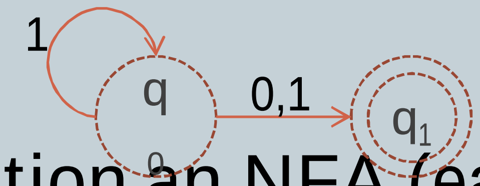
# $NFA_\varepsilon$ - NFA - DFA equivalence

## What is an NFA

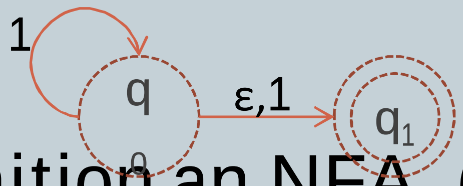
- An NFA is an automaton that its states might have none, one or more outgoing arrows under a specific symbol.



- A DFA is by definition an NFA (each state has exactly one outgoing arrow under each symbol).

# What is an NFA$_\varepsilon$

- An NFA$_\varepsilon$ is an NFA that might have ε-moves. In an ε-move we can transport from one state to the other without having any symbols.

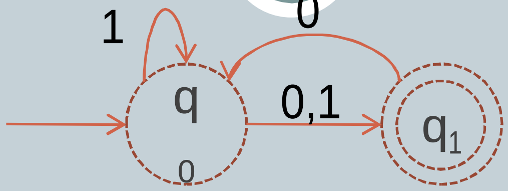


- An NFA is by definition an NFA$_\varepsilon$ (but with no ε-moves).

## NFA computation

- An NFA illustrates a machine that can have choices (just like our brains). If there are two arrows under a specific symbol it can choose either of them and follow it.
- So it works more or less like you. Suppose that you want to do something and that you can think of several methods to do it. Some of them can possibly fail but it suffices to find one that succeeds.
- An NFA accepts a string if there ***exists*** a path following arrows under the symbols of the string consecutively that takes us to an accept state.

## Example



- This automaton accepts the string 1110 because there is a path under 1110 that takes us to an accept state (the path $q_0$ , $q_0$ , $q_0$ , $q_1$), beside the fact that there are paths under 1110 that fail (for example the path $q_0$ , $q_0$ , $q_1$, $q_0$).
- However it doesn't accept the string 00 because there are no paths under 00 that can take us to $q_1$.

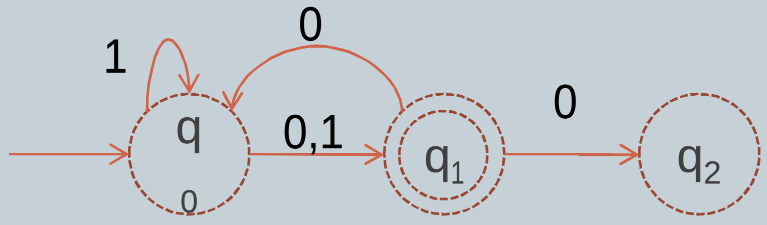
## NFA – DFA equivalence

- The language that an NFA recognizes is the set of strings which the NFA accepts.
- To see if a string is accepted it suffices to find the set of the possible states in which I can be with this string as input and see if a final state is contained in this set.

# NFA – DFA eqivalence

- Whenever an arrow is followed, there is a set of possible following states that the NFA can be. This set of states is a subset of Q.
- For example with 0010 I have the following sequence of set of states:

$$q_0 \xrightarrow{0} \{q_1\} \xrightarrow{0} \{q_0, q_2\} \xrightarrow{1} \{q_0, q_1\} \xrightarrow{0} \{q_0, q_1, q_2\}$$



## NFA – DFA equivalence

- So I only want to keep information about these subsets of states that can be reached from the initial state after following arrows.
- Since all the subsets of Q are $2^{|Q|}$ in total, this should be a finite ($<= 2^{|Q|}$) number of subsets.
- I consider each subset of states of the NFA as a state of the DFA and every subset of states containing a final state as a final state of the DFA.

## NFA – DFA equivalence

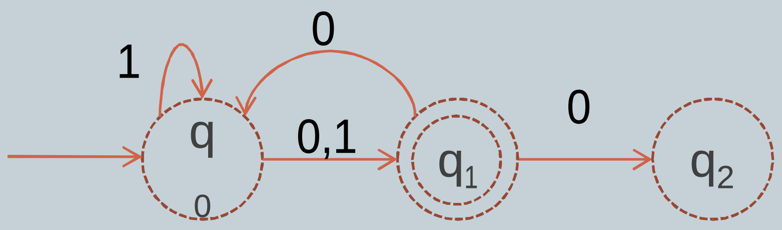
Suppose that you want to find an equivalent DFA for an NFA . The algorithm is the following:

- Start from the start state and see where 0 or 1 takes you.
- For every new subset you find, see where 0 or 1 takes you.
- Repeat until no new subsets are found.

# NFA – DFA equivalence (example)

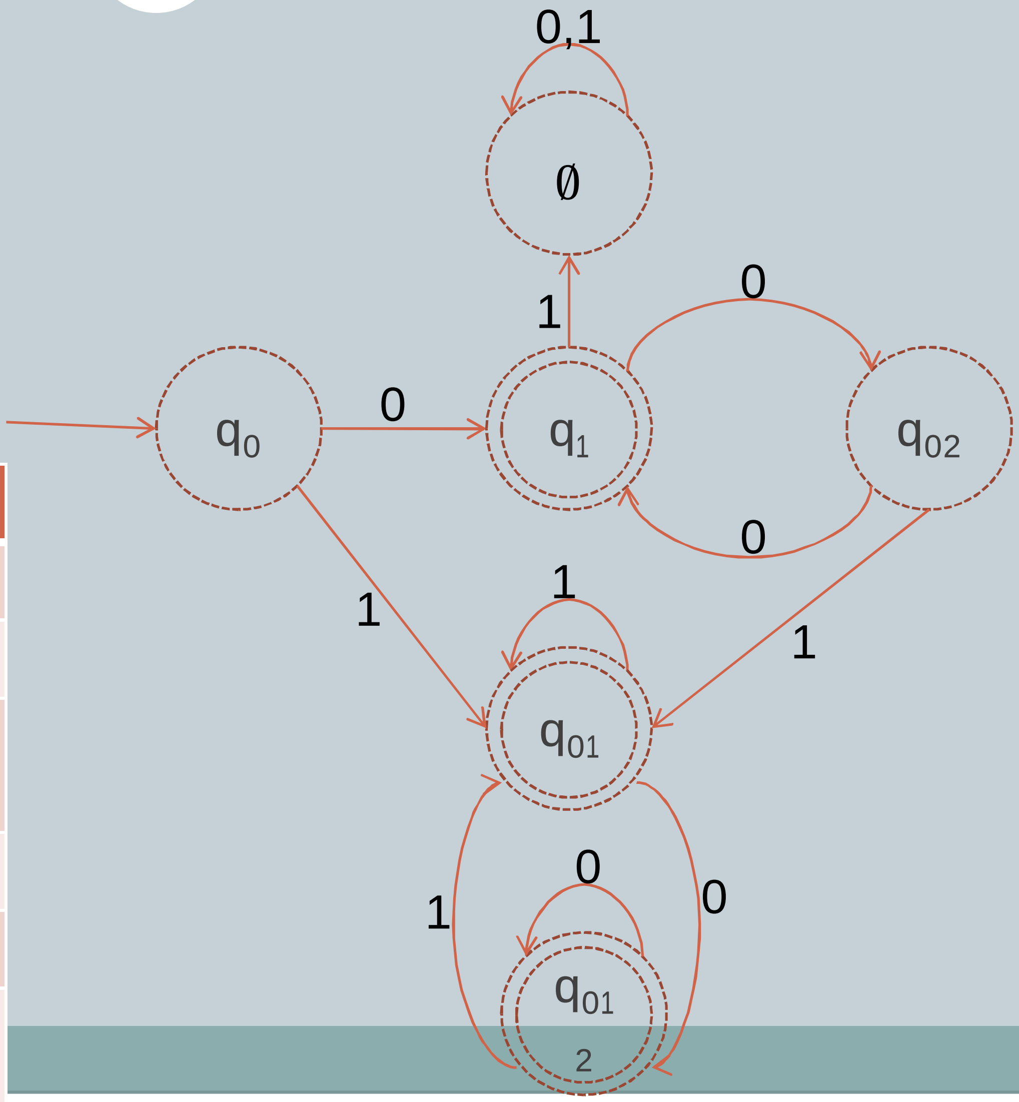
To find an equivalent DFA with the NFA of the figure I should complete the following table:

| | 0 | 1 |
|---|---|---|
| $q_0$ | $\{q_1\}$ | $\{q_0, q_1\}$ |
| $\{q_1\}$ | $\{q_0, q_2\}$ | $\emptyset$ |
| $\{q_0, q_1\}$ | $\{q_0, q_1, q_2\}$ | $\{q_0, q_1\}$ |
| $\{q_0, q_2\}$ | $\{q_1\}$ | $\{q_0, q_1\}$ |
| $\emptyset$ | $\emptyset$ | $\emptyset$ |
| $\{q_0, q_1, q_2\}$ | $\{q_0, q_1, q_2\}$ | $\{q_0, q_1\}$ |



# NFA – DFA equivalence (example)

So the DFA is:



| | 0 | 1 |
|---|---|---|
| $q_0$ | $\{q_1\}$ | $\{q_0, q_1\}$ |
| $\{q_1\}$ | $\{q_0, q_2\}$ | ∅ |
| $\{q_0, q_1\}$ | $\{q_0, q_1, q_2\}$ | $\{q_0, q_1\}$ |
| $\{q_0, q_2\}$ | $\{q_1\}$ | $\{q_0, q_1\}$ |
| ∅ | ∅ | ∅ |
| $\{q_0, q_1, q_2\}$ | $\{q_0, q_1, q_2\}$ | $\{q_0, q_1\}$ |

## $NFA_\varepsilon$ – NFA equivalence

- An NFAε is an NFA in which I can have ε-moves.
- I want to get rid of ε-moves.

## NFA$_\varepsilon$ – NFA equivalence

- Suppose that I have an $\varepsilon$-move like in the figure. Since an $\varepsilon$-move is like teleporting from $q_1$ to $q_2$ I can remove the arrow with the $\varepsilon$ and add several arrows from $q_1$ to every neighbor of $q_2$.

$$\begin{tikzcd}
 & & q_3 \\
q_1 \arrow[r, "\varepsilon"] & q_2 \arrow[ur, bend left, "0"] \arrow[r, "0"] \arrow[dr, bend right, "1"'] & q_4 \\
 & & q_5
\end{tikzcd}$$

$$\begin{tikzcd}
 & q_3 & \\
q_1 \arrow[ur, bend left, "0"] \arrow[r, "0"] \arrow[dr, bend right, "1"'] & q_4 & q_2 \arrow[ul, bend right, "0"'] \arrow[l, "0"'] \arrow[dl, bend left, "1"] \\
 & q_5 &
\end{tikzcd}$$

## NFA$_\varepsilon$ – NFA equivalence

- If from $q_1$ an ε-move takes me to q2 which is accept state then, when I remove the ε-move I should convert q1 to accept state.