

NFA (Non-Deterministic finite automata)

- NFA stands for non-deterministic finite automata. It is easy to construct an NFA than DFA for a given regular language.
- The finite automata are called NFA when there exist many paths for specific input from the current state to the next state.
- Every NFA is not DFA, but each NFA can be translated into DFA.
- NFA is defined in the same way as DFA but with the following two exceptions, it contains multiple next states, and it contains ϵ transition.

In the following image, we can see that from state q_0 for input a , there are two next states q_1 and q_2 , similarly, from q_0 for input b , the next states are q_0 and q_1 . Thus it is not fixed or determined that with a particular input where to go next. Hence this FA is called non-deterministic finite automata.

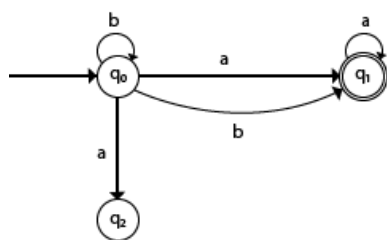


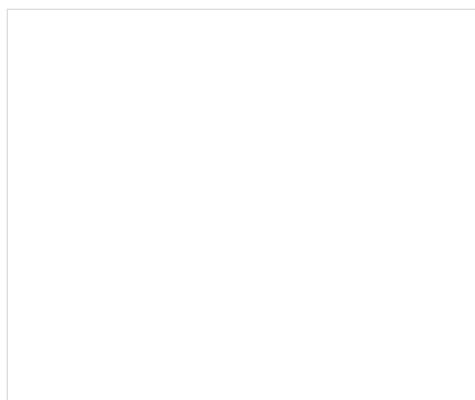
Fig:- NDFA

Formal definition of NFA:

NFA also has five states same as DFA, but with different transition function, as shown follows:

$$\delta: Q \times \Sigma \rightarrow 2^Q$$

where,



Q : finite set of states
 Σ : finite set of the input symbol
 q_0 : initial state
 F : **final** state
 δ : Transition function

Graphical Representation of an NFA

An NFA can be represented by digraphs called state diagram. In which:

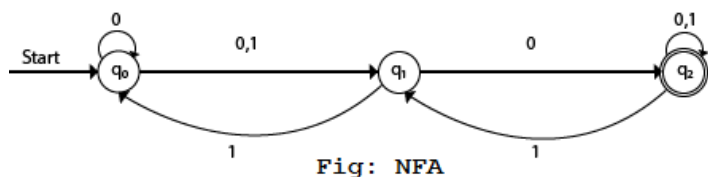
1. The state is represented by vertices.
2. The arc labeled with an input character show the transitions.
3. The initial state is marked with an arrow.
4. The final state is denoted by the double circle.

Example 1:

$Q = \{q_0, q_1, q_2\}$
 $\Sigma = \{0, 1\}$
 $q_0 = \{q_0\}$
 $F = \{q_2\}$

Solution:

Transition diagram:



Transition Table:

Present State	Next state for Input 0	Next State of Input 1
→q0	q0, q1	q1
q1	q2	q0
*q2	q2	q1, q2

In the above diagram, we can see that when the current state is q_0 , on input 0, the next state will be q_0 or q_1 , and on 1 input the next state will be q_1 . When the current state is q_1 , on input 0 the next state will be q_2 and on 1 input, the next state will be q_0 . When the current state is q_2 , on 0 input the next state is q_2 , and on 1 input the next state will be q_1 or q_2 .

Example 2:

NFA with $\Sigma = \{0, 1\}$ accepts all strings with 01.

Solution:

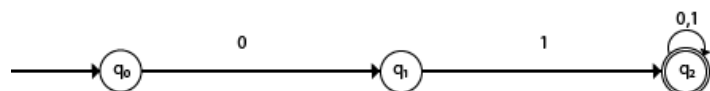


Fig: NFA

Transition Table:

Present State	Next state for Input 0	Next State of Input 1
→ q_0	q_1	ϵ
q_1	ϵ	q_2
* q_2	q_2	q_2

Example 3:

NFA with $\Sigma = \{0, 1\}$ and accept all string of length atleast 2.

Solution:

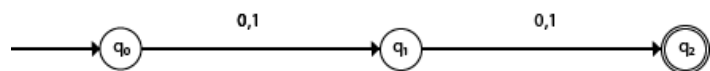


Fig: NFA


Transition Table:

Present State	Next state for Input 0	Next State of Input 1
→ q_0	q_1	q_1
q_1	q_2	q_2
* q_2	ϵ	ϵ




Help Others, Please Share



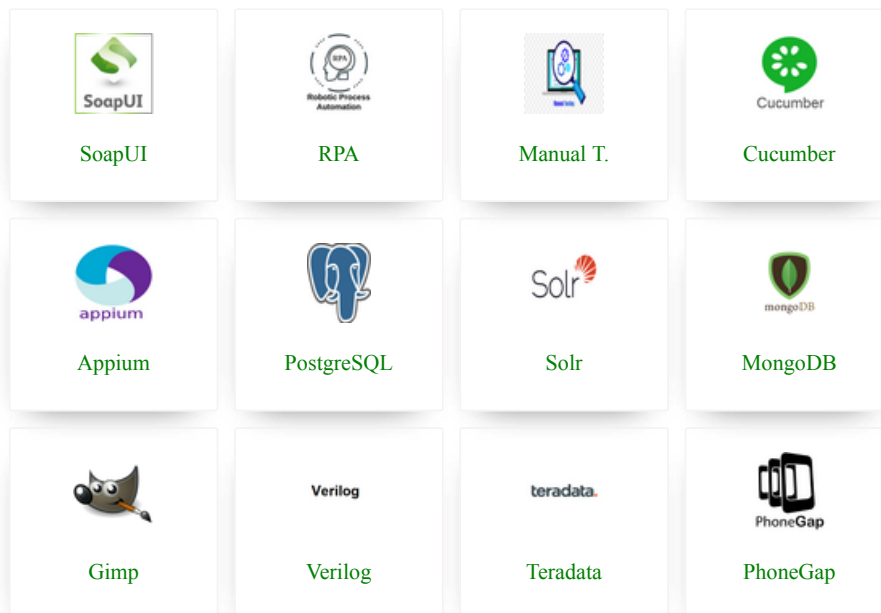
TestProject

Test in Under 15 Minutes

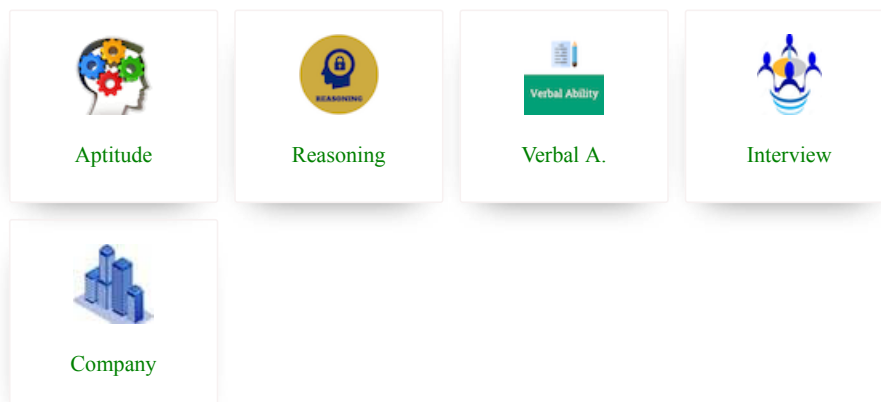
Join the #1 test automation community trusted by over 1 million happy users.



Learn Latest Tutorials



Preparation



Trending Technologies

