# NONDETERMINISTIC FINITE AUTOMATA

iACADEMY
SCHOOL OF COMPUTING • SCHOOL OF BUSINESS • SCHOOL OF DESIGN

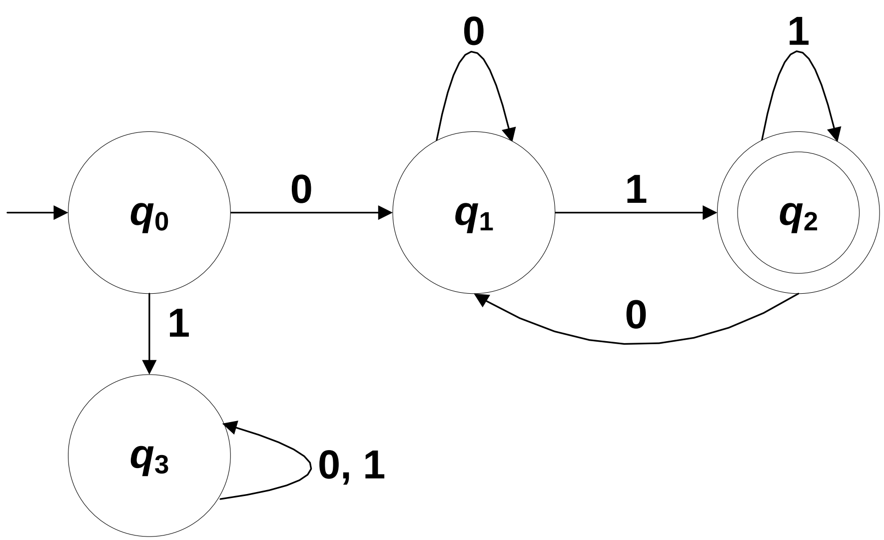


MITCH M. ANDAYA

# NONDETERMINISM

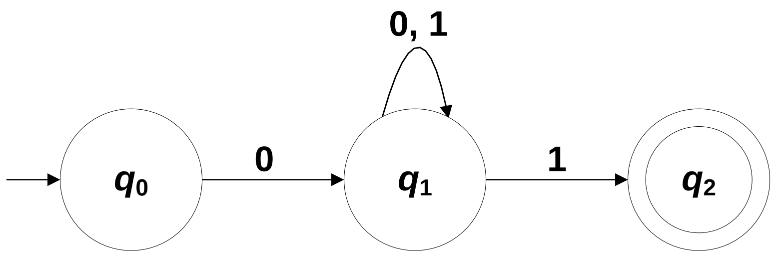
- The main characteristic of a DFA is that for each input symbol, the automaton can move to only one state from any given current state.

- Consider a DFA that accepts strings over the alphabet $\Sigma = \{0, 1\}$ that starts with a 0 and ends with a 1.



# NONDETERMINISM

- Consider now the following finite automaton:

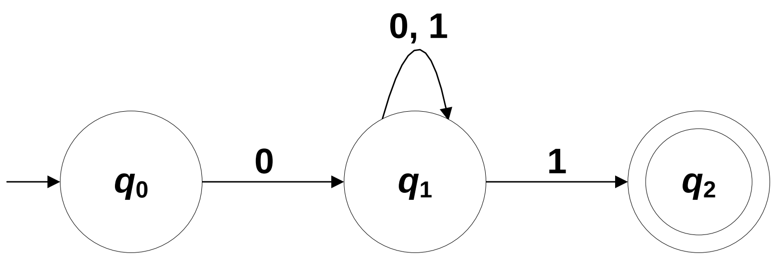



This machine also accepts strings that start with a 0 and end with 1.

Observe that if the machine is at state $q_1$ and a 1 arrives, the machine goes to state $q_2$ and stays at $q_1$ at the same time.

This machine is not a DFA because there is a situation in which an input symbol causes the automaton to move to more than one state.

# NONDETERMINISM



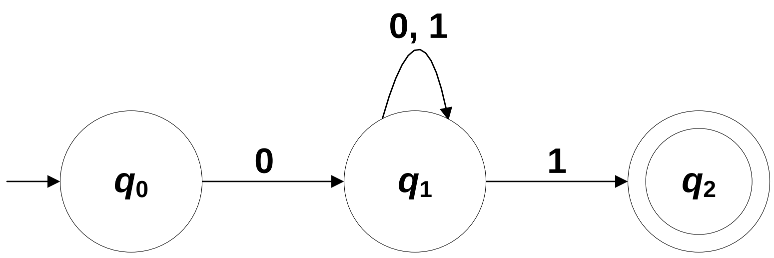

In this case, the NFA will consider both possibilities.

The NFA will make a "copy" of itself. One copy continues the computation at state $q_1$ while the other continues at state $q_2$.

Effectively, the NFA is said to be in two states, $q_1$ and $q_2$.

This is in contrast with a DFA since DFAs can only be in one state at any given time.

# NONDETERMINISM



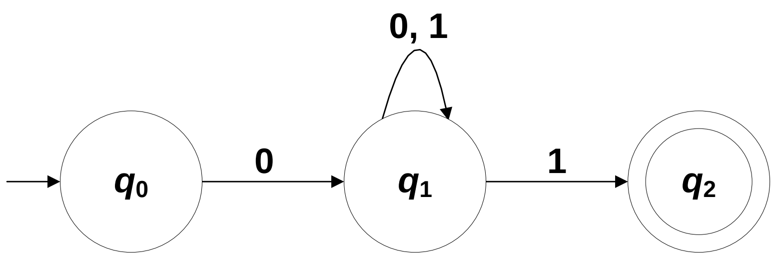

- The given graph is the state diagram for a ***nondeterministic finite automaton*** (NFA).

- Because an NFA can move to more than one state given a single input symbol, an NFA is said to have "choices."

- Take the given NFA as an example.  If the NFA is currently in state $q_1$ and an input symbol 1 arrives, the NFA has the option of staying in state $q_1$ or going to state $q_2$.

# NONDETERMINISM

- Notice also that in an NFA, it is possible for a state not to have a transition edge for one or more input symbols.



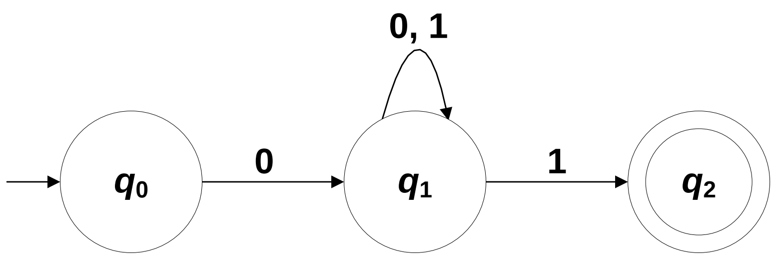

In the given example, take note that for state $q_o$, there is a transition edge for input symbol 0 but none for 1.

If a copy of the NFA is at state $q_o$ and an input symbol 1 arrives, this copy stops processing because it has nowhere to go. This copy of the NFA simply "dies." Similarly, if a copy of the NFA is at $q_2$ and a 1 or a 0 arrives, this copy also stops processing and dies.

# NONDETERMINISM

- Here is an example to illustrate how an NFA processes strings:

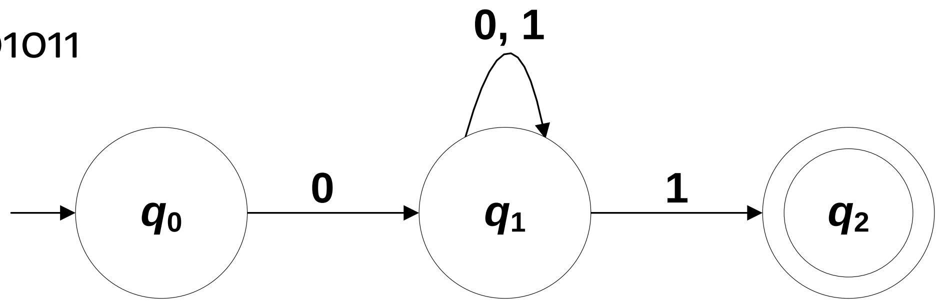
  Given the following NFA:




Determine if the string 01011 will be accepted.

# NONDETERMINISM

01011



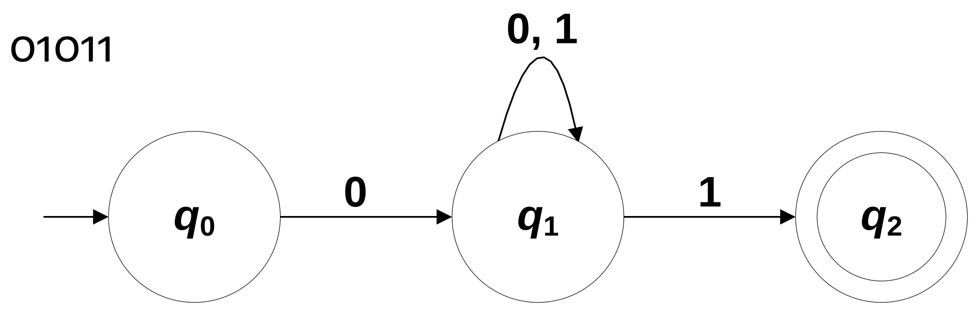

| input | state |
|---|---|
| | $q_0$ (start) |
| 0 | ↓ |
| | $q_1$ |

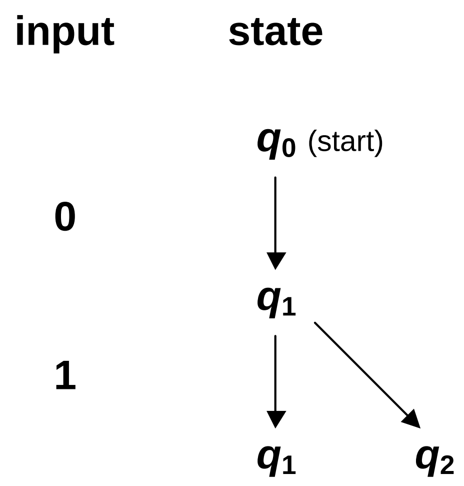
1. Initially, the NFA will be at state $q_o$.

   Assume the first input symbol (0) arrives.

   So the NFA goes to state $q_1$.

# NONDETERMINISM




2. The second input symbol (1) arrives.

   The NFA has two options, either it stays at $q_1$ or goes to $q_2$.

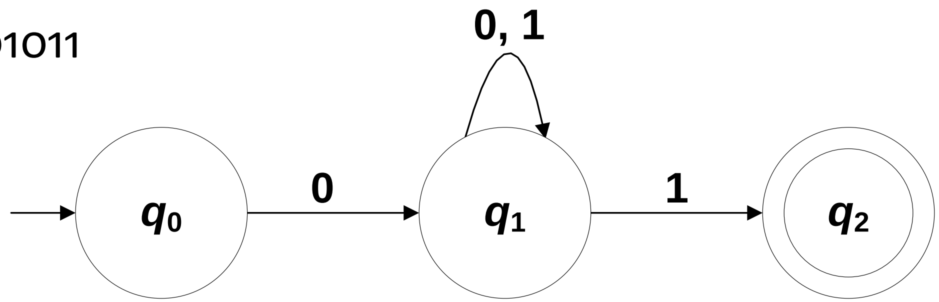
   The machine will try both possibilities at the same time.

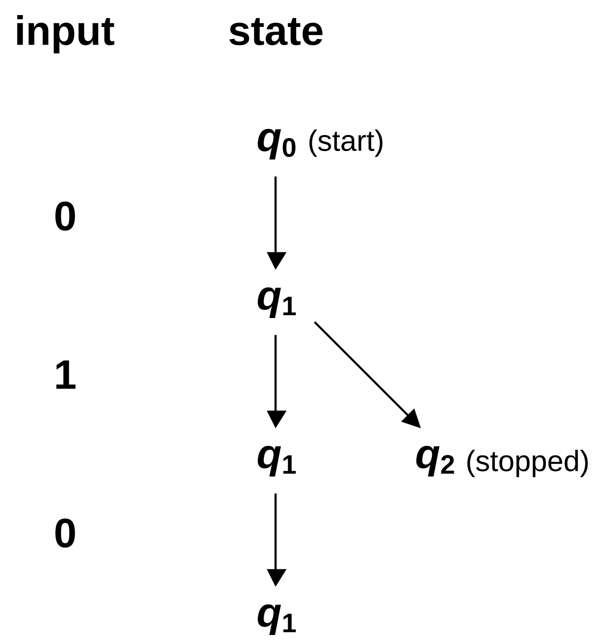
The machine made another copy of itself.

One copy to continue the processing at $q_1$ and the other at $q_2$.

# NONDETERMINISM

01011




3. Assume now that the third input symbol (0) arrives.

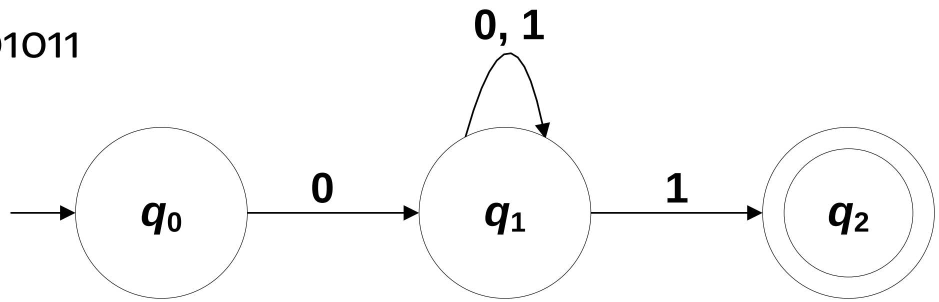
   One copy of the NFA (the one at $q_1$) stays at $q_1$.

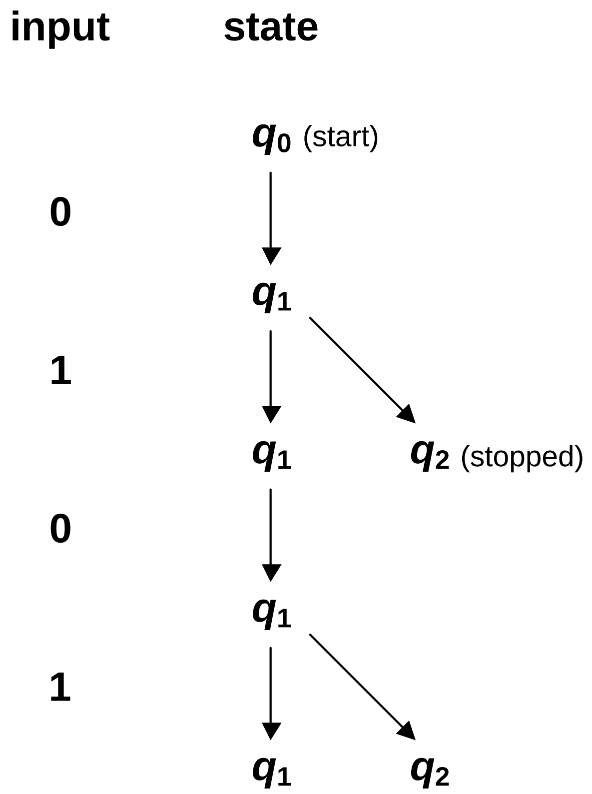
   The other copy (the one at $q_2$) dies because it has nowhere to go.

At this point, there will only be one remaining copy of the NFA (the one that is still at $q_1$).

# NONDETERMINISM

01011



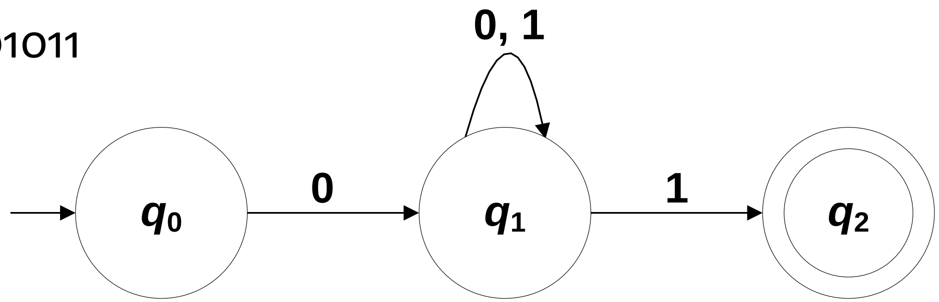

4. Assume now that the fourth input symbol (1) arrives.

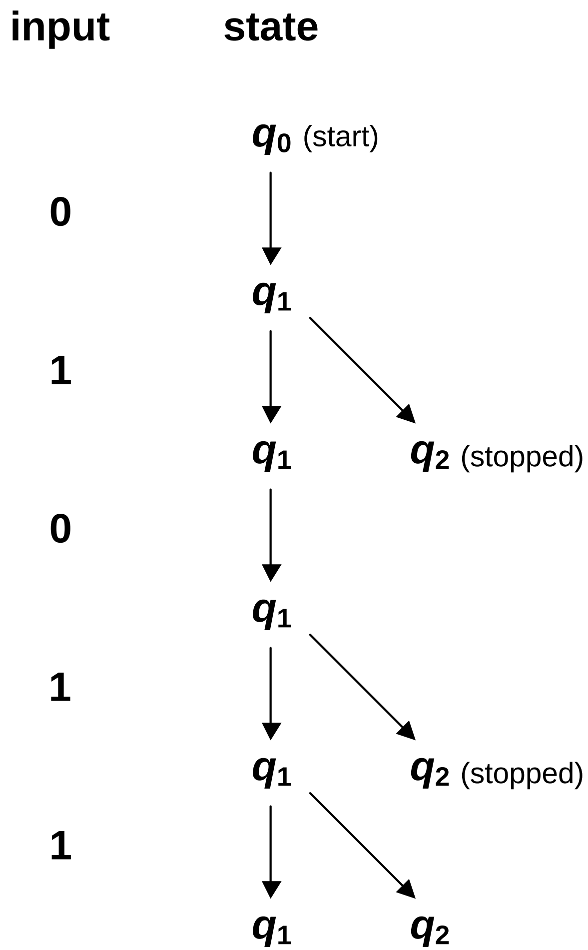
   There are again two options, either stay at $q_1$ or go to $q_2$.

   The NFA will again consider both possibilities.

# NONDETERMINISM

01011



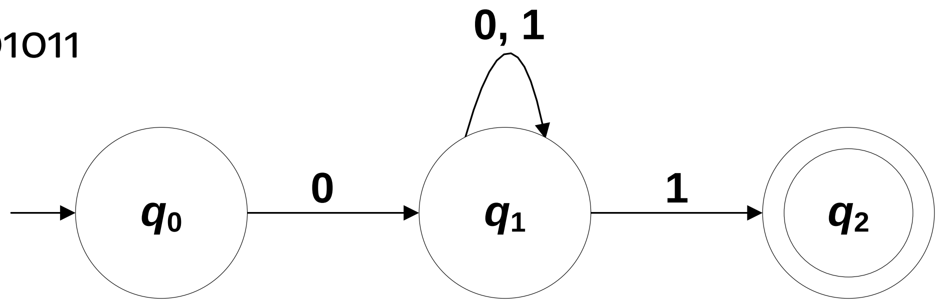

5. Assume now that the fifth and last input symbol (1) arrives.

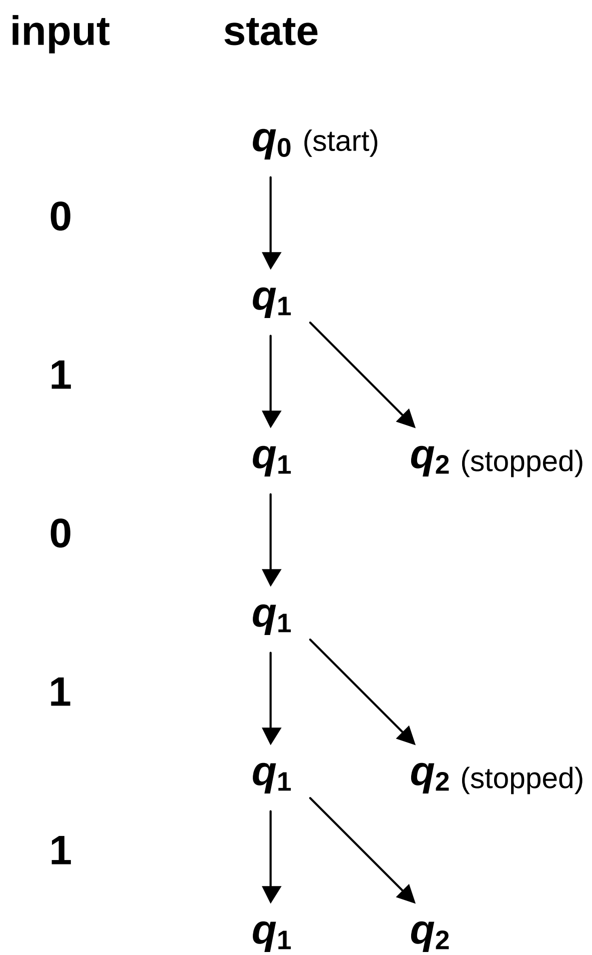
   One copy of the NFA (the one at $q_1$) has the option of staying at $q_1$ or going to $q_2$. As usual, the NFA will consider both.

   The other copy (the one at $q_2$) stops processing.

# NONDETERMINISM

01011




There are two copies of the NFA.

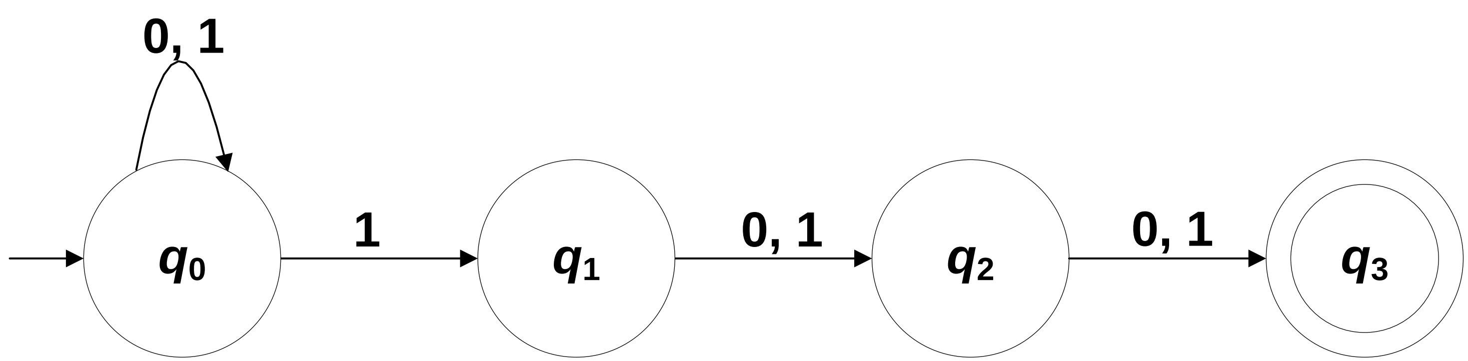
One copy is currently at state $q_1$ and the other at state $q_2$. Since one of the copies is at a final state, the NFA accepts the string 01011.

If there is no copy of the NFA that is in a final state, then the string will be rejected.

# NONDETERMINISM

- Another example:

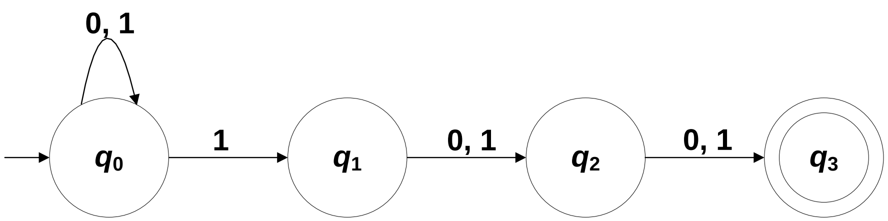
  Given the following NFA:




Determine if the string 110110 will be accepted.

# NONDETERMINISM

110110





1. Initially, the NFA will be at state $q_o$.

   Assume that the first input symbol (1) arrives.

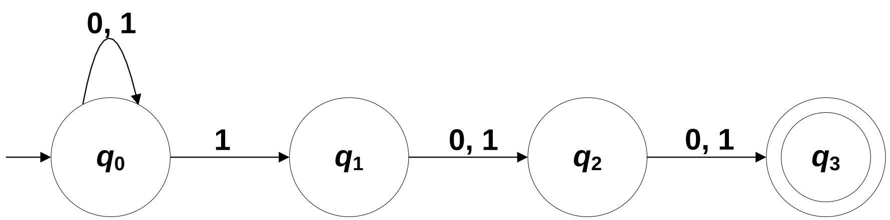
   The NFA has two options, either to stay at $q_o$ or go to $q_1$. The machine will try both possibilities.

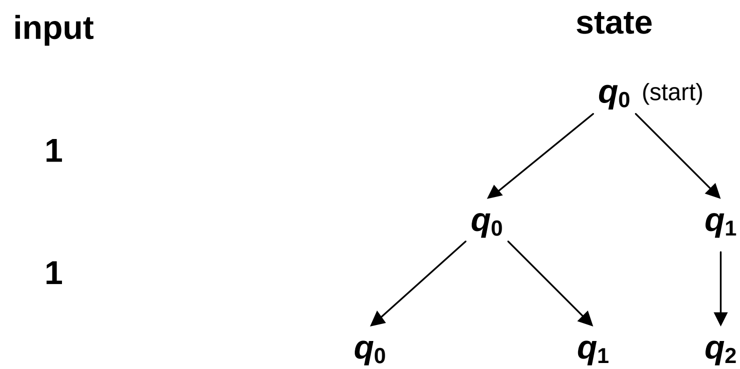
There will now be two copies of the NFA.

One copy is at state $q_o$ while the other is at state $q_1$.

# NONDETERMINISM

110110




2. Assume the second input symbol (1) arrives

   The copy of the NFA which is at state $q_0$ has two options— stay at $q_0$ or go to $q_1$. The machine will try both possibilities.

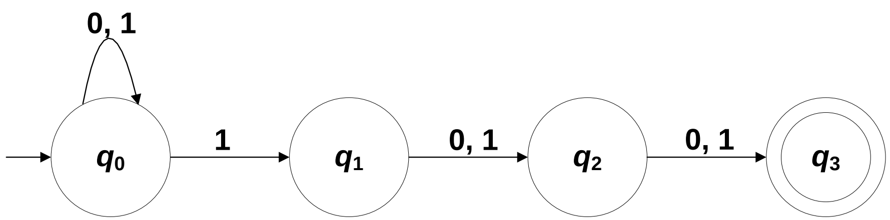
   The copy which is at state $q_1$ will go to $q_2$.

There will now be three copies of the NFA.

One is at state $q_0$, the second one is at state $q_1$, and the third is at state $q_2$.

# NONDETERMINISM

110110

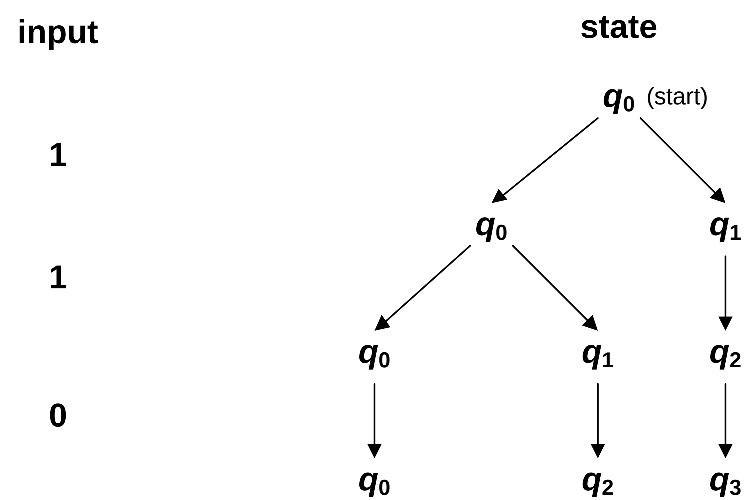



3. Assume the third input symbol (0) arrives.

   The copy of the NFA which is at state $q_0$ stays at $q_0$.

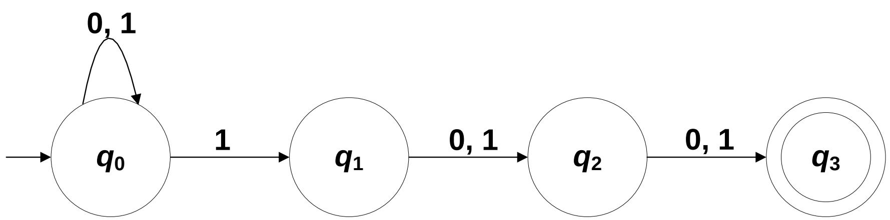
   The copy which is at state $q_1$ goes to $q_2$.

   And the copy which is at $q_2$ goes to state $q_3$.



# NONDETERMINISM

110110

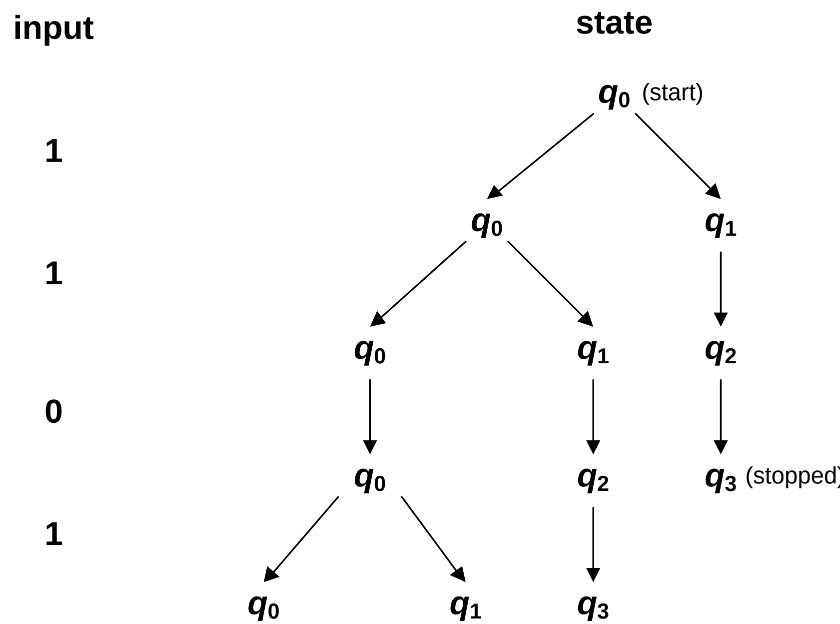



4. Assume the fourth input symbol (1) arrives.

   The copy of the NFA which is at state $q_0$ has two options. Stay at $q_0$ or go to $q_1$. The machine will try both possibilities.

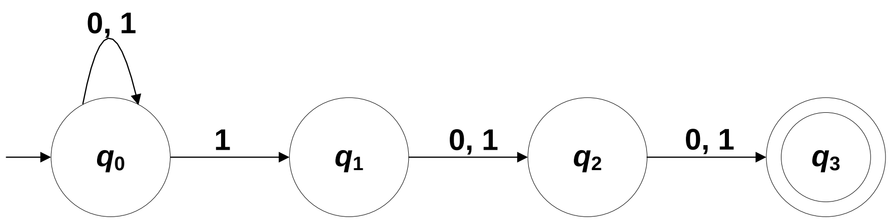
   The copy which is at state $q_2$ will go to $q_3$.

   The copy which is at state $q_3$ stops computing because it has nowhere to go.



# NONDETERMINISM

110110

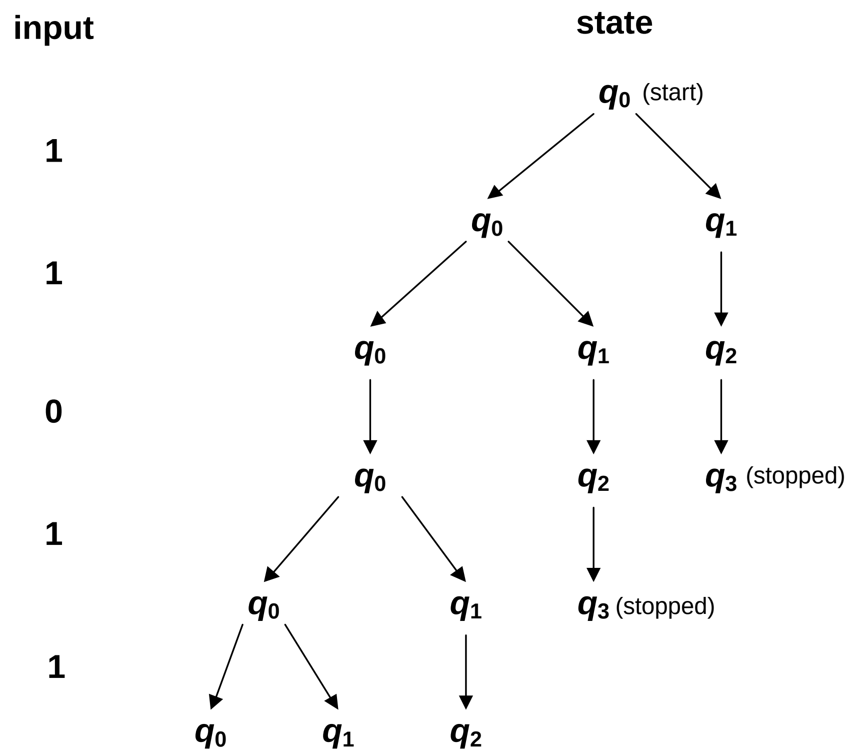



5. Assume the fifth input symbol (1) arrives.

   The copy of the NFA which is at state $q_0$ has two options—stay at $q_0$ or go to $q_1$. The machine will try both possibilities.

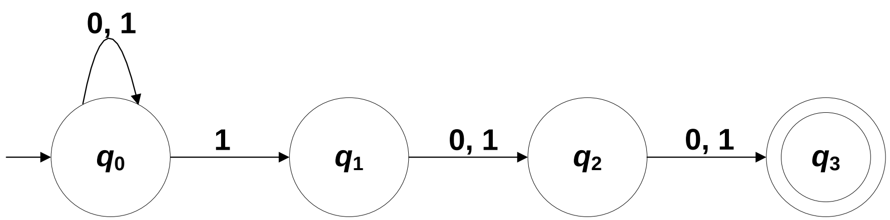
   The copy which is at state $q_1$ will go to $q_2$.

   The copy which is at $q_3$ dies.



# NONDETERMINISM

110110

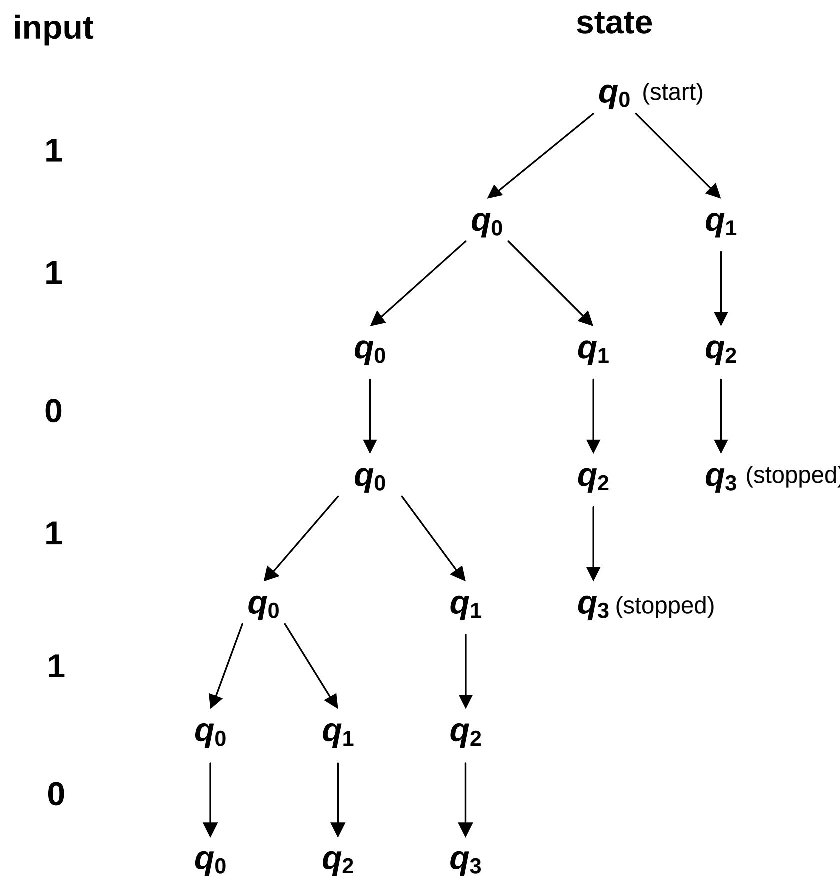



6. Assume the sixth and final input symbol (0) arrives.

   The copy of the NFA which is at state $q_0$ stays at $q_0$.

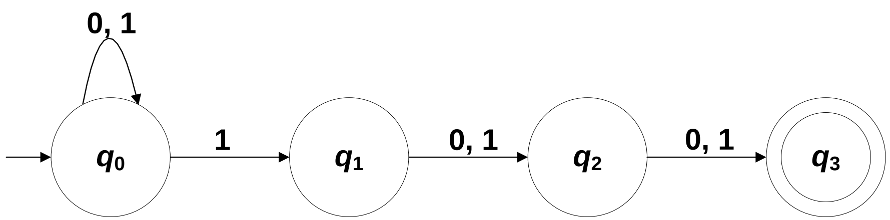
   The copy at state $q_1$ goes to $q_2$.

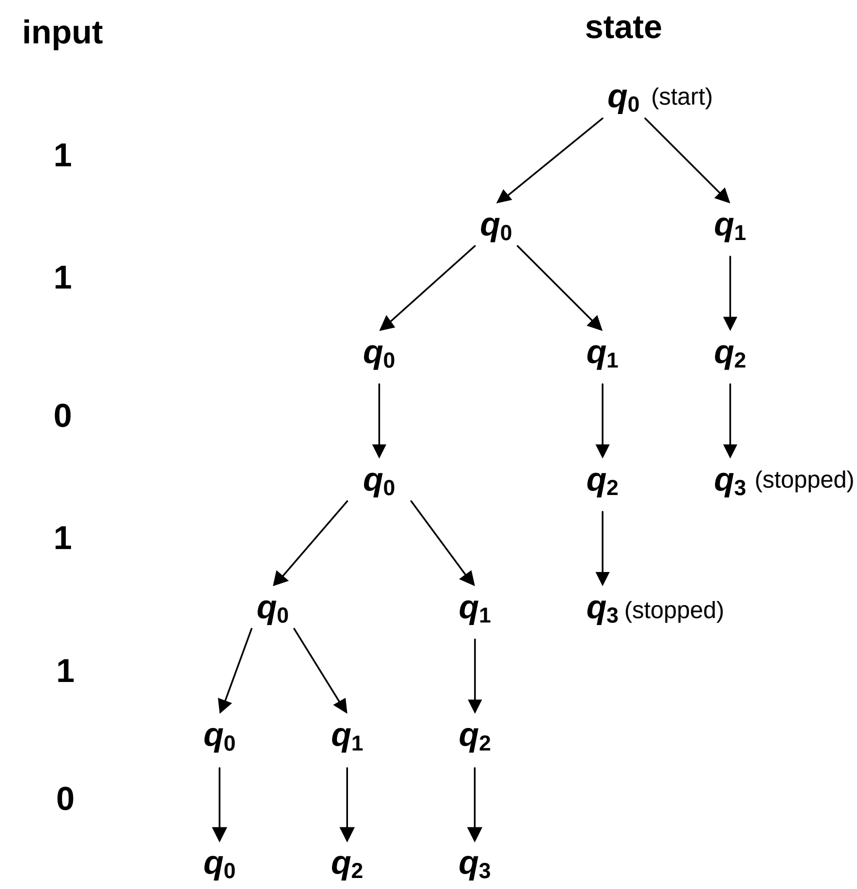
   The copy at state $q_2$ goes to $q_3$.



# NONDETERMINISM

110110



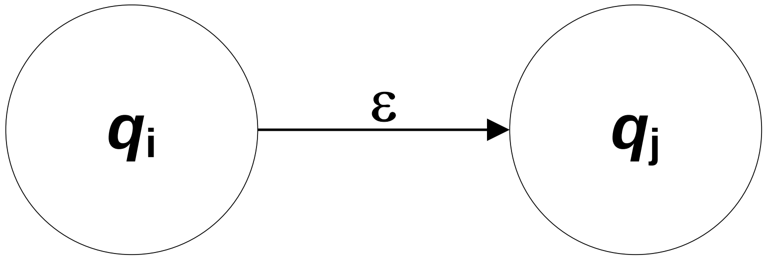




The NFA is now at three states, $q_0$, $q_2$, and $q_3$.

Since one of these states is a final state ($q_3$), then the NFA accepts the input string 110110.

# ε - TRANSITIONS

- One other difference between an NFA and a DFA is that an NFA may have ε-transitions. An ε-transition is shown below:

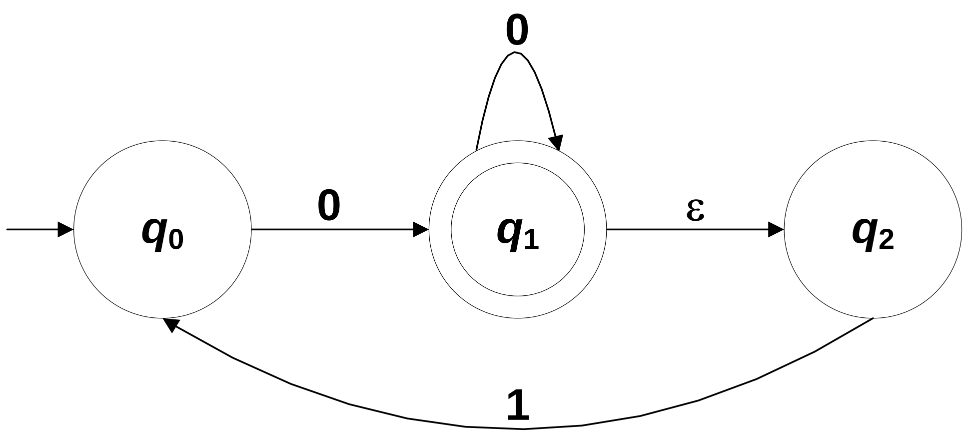



Whenever there is an ε-transition from state $q_i$ to state $q_j$, this means that once an NFA goes to state $q_i$, it has the option of staying at $q_i$ or it can go right away to state $q_j$ even without any additional input arriving.

# ε - TRANSITIONS

- Example:

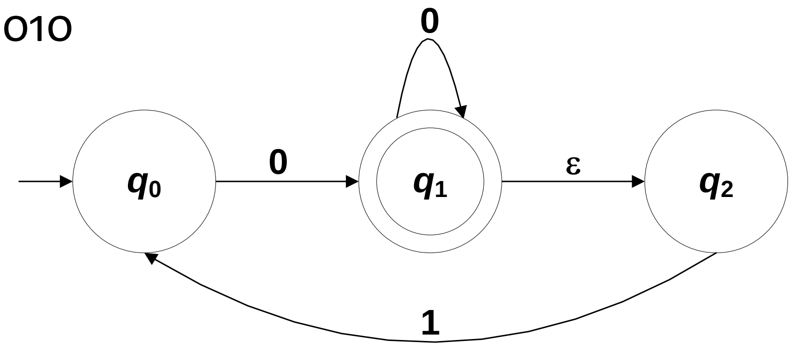
  Given the following NFA:

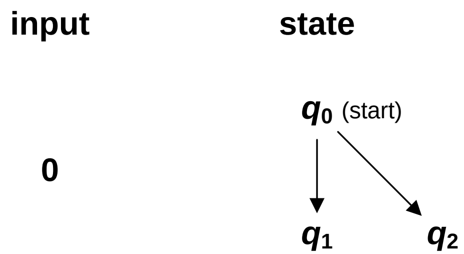



Determine if the string 010010 will be accepted

# ε - TRANSITIONS

010010




1. Initially, the NFA will be at state $q_0$.

   Assume the first input symbol (0) arrives.

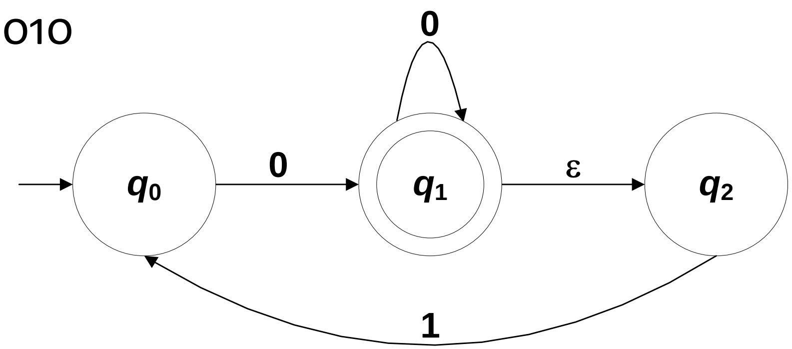
   The NFA goes to $q_1$.

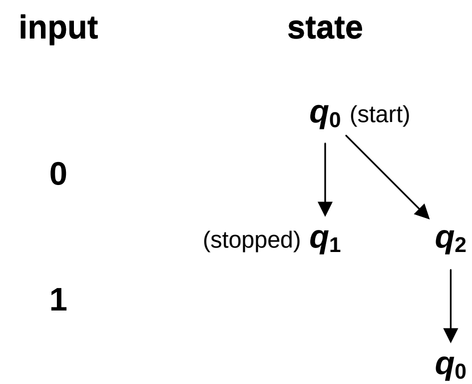
   At $q_1$, it has the option of immediately going to $q_2$ without any new input symbol arriving (because of the ε-transition from $q_1$ to $q_2$).

   The NFA will consider both and will be in two states, $q_1$ and $q_2$.

# ε - TRANSITIONS

010010




2. Assume the second input symbol (1) arrives.

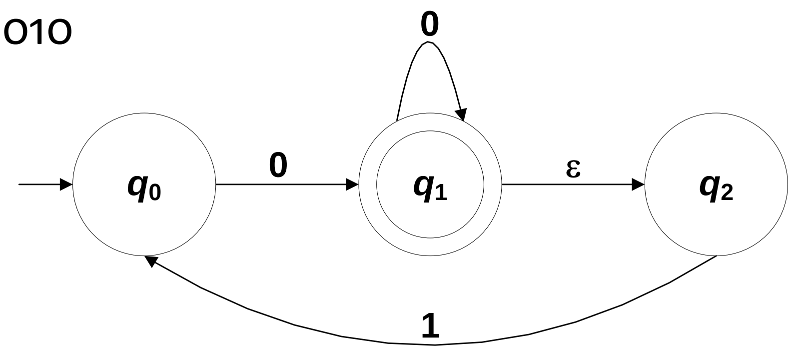
   The copy of the NFA which is at state $q_1$ stops computing since it has nowhere to go.

   The copy at $q_2$ will go to $q_0$.

   The NFA will only be in one state which is $q_0$.

# ε - TRANSITIONS

010010

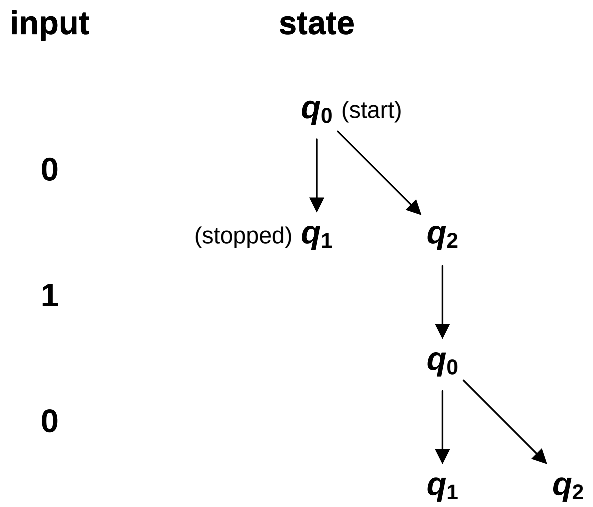



3. Assume the third input symbol (0) arrives.

   The NFA goes to $q_1$.

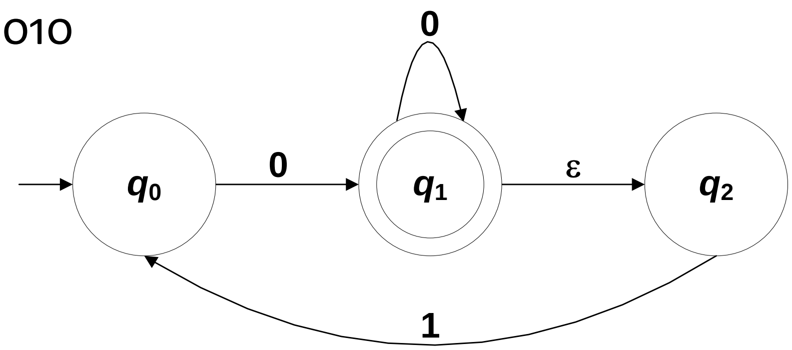
   Once at $q_1$, it has the option of immediately going to $q_2$ without any new input symbol arriving.

   The NFA will consider both and will be in two states, $q_1$ and $q_2$.



# ε - TRANSITIONS

010010

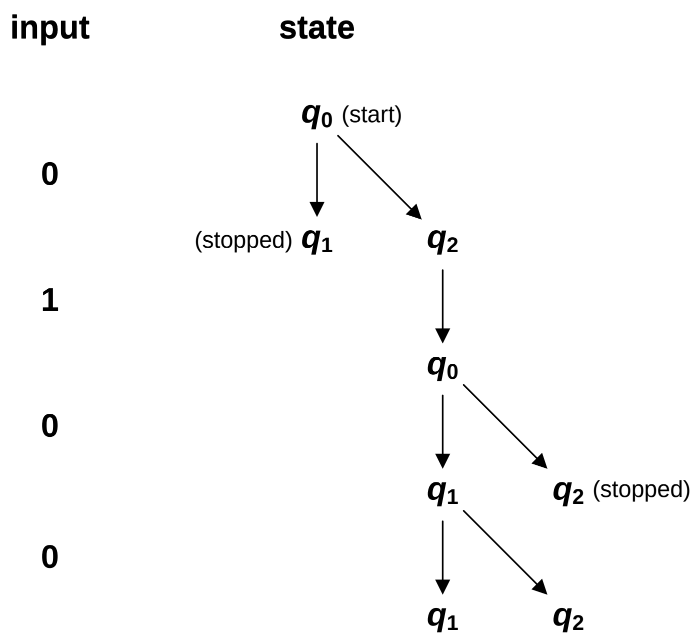



4. Assume the fourth input symbol (0) arrives.

   The copy of the NFA which is at state $q_1$ stays at $q_1$.

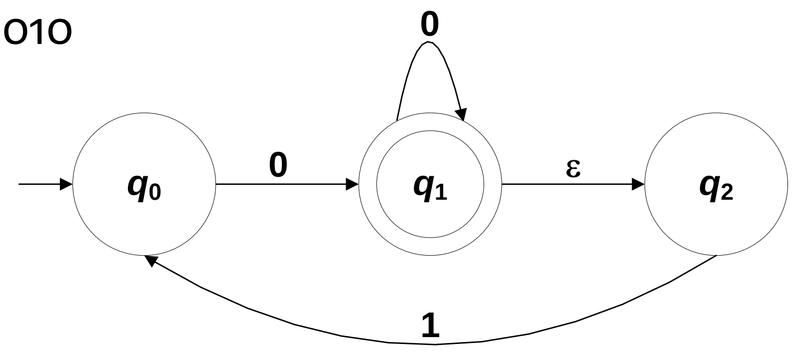
   Since there is an ε-transition from $q_1$ to $q_2$, it will also immediately go to $q_2$.

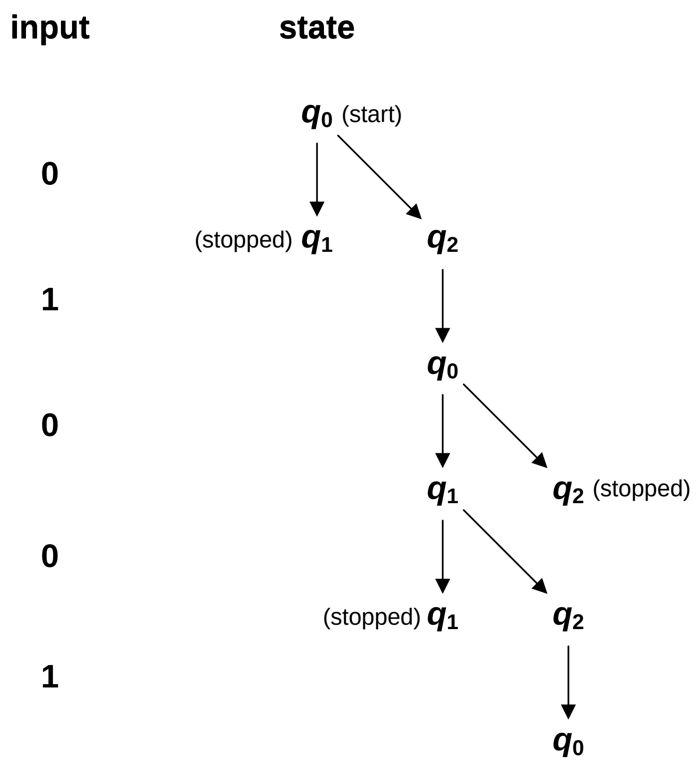
   The other copy of the NFA (the one at state $q_2$) stops because it has nowhere to go.



# ε - TRANSITIONS

010010



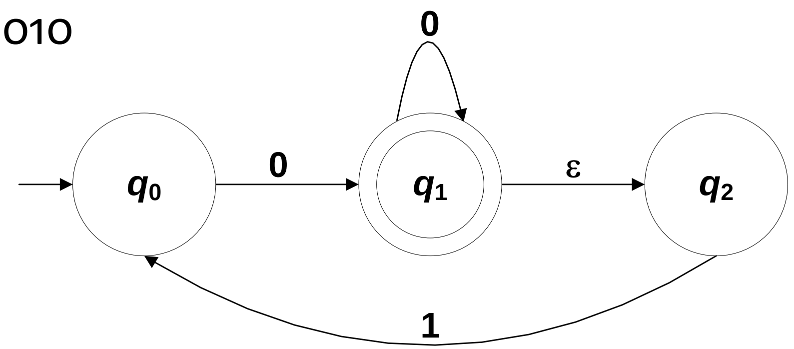

5. Assume the fifth input symbol (1) arrives.

   The copy of the NFA which is at state $q_1$ stops since it has nowhere to go.

   The remaining copy of the NFA (the one at state $q_2$) goes to $q_0$.

# ε - TRANSITIONS

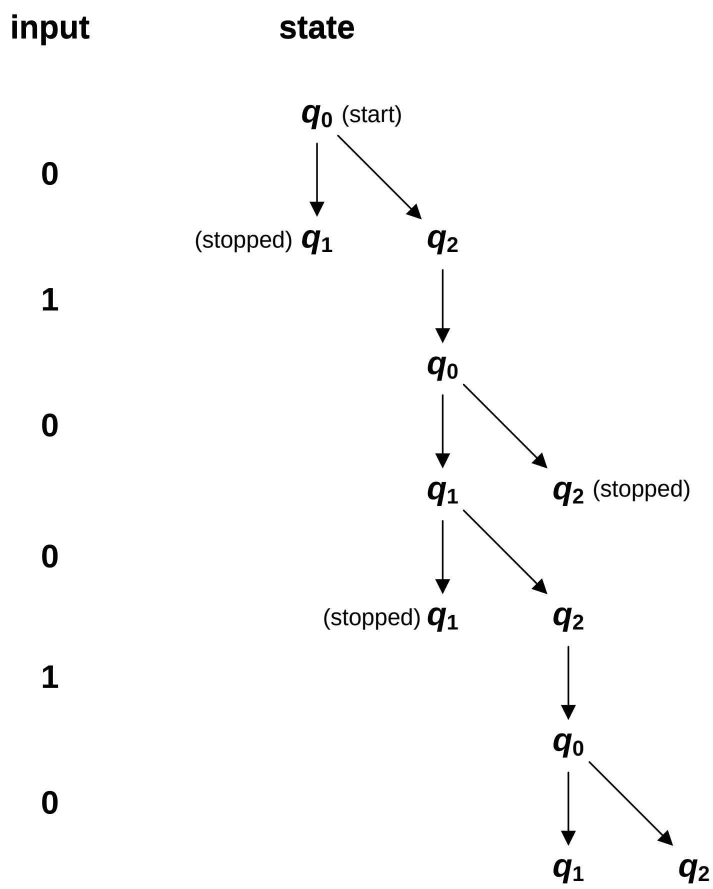
010010




6. Assume the sixth and last input symbol (0) arrives.

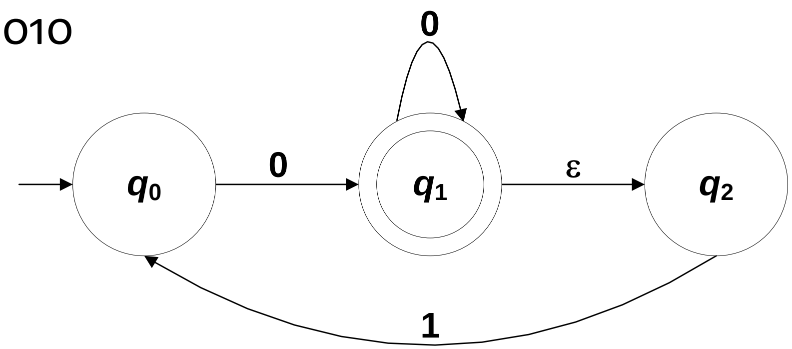
   The NFA goes to $q_1$.

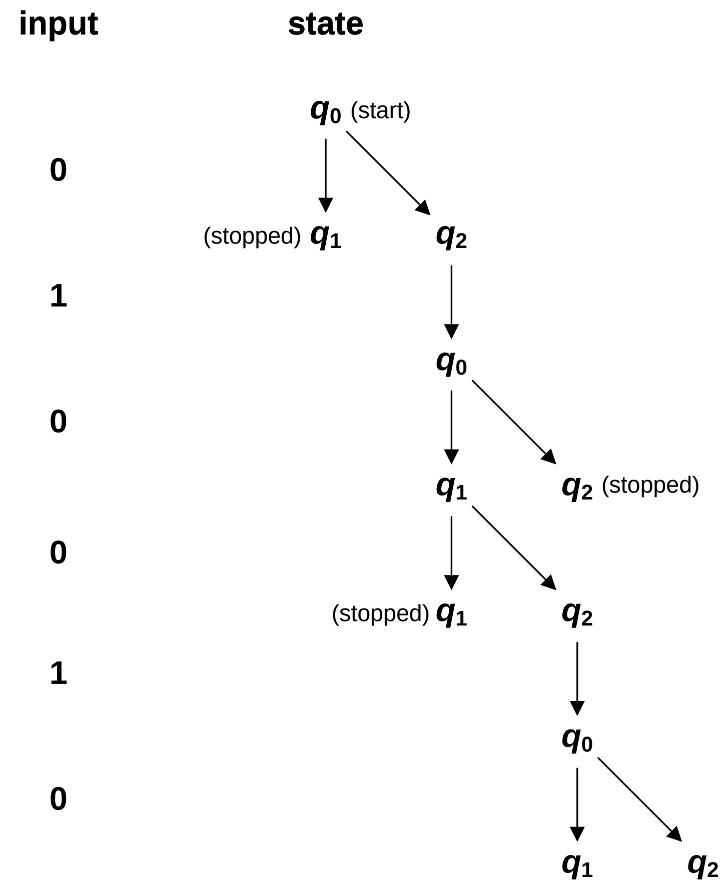
   At $q_1$, it has the option of immediately going to $q_2$ without any new input symbol arriving (because of the ε-transition from $q_1$ to $q_2$).



# ε - TRANSITIONS

010010




The NFA will then be at state $q_1$ and state $q_2$.

Since one of these states is a final state ($q_1$), the string 010010 is accepted.

# FORMAL DEFINITION OF NFAs

- The formal definition of a NFA is similar to that of a DFA.
- The main difference lies in the transition function.
- The transition function of an NFA must consider the possibility of state transitions even if there is no input symbol (ε-transitions).
- The transition function must also consider the possibility that the NFA may go to several states given the same input symbol.

# DIFFERENCES BETWEEN NFAs AND DFAs

- To summarize the differences between an NFA and a DFA:

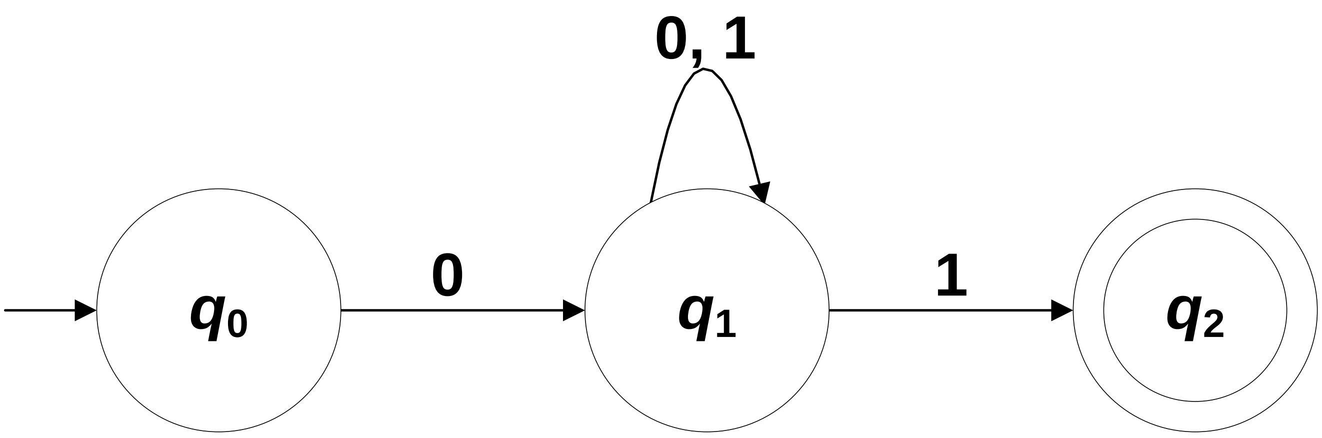
1. NFAs can move to more than one state from any given current state. They can therefore be in several states at the same time.

2. In a given state, NFAs may not necessarily have a transition edge for each symbol in the alphabet.

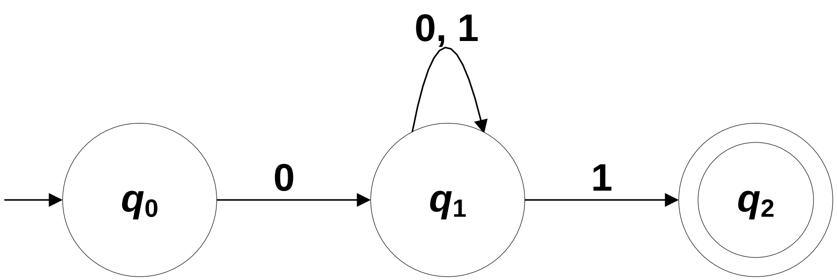
3. NFAs can move to a state even without any input symbol arriving (ε-transitions).

# FORMAL DEFINITION OF NFAs

- Example:



# FORMAL DEFINITION OF NFAs



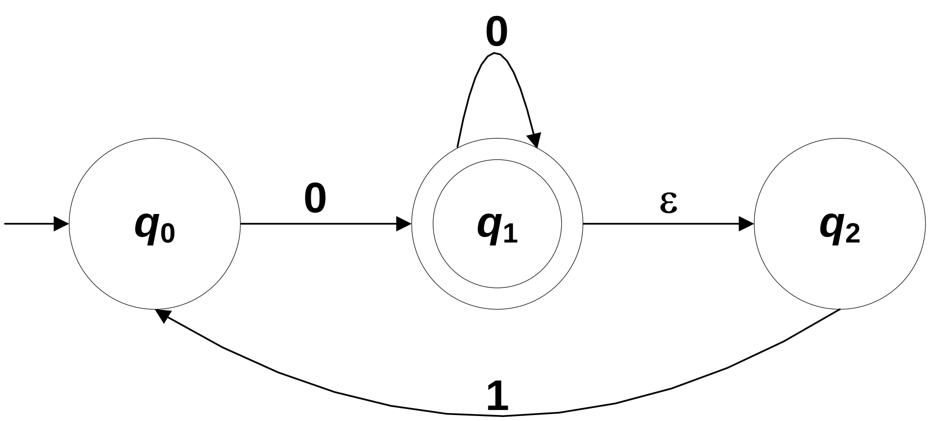

- The formal definition of this NFA is a 5-tuple $N_1 = \{Q, \Sigma, \delta, q_0, F\}$ where:

1. $Q = \{q_0, q_1, q_2\}$
2. $\Sigma = \{0, 1\}$
3. $\delta$ :

| | 0 | 1 | ε |
|---|---|---|---|
| $q_0$ | $q_1$ | -- | -- |
| $q_1$ | $q_1$ | $q_1, q_2$ | -- |
| $q_2$ | -- | -- | -- |

4. Start State = $q_0$
5. $F = \{q_2\}$

# FORMAL DEFINITION OF NFAs




- The formal definition of this NFA is a 5-tuple $N_2 = \{Q, \Sigma, \delta, q_0, F\}$ where:

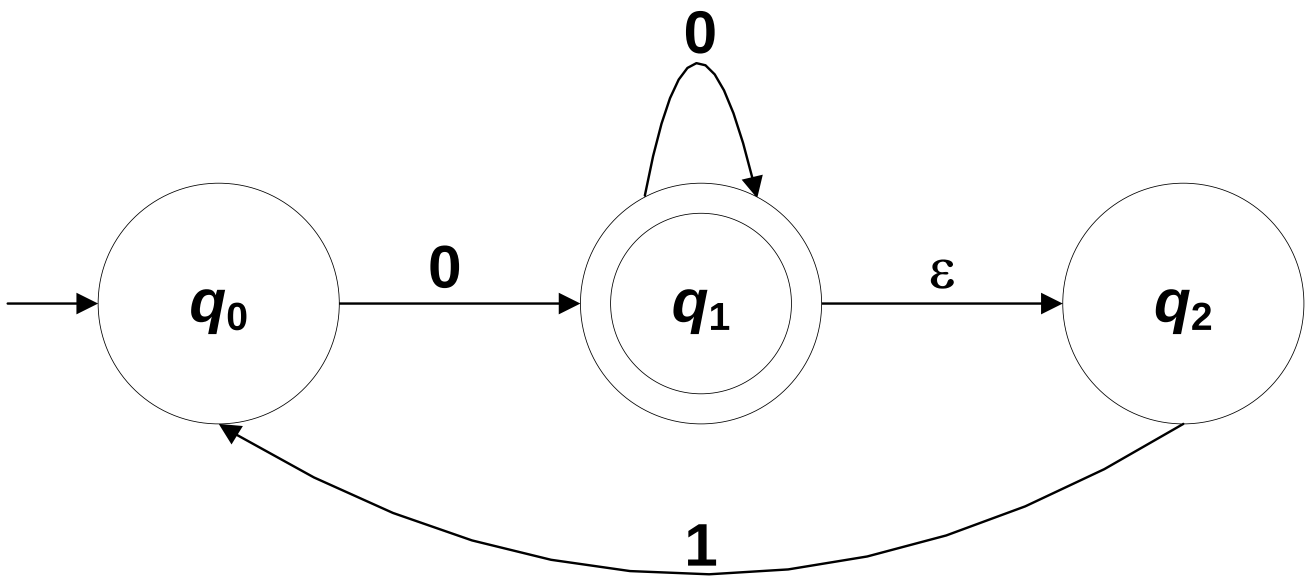
1. $Q = \{q_0, q_1, q_2\}$
2. $\Sigma = \{0, 1\}$
3. $\delta$:

| | 0 | 1 | ε |
|---|---|---|---|
| $q_0$ | $q_1$ | -- | -- |
| $q_1$ | $q_1$ | -- | $q_2$ |
| $q_2$ | -- | $q_0$ | -- |

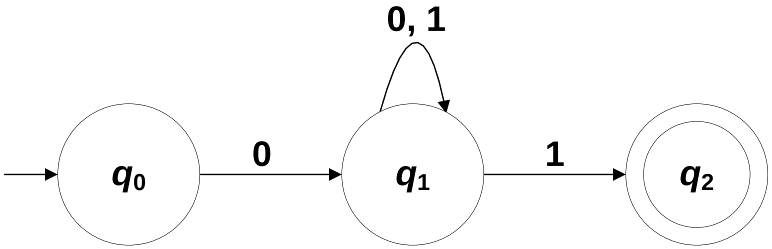
4. Start State = $q_0$
5. $F = \{q_1\}$

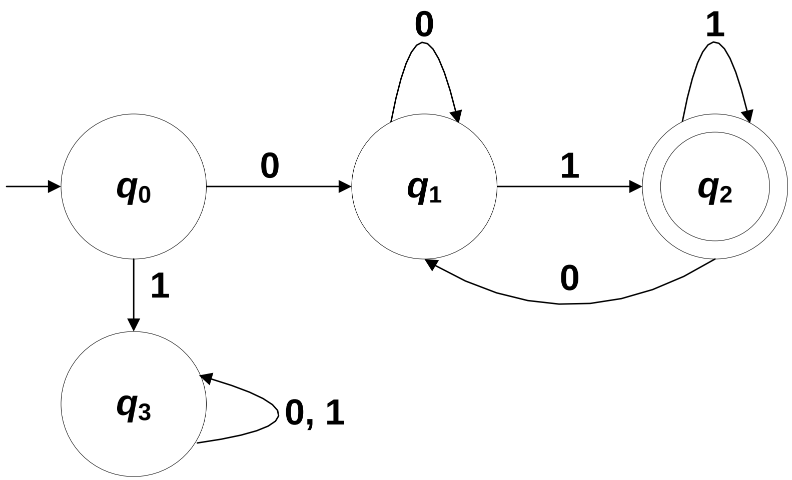
# FORMAL DEFINITION OF NFAs

- Another example:



# NFA AND DFA EQUIVALENCE




- In an NFA, a state may have zero, one, or many outgoing edges for each input symbol

- DFAs can only have one outgoing edge for each input symbol.

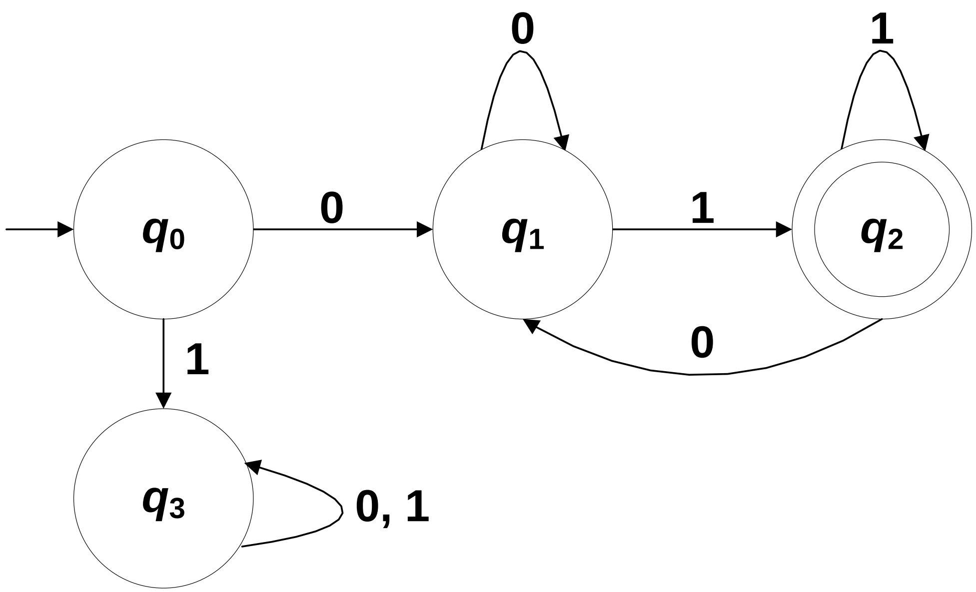
- This means that a DFA is a restricted type of NFA. It can therefore be said that all DFAs are NFAs.

- However, not all NFAs are DFAs.

- In other words, nondeterminism is a generalization of determinism.

# NFA AND DFA EQUIVALENCE

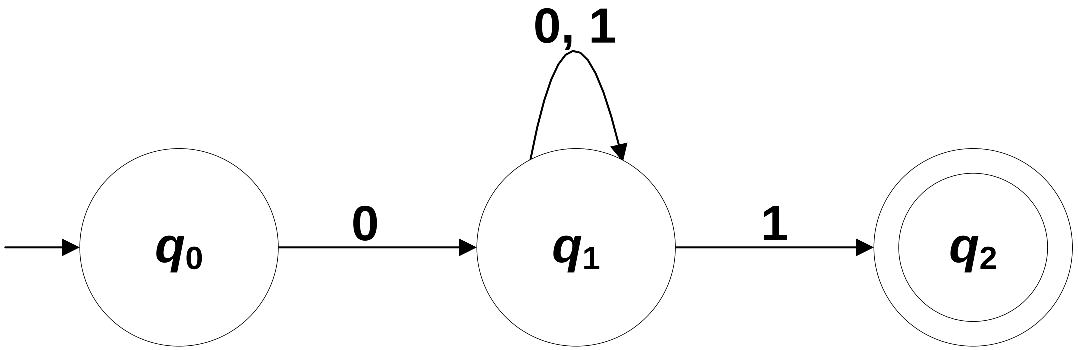
- Consider the following DFA $M_1$ from the previous discussions:




$M_1$ accepts strings over the alphabet $\Sigma = \{0, 1\}$ that start with a 0 and end with a 1.

# NFA AND DFA EQUIVALENCE

- Consider now the following NFA $N_1$ also from the previous discussions:




$N_1$ also accepts strings over the alphabet $\Sigma = \{0, 1\}$ that start with a 0 and end with 1.

In other words, DFA $M_1$ and NFA $N_1$ recognize the same language. That is: $L(M_1) = L(N_1) = \{w \mid w \text{ starts with 0 and ends with 1}\}$

# NFA AND DFA EQUIVALENCE

- Two machines are equivalent if they recognize the same language. Therefore, $M_1$ is equivalent to $N_1$.
- If a language is recognized by a DFA, then there is an NFA that will also recognize it. This is because all DFAs are NFAs.
- The question now is: if there is a language that is recognized by an NFA, is there a DFA that will also recognize it?
- In other words, is there an equivalent DFA for any NFA?
- To answer this question, a procedure must be constructed to convert any NFA to its equivalent DFA.

# NFA AND DFA EQUIVALENCE

- Recall that an NFA can be in one or more states at any given time.
- Consider NFA $N_1$ which has three states $q_0$, $q_1$, and $q_2$. At any time, the NFA may be in:

1. state $q_0$ only
2. state $q_1$ only
3. state $q_2$ only
4. states $q_0$ and $q_1$
5. states $q_0$ and $q_2$
6. states $q_1$ and $q_2$
7. states $q_0$, $q_1$, and $q_2$
8. dead state (stop processing)

Observe that the possible states the NFA can be is the power set (set of all subsets) of its set of states $Q$. The dead state is equivalent to the null set.

iACADEMY SCHOOL OF COMPUTING • SCHOOL OF BUSINESS • SCHOOL OF DESIGN

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

- The DFA to be constructed from this NFA will have a state that will represent each of the subset of the power set.

- For example, the state in which the NFA is in states $q_o$ and $q_1$ will be represented by a single state (that may be called $q_{o1}$) in its equivalent DFA.

- If the NFA has $n$ states, then the equivalent DFA will have $2^n$ states (number of states is finite).

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

- So the states of the equivalent DFA will be:

| NFA State | DFA State |
|---|---|
| $q_0$ | $q_0$ |
| $q_1$ | $q_1$ |
| $q_2$ | $q_2$ |
| $q_0, q_1$ | $q_{01}$ |
| $q_0, q_2$ | $q_{02}$ |
| $q_1, q_2$ | $q_{12}$ |
| $q_0, q_1, q_2$ | $q_{012}$ |
| Dead State | $q_{dead}$ |

- The initial state of the NFA $N_1$ is $q_o$ so the initial state of the DFA will also be $q_o$.

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

| NFA State | DFA State |
|---|---|
| $q_0$ | $q_0$ |
| $q_1$ | $q_1$ |
| $q_2$ | $q_2$ |
| $q_0, q_1$ | $q_{01}$ |
| $q_0, q_2$ | $q_{02}$ |
| $q_1, q_2$ | $q_{12}$ |
| $q_0, q_1, q_2$ | $q_{012}$ |
| Dead State | $q_{dead}$ |

- The final state of NFA $N_1$ is $q_2$. This means that the NFA will be in a final state if $q_2$ is included in the subset.

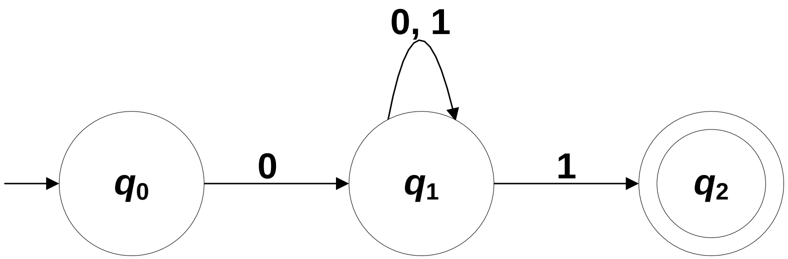
  Therefore, the final states for the DFA are $q_2$, $q_{02}$, $q_{12}$, and $q_{012}$.

- Take note that not all of the states of the equivalent DFA are reachable from the start state. So it is possible that some of these states will be removed eventually.

  The states that will be removed will be determined upon the construction of the transition function of the DFA.

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_1$:




Starting at state $q_0$:

If input = 0, the NFA will go to state $q_1$. For the DFA, this will be state $q_1$.

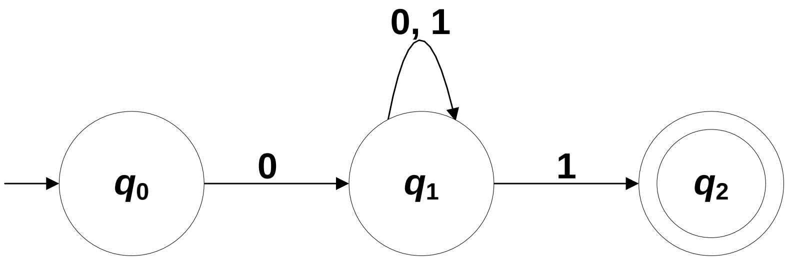
If input = 1, the NFA will go to a dead state. For the DFA, this will be state $q_{dead}$.

The next states to be analyzed are states $q_1$ and $q_{dead}$ (the states reachable from $q_0$).

| | 0 | 1 |
|---|---|---|
| $q_0$ | $q_1$ | $q_{dead}$ |
| $q_1$ | | |
| $q_2$ | | |
| $q_{01}$ | | |
| $q_{02}$ | | |
| $q_{12}$ | | |
| $q_{012}$ | | |
| $q_{dead}$ | | |

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_1$:




| | 0 | 1 |
|---|---|---|
| $q_0$ | $q_1$ | $q_{dead}$ |
| $q_1$ | $q_1$ | $q_{12}$ |
| $q_2$ | | |
| $q_{01}$ | | |
| $q_{02}$ | | |
| $q_{12}$ | | |
| $q_{012}$ | | |
| $q_{dead}$ | | |

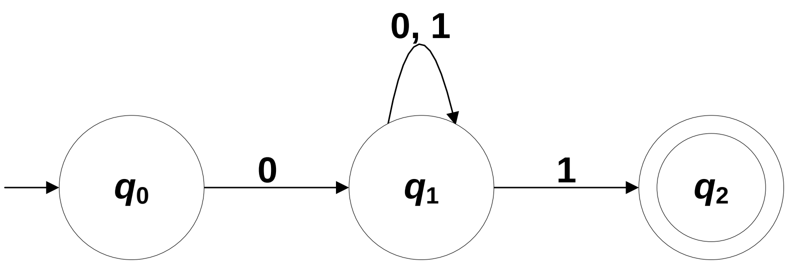
From state $q_1$:

If input = 0, the NFA will be at state $q_1$ only. For the DFA, this will be state $q_1$.

If input = 1, the NFA will be in states $q_1$ and $q_2$. For the DFA, this will be state $q_{12}$.

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_1$:




| | 0 | 1 |
|---|---|---|
| $q_0$ | $q_1$ | $q_{dead}$ |
| $q_1$ | $q_1$ | $q_{12}$ |
| $q_2$ | | |
| $q_{01}$ | | |
| $q_{02}$ | | |
| $q_{12}$ | | |
| $q_{012}$ | | |
| $q_{dead}$ | $q_{dead}$ | $q_{dead}$ |

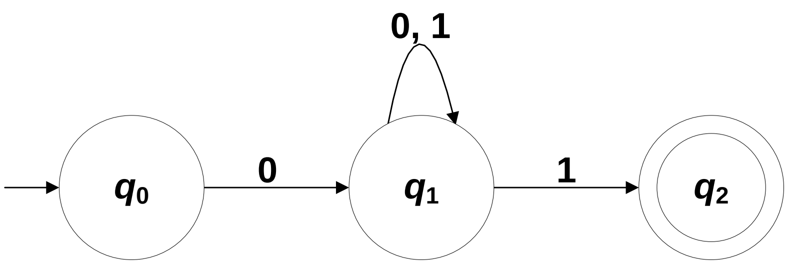
From state $q_{dead}$:

The NFA will stay at its dead state regardless of the input symbol received.

The next state to be analyzed is state $q_{12}$ since it is reachable from state $q_0$ (through state $q_1$).

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_1$:




| | 0 | 1 |
|---|---|---|
| $q_0$ | $q_1$ | $q_{dead}$ |
| $q_1$ | $q_1$ | $q_{12}$ |
| $q_2$ | | |
| $q_{01}$ | | |
| $q_{02}$ | | |
| $q_{12}$ | $q_1$ | |
| $q_{012}$ | | |
| $q_{dead}$ | $q_{dead}$ | $q_{dead}$ |

From state $q_{12}$:

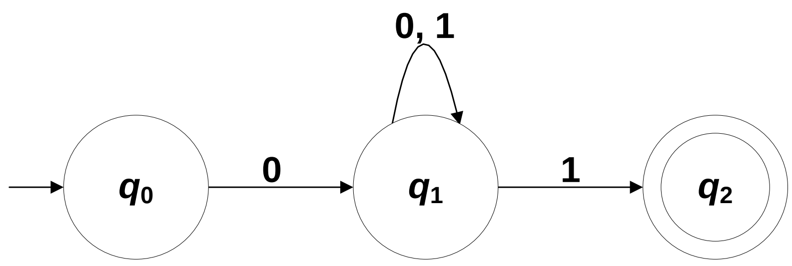
If input = 0, the copy of the NFA at $q_1$ stays at $q_1$.

The copy at $q_2$ will stop processing.

So the NFA will be at state $q_1$ only. For the DFA, this will be state $q_1$.

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_1$:




| | 0 | 1 |
|---|---|---|
| $q_0$ | $q_1$ | $q_{dead}$ |
| $q_1$ | $q_1$ | $q_{12}$ |
| $q_2$ | | |
| $q_{01}$ | | |
| $q_{02}$ | | |
| $q_{12}$ | $q_1$ | $q_{12}$ |
| $q_{012}$ | | |
| $q_{dead}$ | $q_{dead}$ | $q_{dead}$ |

From state $q_{12}$:

If input = 1, the copy of the NFA at state $q_1$ will go to states $q_1$ and $q_2$.

While the copy at $q_2$ will stop processing. So the NFA will be at states $q_1$ and $q_2$.

For the DFA, this will be state $q_{12}$.

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

The final transition table will be:

| | 0 | 1 |
|---|---|---|
| $q_0$ | $q_1$ | $q_{dead}$ |
| $q_1$ | $q_1$ | $q_{12}$ |
| $q_2$ | | |
| $q_{01}$ | | |
| $q_{02}$ | | |
| $q_{12}$ | $q_1$ | $q_{12}$ |
| $q_{012}$ | | |
| $q_{dead}$ | $q_{dead}$ | $q_{dead}$ |

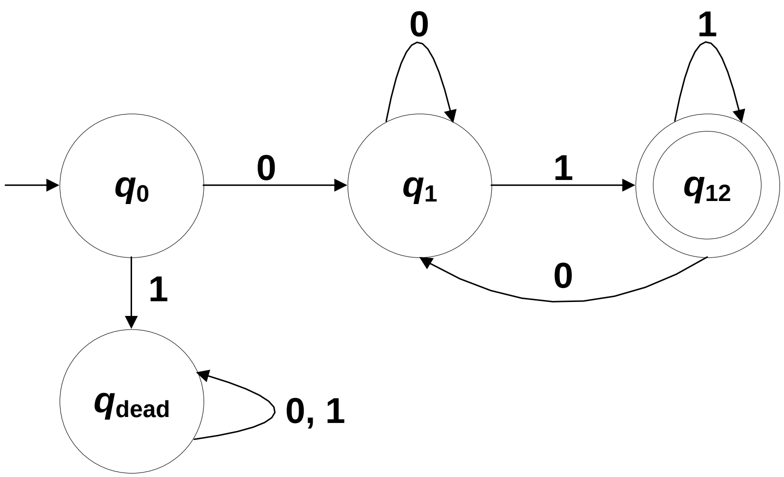
Observe that states $q_2$, $q_{01}$, $q_{02}$, and $q_{012}$ are not reachable from state $q_0$. They can therefore be removed from the transition table.

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

Since all the states reachable from $q_o$ have been analyzed, the transition function will be:

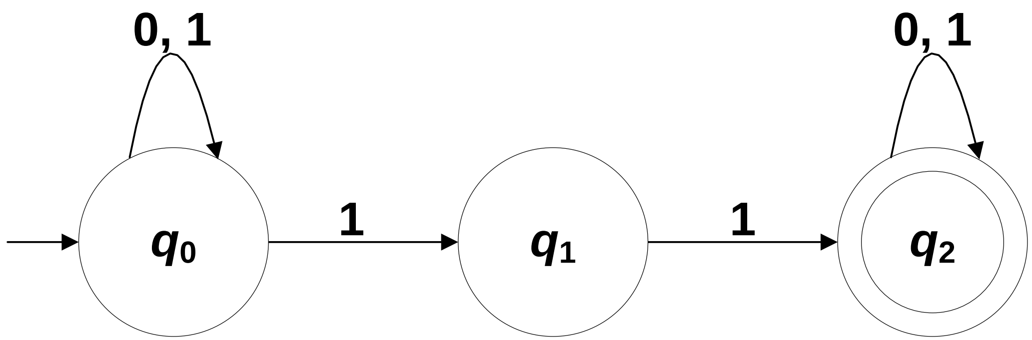
| | 0 | 1 |
|---|---|---|
| $q_0$ | $q_1$ | $q_{dead}$ |
| $q_1$ | $q_1$ | $q_{12}$ |
| $q_{12}$ | $q_1$ | $q_{12}$ |
| $q_{dead}$ | $q_{dead}$ | $q_{dead}$ |

The state diagram of the equivalent DFA is:



# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

- As another example, consider now the following NFA $N_2$:




$N_2$ accepts strings over the alphabet $\Sigma = \{0, 1\}$ that have the substring 11.

Since the NFA has three states, the possible states it can assume are $q_o$ only, $q_1$ only, $q_2$ only, $q_o$ and $q_1$, $q_o$ and $q_2$, $q_1$ and $q_2$, $q_1$ and $q_2$ and $q_3$, and the dead state.

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

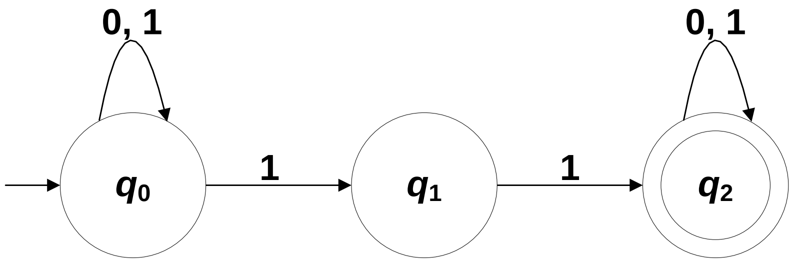
- So the states of the equivalent DFA will be:

| NFA State | DFA State |
|---|---|
| $q_0$ | $q_0$ |
| $q_1$ | $q_1$ |
| $q_2$ | $q_2$ |
| $q_0, q_1$ | $q_{01}$ |
| $q_0, q_2$ | $q_{02}$ |
| $q_1, q_2$ | $q_{12}$ |
| $q_0, q_1, q_2$ | $q_{012}$ |
| Dead State | $q_{dead}$ |

The initial state of the NFA is $q_o$ so the initial state of the DFA will also be $q_o$. The final state of the NFA is $q_2$. This means that the NFA will be in a final state if $q_2$ is included in the subset.

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_2$:




| | 0 | 1 |
|---|---|---|
| $q_0$ | $q_0$ | $q_{01}$ |
| $q_1$ | | |
| $q_2$ | | |
| $q_{01}$ | | |
| $q_{02}$ | | |
| $q_{12}$ | | |
| $q_{012}$ | | |
| $q_{dead}$ | | |

Starting at state $q_0$:

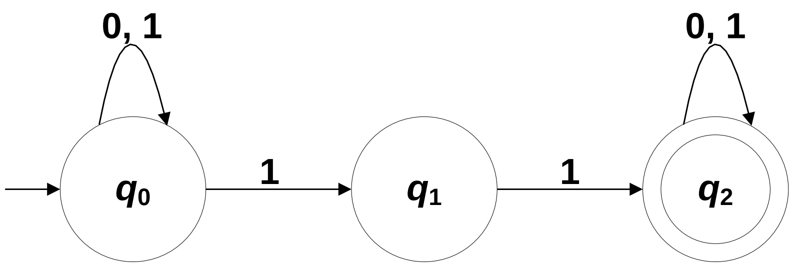
If input = 0, the NFA will stay at $q_0$. For the DFA, this will be state $q_0$.

If input = 1, the NFA will be at $q_0$ and $q_1$. For the DFA, this will be state $q_{01}$.

The next state to be analyzed is state $q_{01}$.

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_2$:




| | 0 | 1 |
|---|---|---|
| $q_0$ | $q_0$ | $q_{01}$ |
| $q_1$ | | |
| $q_2$ | | |
| $q_{01}$ | $q_0$ | |
| $q_{02}$ | | |
| $q_{12}$ | | |
| $q_{012}$ | | |
| $q_{dead}$ | | |

From state $q_{01}$:

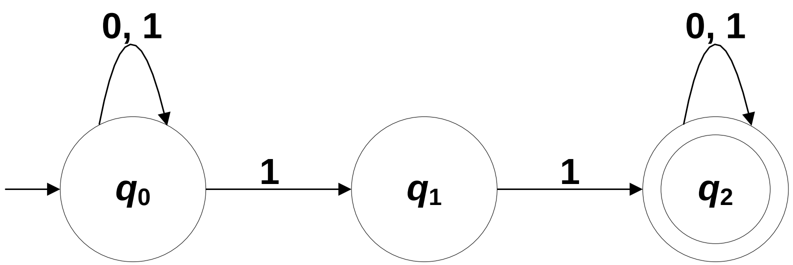
If input = 0, the copy of the NFA at state $q_0$ will stay at $q_0$.

The copy at $q_1$ will stop processing.

So the NFA will be at state $q_0$ only. For the DFA, this will be state $q_0$.

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_2$:




| | 0 | 1 |
|---|---|---|
| $q_0$ | $q_0$ | $q_{01}$ |
| $q_1$ | | |
| $q_2$ | | |
| $q_{01}$ | $q_0$ | $q_{012}$ |
| $q_{02}$ | | |
| $q_{12}$ | | |
| $q_{012}$ | | |
| $q_{dead}$ | | |

From state $q_{01}$:

If input = 1, the copy of the NFA at state $q_0$ will be at states $q_0$ and $q_1$.

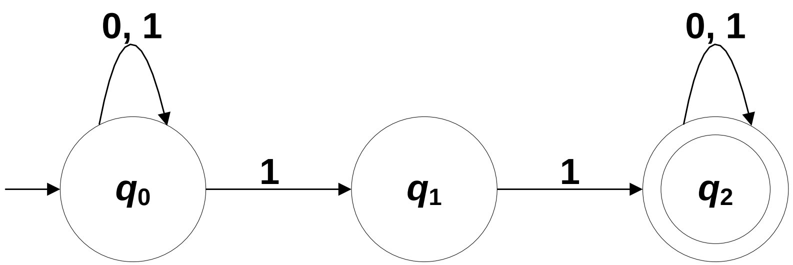
The copy at state $q_1$ will go to state $q_2$.

So the NFA will be at states $q_0$, $q_1$, and $q_2$. For the DFA, this will be state $q_{012}$.

The next state to be analyzed is state $q_{012}$.

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_2$:




| | 0 | 1 |
|---|---|---|
| $q_0$ | $q_0$ | $q_{01}$ |
| $q_1$ | | |
| $q_2$ | | |
| $q_{01}$ | $q_0$ | $q_{012}$ |
| $q_{02}$ | | |
| $q_{12}$ | | |
| $q_{012}$ | $q_{02}$ | |
| $q_{dead}$ | | |

From state $q_{012}$:

If input = 0, the copy of the NFA at $q_0$ will stay at $q_0$.

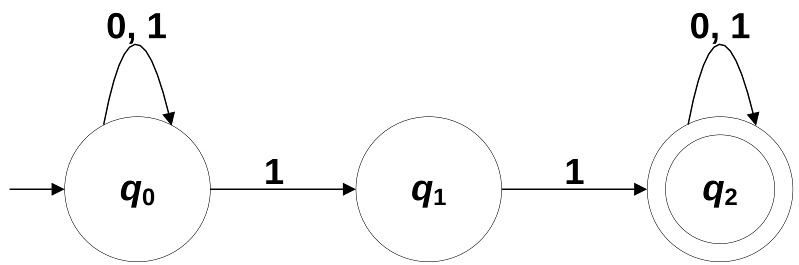
The copy at $q_1$ will stop processing.

And the copy at $q_2$ will stay at $q_2$.

So the NFA will be at states $q_0$ and $q_2$. For the DFA, this will be state $q_{02}$.

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_2$:




| | 0 | 1 |
|---|---|---|
| $q_0$ | $q_0$ | $q_{01}$ |
| $q_1$ | | |
| $q_2$ | | |
| $q_{01}$ | $q_0$ | $q_{012}$ |
| $q_{02}$ | | |
| $q_{12}$ | | |
| $q_{012}$ | $q_{02}$ | $q_{012}$ |
| $q_{dead}$ | | |

From state $q_{012}$:

If input = 1, the copy of the NFA at $q_0$ will be at state $q_0$ and $q_1$.

The copy at $q_1$ will go to state $q_2$.

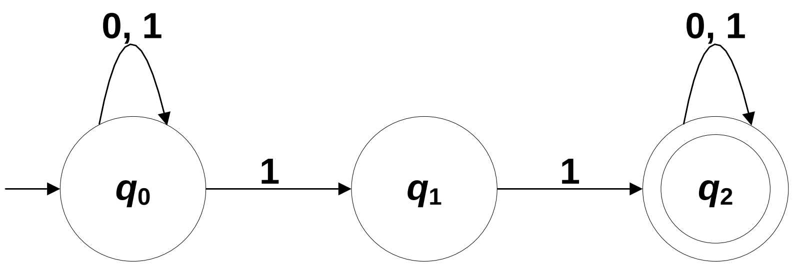
And the copy of $q_2$ will stay at $q_2$.

So the NFA will be at states $q_0$, $q_1$, and $q_2$. For the DFA, this will be state $q_{012}$.

The next state to be analyzed is state $q_{02}$.

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_2$:




| | 0 | 1 |
|---|---|---|
| $q_0$ | $q_0$ | $q_{01}$ |
| $q_1$ | | |
| $q_2$ | | |
| $q_{01}$ | $q_0$ | $q_{012}$ |
| $q_{02}$ | $q_{02}$ | |
| $q_{12}$ | | |
| $q_{012}$ | $q_{02}$ | $q_{012}$ |
| $q_{dead}$ | | |

From state $q_{02}$:

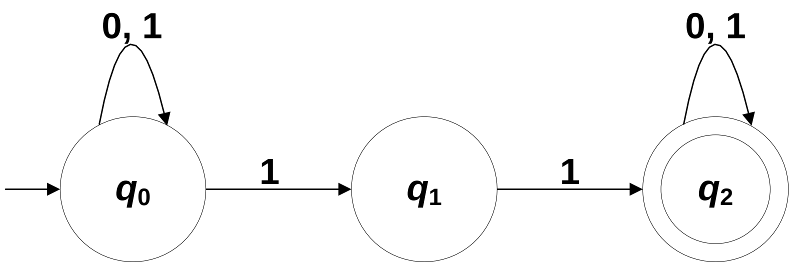
If input = 0, the copy of the NFA at $q_0$ will stay at $q_0$.

The copy at $q_2$ will stay at $q_2$.

So the NFA will be at states $q_0$ and $q_2$. For the DFA, this will be state $q_{02}$.

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_2$:




| | 0 | 1 |
|---|---|---|
| $q_0$ | $q_0$ | $q_{01}$ |
| $q_1$ | | |
| $q_2$ | | |
| $q_{01}$ | $q_0$ | $q_{012}$ |
| $q_{02}$ | $q_{02}$ | $q_{012}$ |
| $q_{12}$ | | |
| $q_{012}$ | $q_{02}$ | $q_{012}$ |
| $q_{dead}$ | | |

From state $q_{02}$:

If input = 1, the copy of the NFA at $q_0$ will be at states $q_0$ and $q_1$.

The copy at $q_2$ will stay at state $q_2$.

So the NFA will be at states $q_0$, $q_1$, and $q_2$. For the DFA, this will be state $q_{012}$.

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

Since all the states reachable from $q_o$ have been analyzed, the transition function will be:

| | 0 | 1 |
|---|---|---|
| $q_0$ | $q_0$ | $q_{01}$ |
| $q_1$ | | |
| $q_2$ | | |
| $q_{01}$ | $q_0$ | $q_{012}$ |
| $q_{02}$ | $q_{02}$ | $q_{012}$ |
| $q_{12}$ | | |
| $q_{012}$ | $q_{02}$ | $q_{012}$ |
| $q_{dead}$ | | |

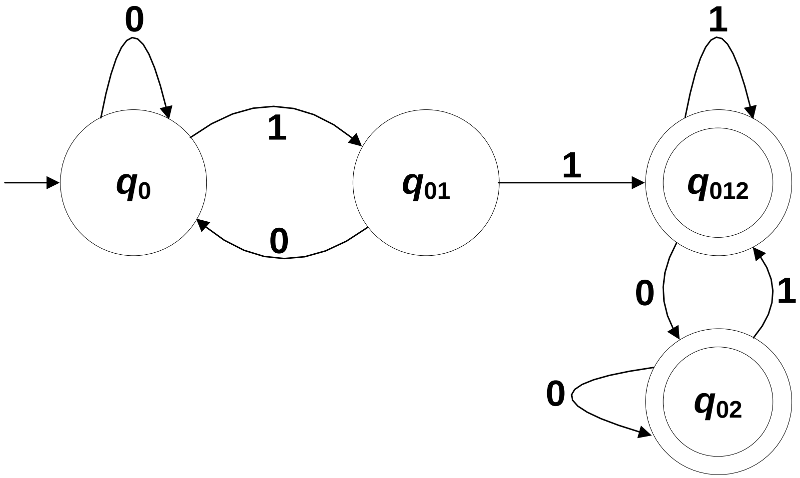
Observe that states $q_1$, $q_2$, $q_{12}$, and $q_{dead}$ are not reachable from state $q_o$. They can therefore be removed from the transition table.

# NFA TO DFA CONVERSION (WITHOUT ε - TRANSITIONS)

The transition function will be:

| | 0 | 1 |
|---|---|---|
| $q_0$ | $q_0$ | $q_{01}$ |
| $q_{01}$ | $q_0$ | $q_{012}$ |
| $q_{02}$ | $q_{02}$ | $q_{012}$ |
| $q_{012}$ | $q_{02}$ | $q_{012}$ |

The state diagram of the equivalent DFA is:



# NFA TO DFA CONVERSION (WITH ε - TRANSITIONS)

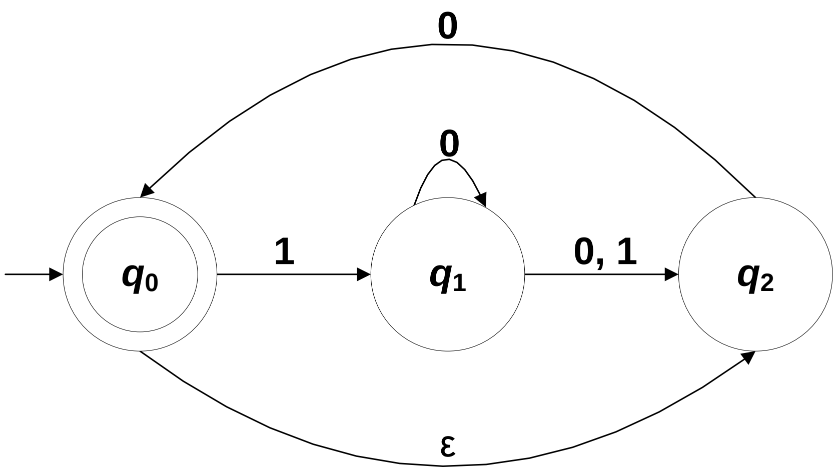
- The procedure for converting an NFA with ε-transitions into its equivalent DFA is similar to the procedure discussed in the previous topic.

- The main difference is that special care has to be given to the ε-transitions.

- The main thing to remember is that once an NFA goes to a certain state $q_i$ which has an ε-transition to state $q_j$, the NFA automatically goes to $q_j$ also.

# NFA TO DFA CONVERSION (WITH ε - TRANSITIONS)

- Consider the following NFA $N_3$:




Since the NFA has three states $q_o$, $q_1$, and $q_2$, the possible states it can assume are:

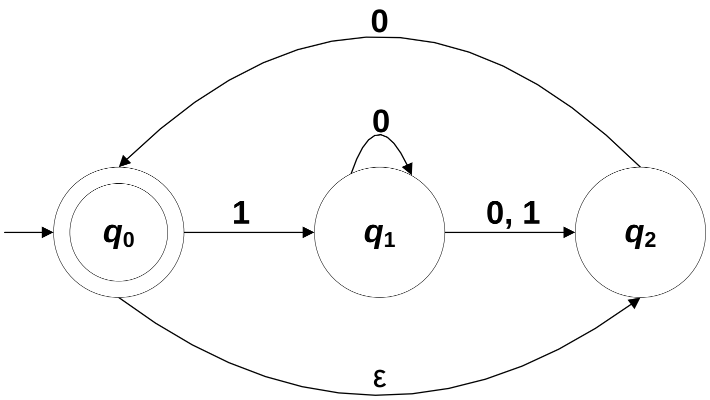
$q_o$ only,
$q_1$ only,
$q_2$ only,
$q_o$, $q_1$ and $q_2$,
$q_o$ and $q_1$,
$q_o$ and $q_2$,
$q_1$ and $q_2$,
the dead state.

# NFA TO DFA CONVERSION (WITH ε - TRANSITIONS)

- So the states of the equivalent DFA will be:

| NFA State | DFA State |
|---|---|
| $q_0$ | $q_0$ |
| $q_1$ | $q_1$ |
| $q_2$ | $q_2$ |
| $q_0, q_1$ | $q_{01}$ |
| $q_0, q_2$ | $q_{02}$ |
| $q_1, q_2$ | $q_{12}$ |
| $q_0, q_1, q_2$ | $q_{012}$ |
| Dead State | $q_{dead}$ |

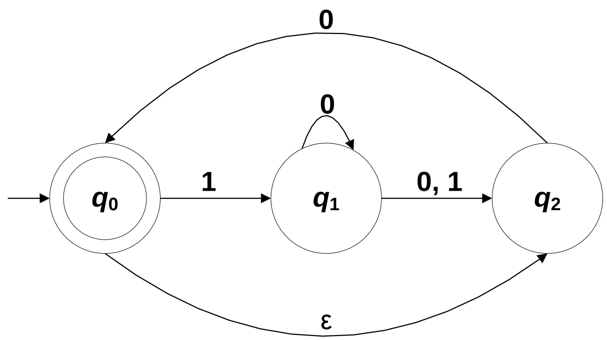
# NFA TO DFA CONVERSION (WITH ε - TRANSITIONS)




- The initial state of the NFA $N_3$ is $q_o$.
- However, state $q_o$ has an ε-transition to state $q_2$.
- This means that before any input arrives, the NFA is at state $q_o$. But because of the ε-transition to state $q_2$, the NFA will also go to state $q_2$. So even before the start of any computation, the NFA will be at states $q_o$ and $q_2$.
- Therefore, the start state of the equivalent DFA is state $q_{o2}$. This is where the construction of the transition function will begin.
- The final state of the NFA is $q_o$. This means that the NFA will be in a final state if $q_o$ is included in the subset. Therefore, the final states for the DFA are $q_o$, $q_{o1}$, $q_{o2}$, and $q_{o12}$.

# NFA TO DFA CONVERSION (WITH ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_3$:




| | 0 | 1 |
|---|---|---|
| $q_0$ | | |
| $q_1$ | | |
| $q_2$ | | |
| $q_{01}$ | | |
| $q_{02}$ | $q_{02}$ | |
| $q_{12}$ | | |
| $q_{012}$ | | |
| $q_{dead}$ | | |

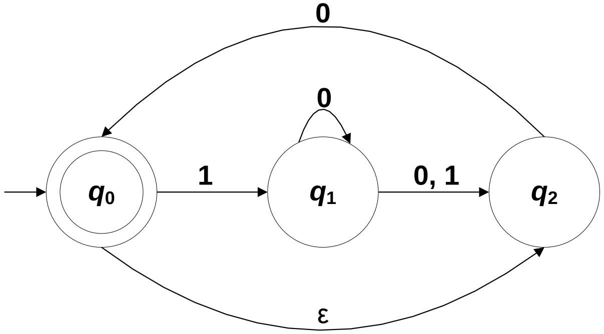
Starting at state $q_{02}$:

If input = 0, the copy of the NFA at state $q_0$ stops processing.

The copy at $q_2$ will go to states $q_0$ and $q_2$. For the DFA, this will be state $q_{02}$.

# NFA TO DFA CONVERSION (WITH ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_3$:




|  | 0 | 1 |
|---|---|---|
| $q_0$ |  |  |
| $q_1$ |  |  |
| $q_2$ |  |  |
| $q_{01}$ |  |  |
| $q_{02}$ | $q_{02}$ | $q_1$ |
| $q_{12}$ |  |  |
| $q_{012}$ |  |  |
| $q_{dead}$ |  |  |

Starting at state $q_{02}$:

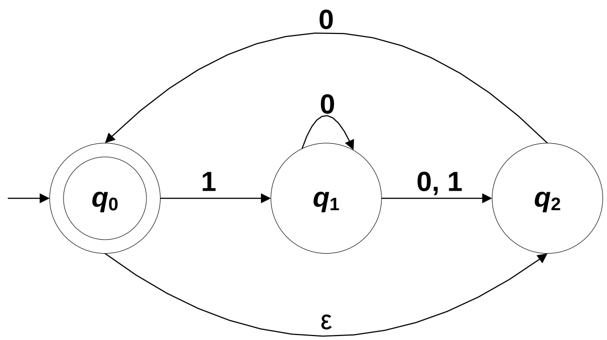
If input = 1, the copy of the NFA at $q_0$ will go to $q_1$.

The copy at $q_2$ stops. For the DFA, this will be state $q_1$.

The next state to be analyzed is state $q_1$.

# NFA TO DFA CONVERSION (WITH ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_3$:




| | 0 | 1 |
|---|---|---|
| $q_0$ | | |
| $q_1$ | $q_{12}$ | $q_2$ |
| $q_2$ | | |
| $q_{01}$ | | |
| $q_{02}$ | $q_{02}$ | $q_1$ |
| $q_{12}$ | | |
| $q_{012}$ | | |
| $q_{dead}$ | | |

From state $q_1$:

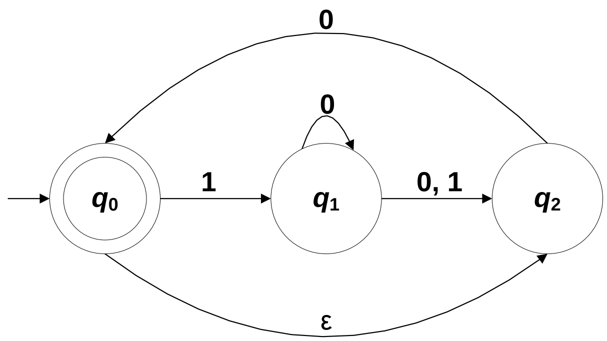
If input = 0, the NFA will go to states $q_1$ and $q_2$. For the DFA, this will be state $q_{12}$.

If input = 1, the NFA will go to state $q_2$. For the DFA, this will be state $q_2$.

The next states to be analyzed are states $q_{12}$ and $q_2$.

# NFA TO DFA CONVERSION (WITH ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_3$:




From state $q_{12}$:

If input = 0, the copy of the NFA at state $q_1$ will go to states $q_1$ and $q_2$.

While the copy at $q_2$ goes to $q_0$ and $q_2$.

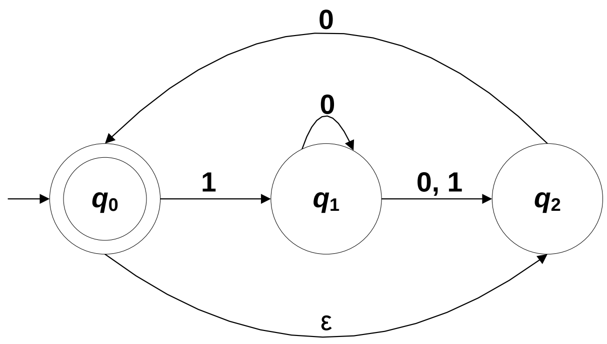
The NFA is therefore at states $q_0$, $q_1$, and $q_2$. For the DFA, this will be state $q_{012}$.

| | 0 | 1 |
|---|---|---|
| $q_0$ | | |
| $q_1$ | $q_{12}$ | $q_2$ |
| $q_2$ | | |
| $q_{01}$ | | |
| $q_{02}$ | $q_{02}$ | $q_1$ |
| $q_{12}$ | $q_{012}$ | |
| $q_{012}$ | | |
| $q_{dead}$ | | |

iACADEMY
SCHOOL OF COMPUTING • SCHOOL OF BUSINESS • SCHOOL OF DESIGN

# NFA TO DFA CONVERSION (WITH ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_3$:




From state $q_{12}$:

If input = 1, the copy of the NFA at state $q_1$ will go to state $q_2$.

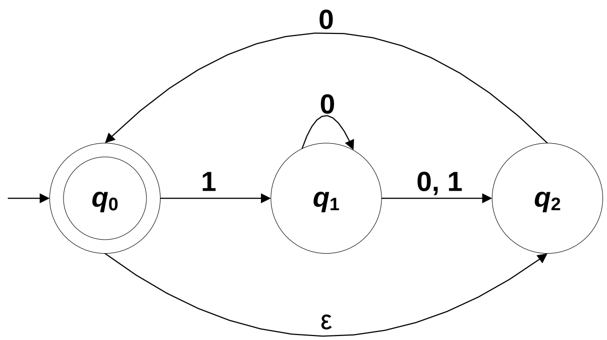
While the copy at state $q_2$ will stop processing.

For the DFA, this will be state $q_2$.

| | 0 | 1 |
|---|---|---|
| $q_0$ | | |
| $q_1$ | $q_{12}$ | $q_2$ |
| $q_2$ | | |
| $q_{01}$ | | |
| $q_{02}$ | $q_{02}$ | $q_1$ |
| $q_{12}$ | $q_{012}$ | $q_2$ |
| $q_{012}$ | | |
| $q_{dead}$ | | |

# NFA TO DFA CONVERSION (WITH ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_3$:




From state $q_2$:

If input = 0, the NFA at will go to states $q_0$ and $q_2$. For the DFA, this will be state $q_{02}$.

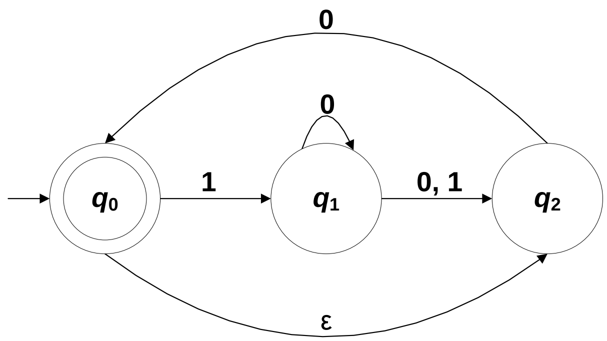
If input = 1, the NFA will stop processing and will cease to exist. For the DFA, this will be state $q_{dead}$.

The next states to be analyzed are states $q_{012}$ and $q_{dead}$.

| | 0 | 1 |
|---|---|---|
| $q_0$ | | |
| $q_1$ | $q_{12}$ | $q_2$ |
| $q_2$ | $q_{02}$ | $q_{dead}$ |
| $q_{01}$ | | |
| $q_{02}$ | $q_{02}$ | $q_1$ |
| $q_{12}$ | $q_{012}$ | $q_2$ |
| $q_{012}$ | | |
| $q_{dead}$ | | |

# NFA TO DFA CONVERSION (WITH ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_3$:




| | 0 | 1 |
|---|---|---|
| $q_0$ | | |
| $q_1$ | $q_{12}$ | $q_2$ |
| $q_2$ | $q_{02}$ | $q_{dead}$ |
| $q_{01}$ | | |
| $q_{02}$ | $q_{02}$ | $q_1$ |
| $q_{12}$ | $q_{012}$ | $q_2$ |
| $q_{012}$ | $q_{012}$ | |
| $q_{dead}$ | | |

From state $q_{012}$:

If input = 0, the copy of the NFA at state $q_0$ will stop processing.

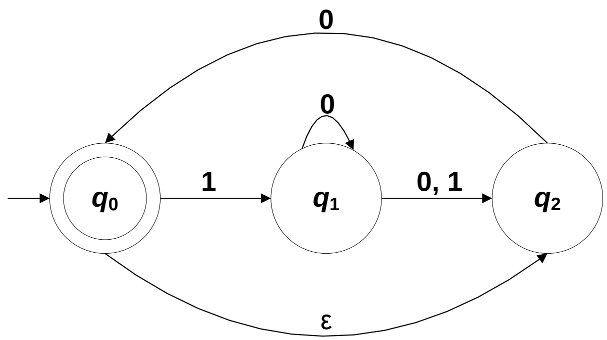
The copy at $q_1$ goes to $q_1$ and $q_2$.

The third copy at $q_2$ goes to $q_0$ and $q_2$.

The NFA is therefore at states $q_0$, $q_1$, and $q_2$. For the DFA, this will be state $q_{012}$.

# NFA TO DFA CONVERSION (WITH ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_3$:




| | 0 | 1 |
|---|---|---|
| $q_0$ | | |
| $q_1$ | $q_{12}$ | $q_2$ |
| $q_2$ | $q_{02}$ | $q_{dead}$ |
| $q_{01}$ | | |
| $q_{02}$ | $q_{02}$ | $q_1$ |
| $q_{12}$ | $q_{012}$ | $q_2$ |
| $q_{012}$ | $q_{012}$ | $q_{12}$ |
| $q_{dead}$ | | |

From state $q_{012}$:

If input = 1, the copy of the NFA at state $q_0$ will go to state $q_1$.

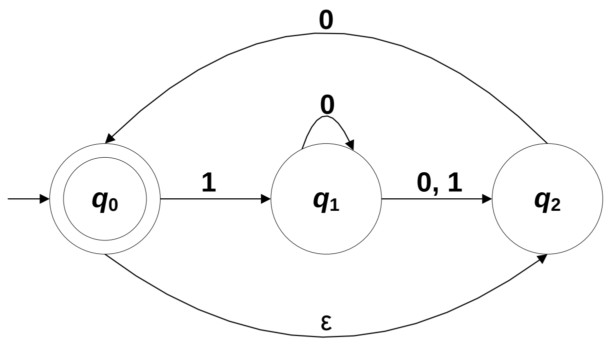
The copy at $q_1$ will go to $q_2$.

The third copy at $q_2$ will stop processing.

The NFA is therefore at states $q_1$ and $q_2$. For the DFA, this will be state $q_{12}$.

# NFA TO DFA CONVERSION (WITH ε - TRANSITIONS)

- Construction of the transition function for the equivalent DFA for NFA $N_3$:




| | 0 | 1 |
|---|---|---|
| $q_0$ | | |
| $q_1$ | $q_{12}$ | $q_2$ |
| $q_2$ | $q_{02}$ | $q_{dead}$ |
| $q_{01}$ | | |
| $q_{02}$ | $q_{02}$ | $q_1$ |
| $q_{12}$ | $q_{012}$ | $q_2$ |
| $q_{012}$ | $q_{012}$ | $q_{12}$ |
| $q_{dead}$ | $q_{dead}$ | $q_{dead}$ |

From state $q_{dead}$:

The NFA will stay at its dead state regardless of the input symbol received.

# NFA TO DFA CONVERSION (WITH ε - TRANSITIONS)

Since all the states reachable from $q_{02}$ have been analyzed, the transition function will be:

| | 0 | 1 |
|---|---|---|
| $q_0$ | | |
| $q_1$ | $q_{12}$ | $q_2$ |
| $q_2$ | $q_{02}$ | $q_{dead}$ |
| $q_{01}$ | | |
| $q_{02}$ | $q_{02}$ | $q_1$ |
| $q_{12}$ | $q_{012}$ | $q_2$ |
| $q_{012}$ | $q_{012}$ | $q_{12}$ |
| $q_{dead}$ | $q_{dead}$ | $q_{dead}$ |

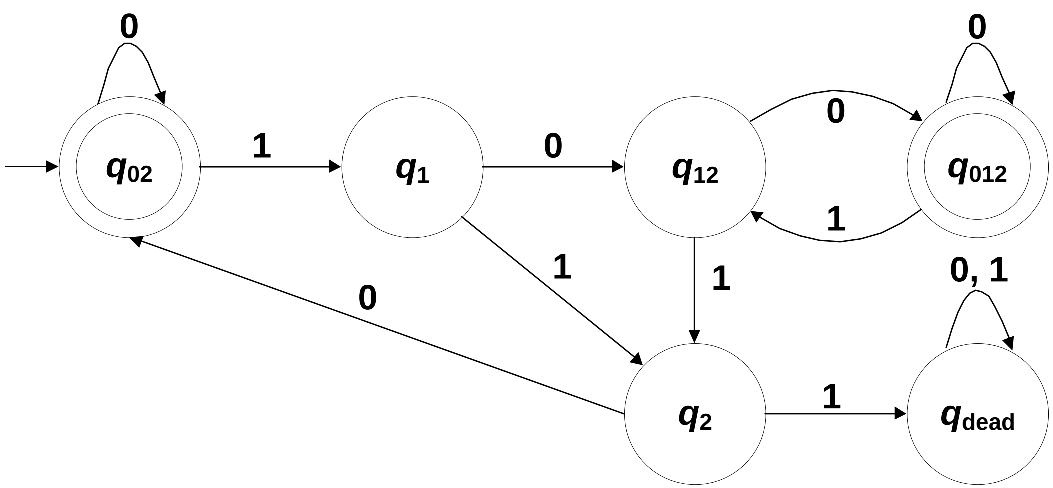
Observe that states $q_0$ and $q_{01}$ are not reachable from state $q_{02}$. These can therefore be removed from the transition table.

# NFA TO DFA CONVERSION (WITH ε - TRANSITIONS)

The final transition function will be:

| | 0 | 1 |
|---|---|---|
| $q_1$ | $q_{12}$ | $q_2$ |
| $q_2$ | $q_{02}$ | $q_{dead}$ |
| $q_{02}$ | $q_{02}$ | $q_1$ |
| $q_{12}$ | $q_{012}$ | $q_2$ |
| $q_{012}$ | $q_{012}$ | $q_{12}$ |
| $q_{dead}$ | $q_{dead}$ | $q_{dead}$ |

The state diagram of the equivalent DFA is:



# FORMAL DEFINITION OF NFA AND DFA EQUIVALENCE

- Since it has been shown that there is a procedure to construct an equivalent DFA for every NFA, it is now time to formalize the discussion on the equivalence of NFAs and DFAs.

- Theorem 1

  ***Every NFA has an equivalent DFA***.

  The proof of this theorem is trivial at this point in time since the procedure for converting any NFA to DFA has been established.

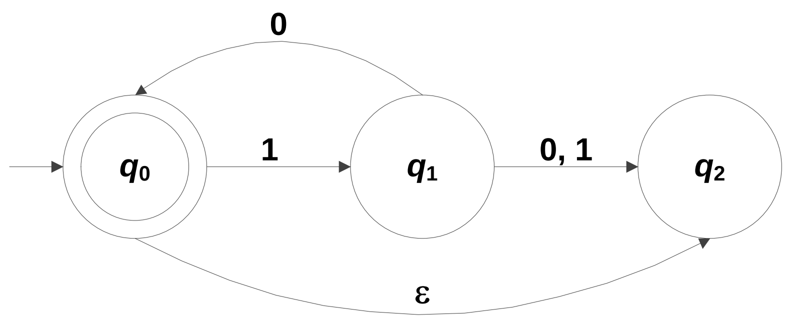
# FORMAL DEFINITION OF NFA AND DFA EQUIVALENCE

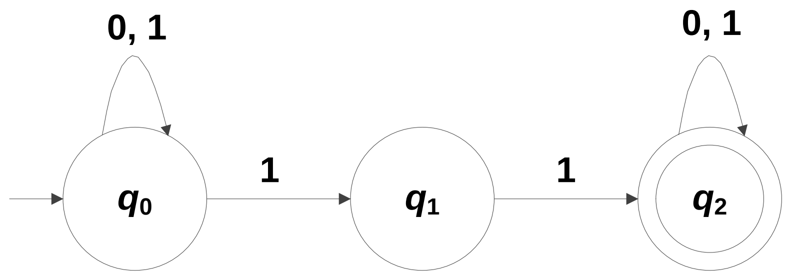
- It was mentioned previously that a language that is recognized by a DFA is called a regular language.
- Since all DFAs are NFAs and theorem 1 ascertains that every NFA has an equivalent DFA, then it can therefore be said that a language is regular if an NFA recognizes it.
- Corollary 1

  ***A language is regular if and only if some nondeterministic finite automaton recognizes it.***

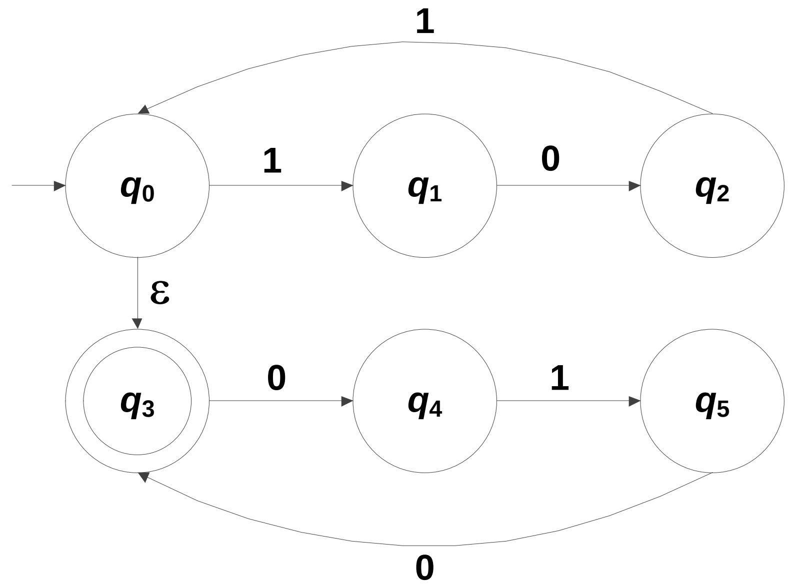
# EXERCISES

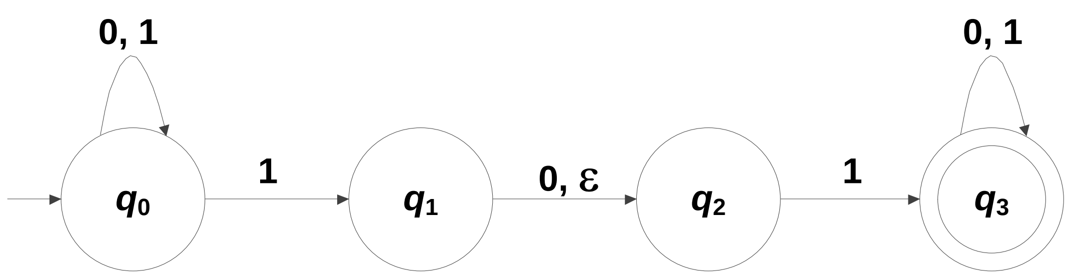
- For each of the following NFA examples, formally describe the NFA by listing down the components of each (set of states, set of symbols, the transition table, the start state, and the final state or states). Then give a generalized statement on the strings accepted by the NFA:



# EXERCISES




iACADEMY
SCHOOL OF COMPUTING • SCHOOL OF BUSINESS • SCHOOL OF DESIGN

# EXERCISES

- Give the state diagram of the following NFAs:

1. An NFA that recognizes the language $L = \{w | w$ ends with a 00$\}$. Assume that the alphabet $\Sigma = \{0, 1\}$.

2. An NFA that recognizes the language $L = \{w | w$ contains the substring 010$\}$. Assume that the alphabet $\Sigma = \{0, 1\}$.

3. An NFA that recognizes the language $L = \{w | w$ does not contain the substring 010$\}$. Assume that the alphabet $\Sigma = \{0, 1\}$.

# EXERCISES

- Construct the equivalent DFA for the following NFA: