', F' > can be constructed for language L as:
Step 1: Initially Q' = ϕ.
Step 2: Add q0 to Q'.
Step 3: For each state in Q', find the possible set of states for each input symbol using transition function of NFA. If this set of states is not in Q', add it to Q'.
Step 4: Final state of DFA will be all states with contain F (final states of NFA)
Example
Consider the following NFA shown in Figure 1.

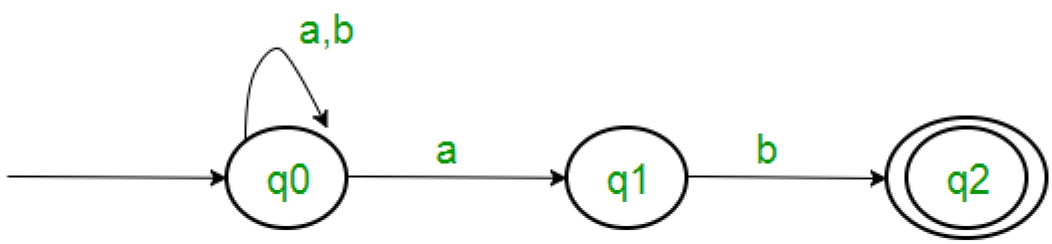


Figure 1

Following are the various parameters for NFA.
Q = { q0, q1, q2 }
∑ = (a, b)
F = { q2 }
δ (Transition Function of NFA)

State	a	b
q0	q0,q1	q0
q1		q2
q2		

Step 1: Q' = ϕ
Step 2: Q' = {q0}
Step 3: For each state in Q', find the states for each input symbol.

Currently, state in Q’ is q0, find moves from q0 on input symbol a and b using transition function of NFA and update the transition table of DFA.

δ’ (Transition Function of DFA)

State	a	b
q0	{q0,q1}	q0

Now { q0, q1 } will be considered as a single state. As its entry is not in Q’, add it to Q’.
So Q’ = { q0, { q0, q1 } }

Now, moves from state { q0, q1 } on different input symbols are not present in transition table of DFA, we will calculate it like:
δ’ ({ q0, q1 }, a) = δ (q0, a) ∪ δ (q1, a) = { q0, q1 }
δ’ ({ q0, q1 }, b) = δ (q0, b) ∪ δ (q1, b) = { q0, q2 }
Now we will update the transition table of DFA.

δ’ (Transition Function of DFA)

State	a	B
q0	{q0,q1}	q0
{q0,q1}	{q0,q1}	{q0,q2}

Now { q0, q2 } will be considered as a single state. As its entry is not in Q’, add it to Q’.
So Q’ = { q0, { q0, q1 }, { q0, q2 } }

Now, moves from state {q0, q2} on different input symbols are not present in transition table of DFA, we will calculate it like:
δ’ ({ q0, q2 }, a) = δ (q0, a) ∪ δ (q2, a) = { q0, q1 }
δ’ ({ q0, q2 }, b) = δ (q0, b) ∪ δ (q2, b) = { q0 }
Now we will update the transition table of DFA.

δ' (Transition Function of DFA)

State	a	B
q0	{q0,q1}	q0
{q0,q1}	{q0,q1}	{q0,q2}
{q0,q2}	{q0,q1}	q0

As there is no new state generated, we are done with the conversion. Final state of DFA will be state which has q2 as its component i.e., { q0, q2 }

Following are the various parameters for DFA.
Q' = { q0, { q0, q1 }, { q0, q2 } }
∑ = (a, b)
F = { { q0, q2 } } and transition function δ' as shown above. The final DFA for above NFA has been shown in Figure 2.

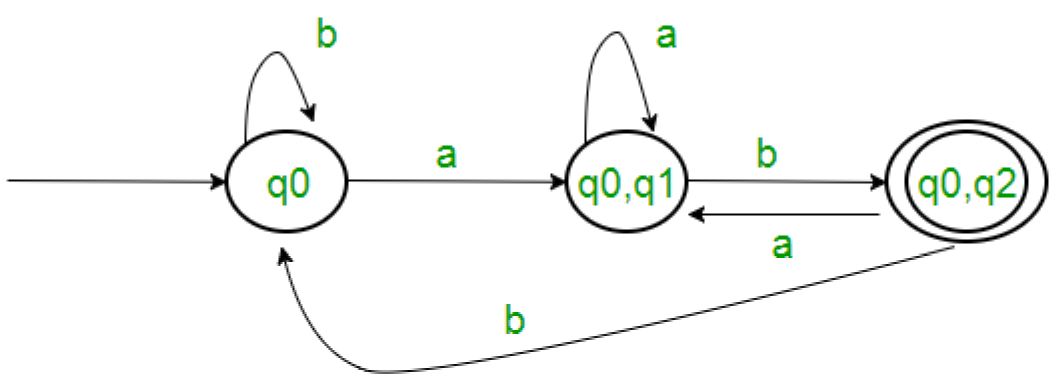
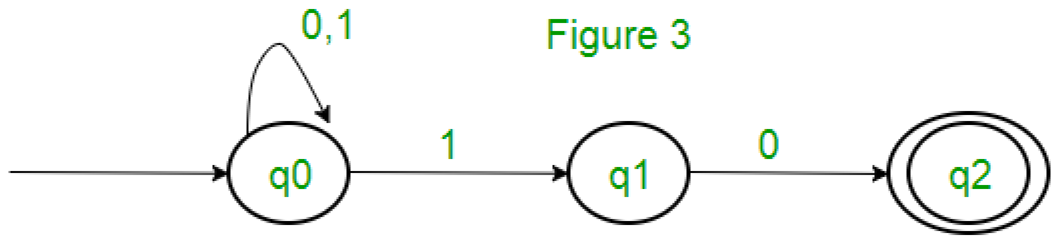


Figure 2

Note : Sometimes, it is not easy to convert regular expression to DFA. First you can convert regular expression to NFA and then NFA to DFA.
Question : The number of states in the minimal deterministic finite automaton corresponding to the regular expression (0 + 1)* (10) is ____________.
Solution : First, we will make an NFA for the above expression. To make an NFA for (0 + 1)*, NFA will be in same state q0 on input symbol 0 or 1. Then for concatenation, we will add two moves (q0 to q1 for 1 and q1 to q2 for 0) as shown in Figure 3.

Figure 3



Using above algorithm, we can convert NFA to DFA as shown in Figure 4

State	**0**	**1**
q0	q0	q0,q1
q0,q1	q0,q2	q0,q1
q0,q2	q0	q0,q1

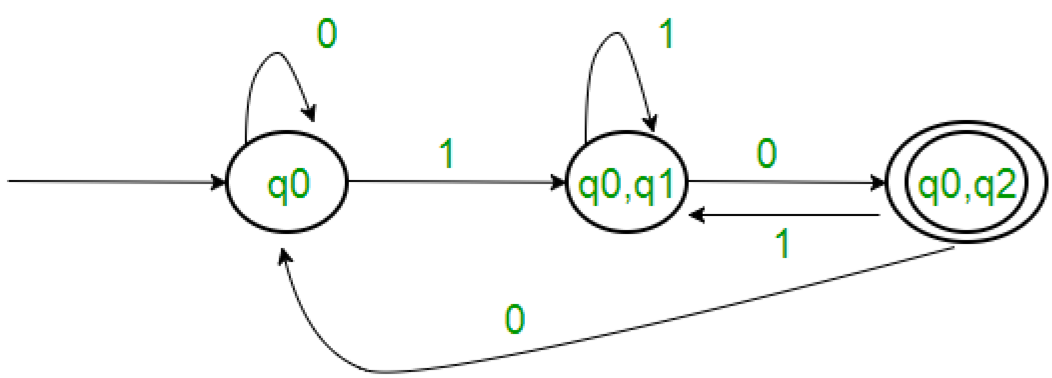


Figure 4

DFA minimization stands for converting a given DFA to its equivalent DFA with minimum number of states. DFA minimization is also called as Optimization of DFA and uses partitioning algorithm.

Minimization of DFA

Suppose there is a DFA D < Q, Σ, q0, δ, F > which recognizes a language L. Then the minimized DFA D < Q', Σ, q0, δ', F' > can be constructed for language L as:

Step 1: We will divide Q (set of states) into two sets. One set will contain all final states and other set will contain non-final states. This partition is called P_0.

Step 2: Initialize k = 1

Step 3: Find P_k by partitioning the different sets of P_{k-1}. In each set of P_{k-1}, we will take all possible pair of states. If two states of a set are distinguishable, we will split the sets into different sets in P_k.

Step 4: Stop when $P_k = P_{k-1}$ (No change in partition)

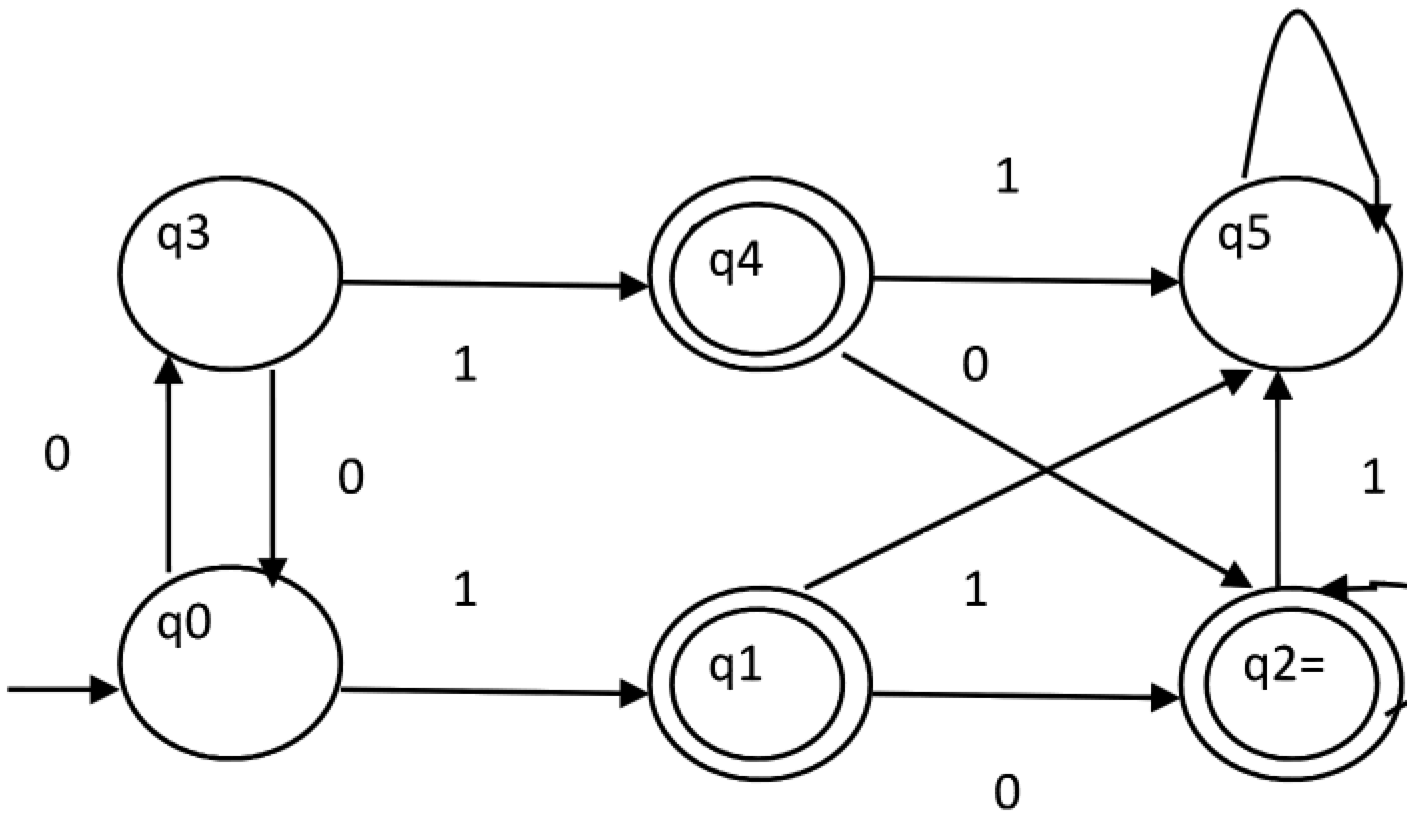
Step 5: All states of one set are merged into one. No. of states in minimized DFA will be equal to no. of sets in P_k.

How to find whether two states in partition P_k are distinguishable ?

Two states (qi, qj) are distinguishable in partition P_k if for any input symbol a, δ (qi, a) and δ (qj, a) are in different sets in partition P_{k-1}.

Example

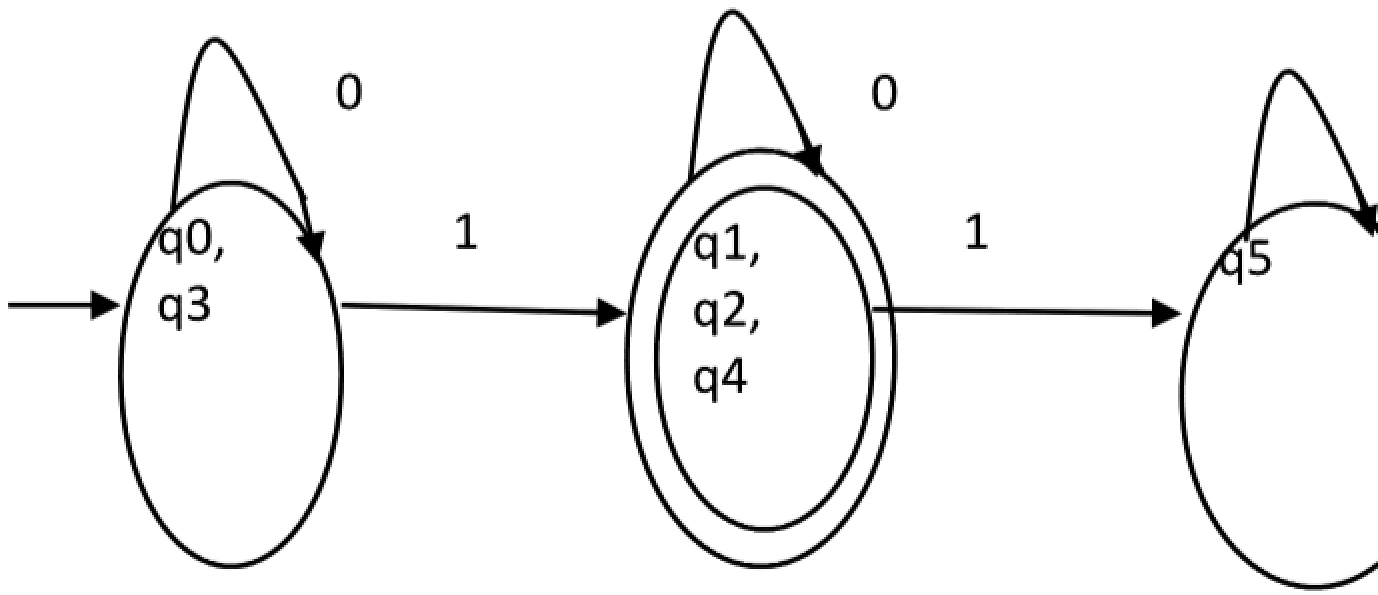
Consider the following DFA shown in figure.



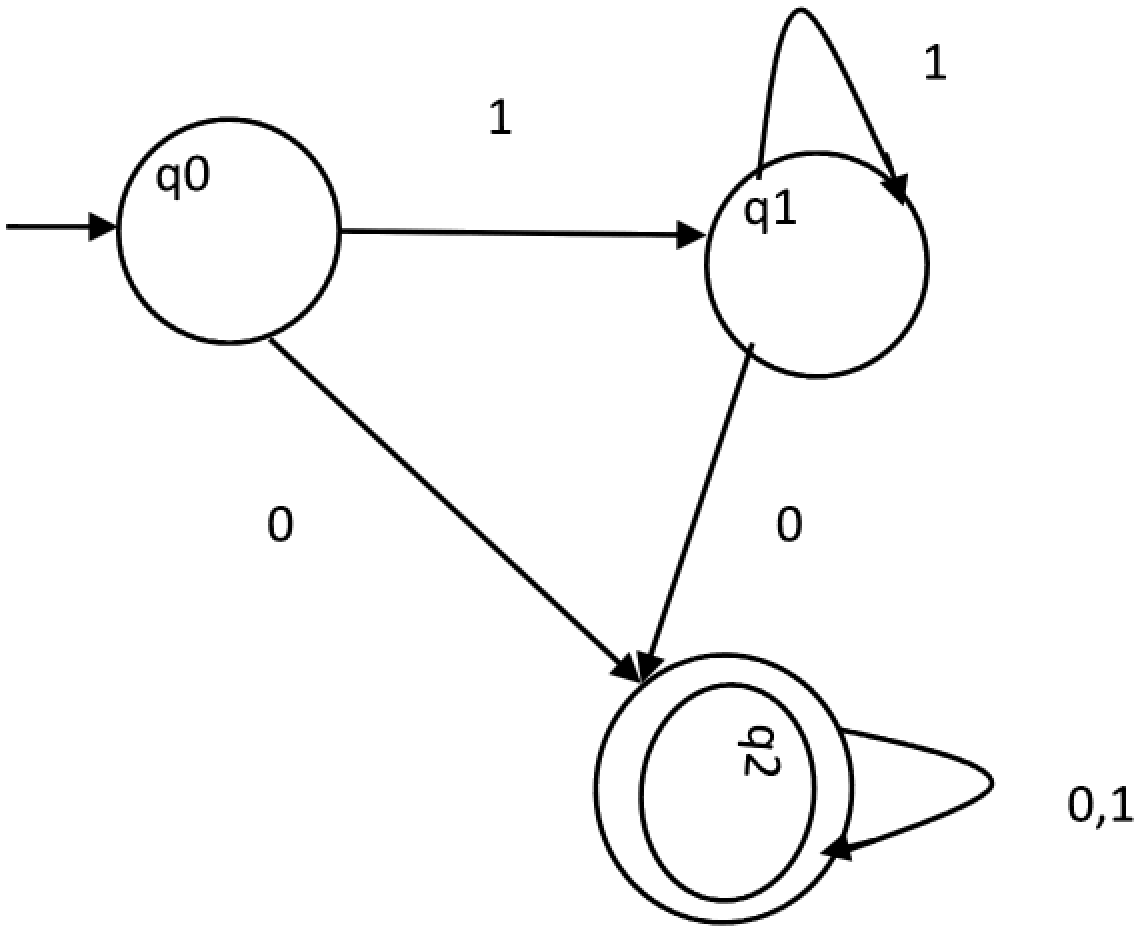
Step 1. P0 will have two sets of states. One set will contain q1, q2, q4 which are final states of DFA and another set will contain remaining states. So P0 = { { q1, q2, q4 }, { q0, q3, q5 } }.
Step 2. To calculate P1, we will check whether sets of partition P0 can be partitioned or not:
i) For set { q1, q2, q4 } :
δ (q1, 0) = δ (q2, 0) = q2 and δ (q1, 1) = δ (q2, 1) = q5, So q1 and q2 are not distinguishable.
Similarly, δ (q1, 0) = δ (q4, 0) = q2 and δ (q1, 1) = δ (q4, 1) = q5, So q1 and q4 are not distinguishable.
Since, q1 and q2 are not distinguishable and q1 and q4 are also not distinguishable, So q2 and q4 are not distinguishable. So, { q1, q2, q4 } set will not be partitioned in P1.
ii) For set { q0, q3, q5 } :
δ (q0, 0) = q3 and δ (q3, 0) = q0
δ (q0, 1) = q1 and δ(q3, 1) = q4
Moves of q0 and q3 on input symbol 0 are q3 and q0 respectively which are in same set in partition P0. Similarly, Moves of q0 and q3 on input symbol 1 are q1 and q4 which are in same set in partition P0. So, q0 and q3 are not distinguishable.

δ (q0, 0) = q3 and δ (q5, 0) = q5 and δ (q0, 1) = q1 and δ (q5, 1) = q5
Moves of q0 and q5 on input symbol 1 are q1 and q5 respectively which are in different set in partition P0. So, q0 and q5 are distinguishable. So, set { q0, q3, q5 } will be partitioned into { q0, q3 } and { q5 }. So,
P1 = { { q1, q2, q4 }, { q0, q3}, { q5 } }

To calculate P2, we will check whether sets of partition P1 can be partitioned or not:
iii)For set { q1, q2, q4 } :
δ (q1, 0) = δ (q2, 0) = q2 and δ (q1, 1) = δ (q2, 1) = q5, So q1 and q2 are not distinguishable.
Similarly, δ (q1, 0) = δ (q4, 0) = q2 and δ (q1, 1) = δ (q4, 1) = q5, So q1 and q4 are not distinguishable.
Since, q1 and q2 are not distinguishable and q1 and q4 are also not distinguishable, So q2 and q4 are not distinguishable. So, { q1, q2, q4 } set will not be partitioned in P2.
iv)For set { q0, q3 } :
δ (q0, 0) = q3 and δ (q3, 0) = q0
δ (q0, 1) = q1 and δ (q3, 1) = q4
Moves of q0 and q3 on input symbol 0 are q3 and q0 respectively which are in same set in partition P1. Similarly, Moves of q0 and q3 on input symbol 1 are q1 and q4 which are in same set in partition P1. So, q0 and q3 are not distinguishable.
v) For set { q5 }:
Since we have only one state in this set, it can’t be further partitioned. So,
P2 = { { q1, q2, q4 }, { q0, q3 }, { q5 } }
Since, P1=P2. So, this is the final partition. Partition P2 means that q1, q2 and q4 states are merged into one. Similarly, q0 and q3 are merged into one.
Minimized DFA corresponding to DFA of Figure 1 is shown in Figure 2 as:



Question : Consider the given DFA. Which of the following is false?
1. Complement of L(A) is context-free.
2. L(A) = L ((11 * 0 + 0) (0 + 1)* 0* 1*)
3. For the language accepted by A, A is the minimal DFA.
4. A accepts all strings over { 0, 1 } of length atleast two.



A. 1 and 3 only
B. 2 and 4 only
C. 2 and 3 only
D. 3 and 4 only

Solution : Statement 4 says, it will accept all strings of length atleast 2. But it accepts 0 which is of length 1. So, 4 is false.
Statement 3 says that the DFA is minimal. We will check using the algorithm discussed above.
P0 = { { q2 }, { q0, q1 } }
P1 = { q2 }, { q0, q1 } }. Since, P0 = P1, P1 is the final DFA. q0 and q1 can be merged. So minimal DFA will have two states. Therefore, statement 3 is also false.
So correct option is (D).