NFA with ϵ-transitions

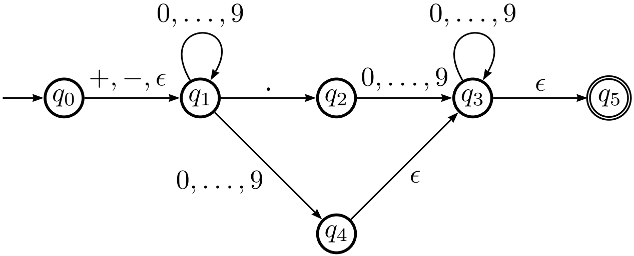
We shall now introduce another extension to NFA, called ϵ-NFA, which is a NFA whose labels can be the empty string, noted ϵ.

The interpretation of this new kind of transition, called ϵ-transition, is that the current state changes by following this transition *without reading any input*. This is sometimes referred as a **spontaneous transition**.

The rationale, i.e., the intuition behind that, is that $\epsilon a = a\epsilon = a$, so recognising ϵa or $a\epsilon$ is the same as recognising a. In other words, we do not need to read something more than a as input.

ϵ-NFA/Example

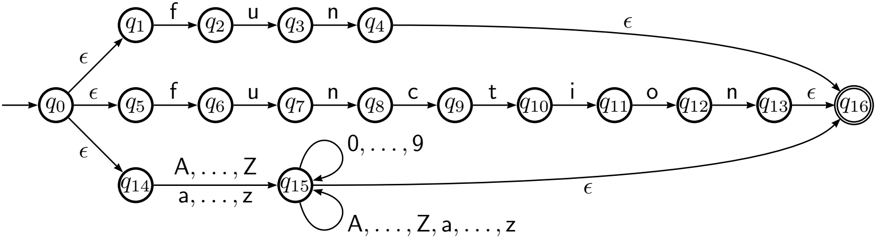
For example, we can specify signed natural and decimal numbers by means of the ϵ-NFA



This is not the simplest ϵ-NFA we can imagine for these numbers, but note the utility of the ϵ-transition between q_0 and q_1.

ϵ-NFA (cont)

In the case of compilers, ϵ-NFA allow to design separately a NFA for each token, then create an initial (respectively final) state connected to all their initial (respectively, final) states with an ϵ-transition. For instance, for keywords **fun** and **function** and identifiers:



ϵ-NFA (cont)

In order to perform a text search, once we have a single ϵ-NFA, we can

1. either remove all the ϵ-transitions and
   1.1 either create a NFA and then maybe a DFA;
   1.2 or create directly a DFA,
2. or use a formal definition of ϵ-NFA that directly leads to a recognition algorithm, just as we did for DFA and NFA.

Both methods assume that it is always possible to create an equivalent NFA, hence a DFA, from a given ϵ-NFA.

In other words, **DFA, NFA and ϵ-NFA have the same expressive power.**

ϵ-NFA (cont)

The first method constructs explicitly the NFA and maybe the DFA, while the second does not, at the possible cost of more computations at run-time.

Before entering into the details, we need to define formally an ϵ-NFA, as suggested by the second method.

The only difference between an NFA and an ϵ-NFA is that the transition function δ_E takes as second argument an element in $\Sigma \cup \{\epsilon\}$, with $\epsilon \notin \Sigma$, instead of Σ — but the alphabet still remains Σ.

ϵ-NFA/ϵ-closure

We need now a function called **ϵ-close**, which takes an ϵ-NFA $\mathcal{E}$, a state q of $\mathcal{E}$ and returns all the states which are accessible in $\mathcal{E}$ from q with label ϵ.

The idea is to achieve a **depth-first traversal** of $\mathcal{E}$, starting from q and following only ϵ-transitions.

Let us call ϵ-DFS ("ϵ-Depth-First-Search") the function such that ϵ-DFS(q, Q) is the set of states reachable from q following ϵ-transitions and which is not included in Q, *Q being interpreted as the set of states already visited in the traversal*. The set Q ensures the termination of the algorithm even in presence of cycles in the automaton. Therefore, let

$$\epsilon\text{-close}(q) = \epsilon\text{-DFS}(q, \varnothing) \quad \text{if } q \in Q_E$$

where the ϵ-NFA is $\mathcal{E} = (Q_E, \Sigma, \delta_E, q_0, F_E)$.

ϵ-NFA/ϵ-closure (cont)

Now we define ϵ-DFS as follows:

$$\epsilon\text{-DFS}(q, Q) = \varnothing \qquad \text{if } q \in Q \qquad (3)$$

$$\epsilon\text{-DFS}(q, Q) = \{q\} \cup \bigcup_{p \in \delta_E(q,\epsilon)} \epsilon\text{-DFS}(p, Q \cup \{q\}) \qquad \text{if } q \notin Q \qquad (4)$$

The ϵ-NFA page 139 leads to the following ϵ-closures:

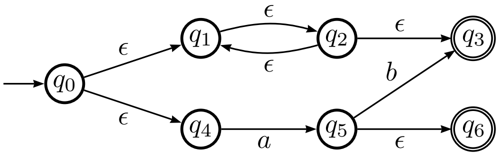
$$\epsilon\text{-close}(q_1) = \{q_1\} \qquad \epsilon\text{-close}(q_2) = \{q_2\}$$

$$\epsilon\text{-close}(q_0) = \{q_0, q_1\} \qquad \epsilon\text{-close}(q_3) = \{q_3, q_5\}$$

$$\epsilon\text{-close}(q_5) = \{q_5\} \qquad \epsilon\text{-close}(q_4) = \{q_4, q_3, q_5\}$$

ϵ-NFA/ϵ-closure (cont)

Consider, as a more difficult example, the following ϵ-NFA $\mathcal{E}$:



$$
\begin{aligned}
\epsilon\text{-close}(q_0) &= \epsilon\text{-DFS}(q_0, \varnothing) && \text{since } q_0 \in Q_E \\
&= \{q_0\} \cup \epsilon\text{-DFS}(q_1, \{q_0\}) \cup \epsilon\text{-DFS}(q_4, \{q_0\}) && \text{by eq. 4} \\
&= \{q_0\} \cup \left(\{q_1\} \cup \bigcup_{p \in \delta_E(q_1,\epsilon)} \epsilon\text{-DFS}(p, \{q_0, q_1\})\right) && \text{by eq. 4} \\
&\qquad \cup \left(\{q_4\} \cup \bigcup_{p \in \delta_E(q_4,\epsilon)} \epsilon\text{-DFS}(p, \{q_0, q_4\})\right) && \text{by eq. 4}
\end{aligned}
$$

ϵ-NFA/ϵ-closure (cont)

$$
\begin{aligned}
\epsilon\text{-close}(q_0) &= \{q_0\} \cup \left(\{q_1\} \cup \bigcup_{p \in \{q_2\}} \epsilon\text{-DFS}(p, \{q_0, q_1\})\right) \\
&\qquad \cup \left(\{q_4\} \cup \bigcup_{p \in \emptyset} \epsilon\text{-DFS}(p, \{q_0, q_4\})\right) \\
&= \{q_0\} \cup (\{q_1\} \cup \epsilon\text{-DFS}(q_2, \{q_0, q_1\})) \cup (\{q_4\} \cup \emptyset) \\
&= \{q_0, q_1, q_4\} \cup \epsilon\text{-DFS}(q_2, \{q_0, q_1\}) \\
&= \{q_0, q_1, q_4\} \cup \left(\{q_2\} \cup \bigcup_{p \in \delta_E(q_2, \epsilon)} \epsilon\text{-DFS}(p, \{q_0, q_1, q_2\})\right)
\end{aligned}
$$

ϵ-NFA/ϵ-closure (cont)

$$
\begin{aligned}
\epsilon\text{-close}(q_0) &= \{q_0, q_1, q_4\} \cup \left(\{q_2\} \cup \bigcup_{p \in \{q_1, q_3\}} \epsilon\text{-DFS}(p, \{q_0, q_1, q_2\})\right) \\
&= \{q_0, q_1, q_2, q_4\} \cup \epsilon\text{-DFS}(q_1, \{q_0, q_1, q_2\}) \\
&\qquad\qquad\qquad\quad \cup\, \epsilon\text{-DFS}(q_3, \{q_0, q_1, q_2\}) \\
&= \{q_0, q_1, q_2, q_4\} \cup \varnothing \qquad\qquad \text{by eq. 3, since } q_1 \in \{q_0, q_1, q_2\} \\
&\qquad \cup \left(\{q_3\} \cup \bigcup_{p \in \delta_E(q_3, \epsilon)} \epsilon\text{-DFS}(p, \{q_0, q_1, q_2, q_3\})\right) \quad \text{by eq. 4} \\
&= \{q_0, q_1, q_2, q_3, q_4\} \cup \bigcup_{p \in \varnothing} \epsilon\text{-DFS}(p, \{q_0, q_1, q_2, q_3\}) \\
&= \{q_0, q_1, q_2, q_3, q_4\}
\end{aligned}
$$

ϵ-NFA/ϵ-closure (cont)

It is useful to extend ϵ-close to sets of states, not just states.

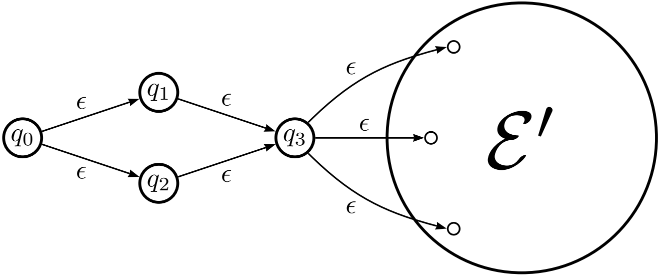
Let us note $\overline{\epsilon\text{-close}}$ this extension, which we can easily define as

$$\overline{\epsilon\text{-close}}(Q) = \bigcup_{q \in Q} \epsilon\text{-close}(q)$$

for any subset $Q \subseteq Q_E$ where the ϵ-NFA is $\mathcal{E} = (Q_E, \Sigma, \delta_E, q_E, F_E)$.

ϵ-NFA/ϵ-closure/Optimisation

Compute the ϵ-closure of q_0 in the following ϵ-NFA $\mathcal{E}$:



where the sub-ϵ-NFA $\mathcal{E}'$ contains only ϵ-transitions and all its Q' states are accessible from q_3.

ϵ-NFA/ϵ-closure/Optimisation (cont)

$$
\begin{aligned}
\epsilon\text{-close}(q_0) &= \epsilon\text{-DFS}(q_0, \varnothing) \\
&= \{q_0\} \cup \epsilon\text{-DFS}(q_1, \{q_0\}) \cup \epsilon\text{-DFS}(q_2, \{q_0\}) \\
&= \{q_0\} \cup (\{q_1\} \cup \epsilon\text{-DFS}(q3, \{q_0, q_1\})) \\
&\qquad\quad \cup (\{q_2\} \cup \epsilon\text{-DFS}(q_3, \{q_0, q_2\})) \\
&= \{q_0, q_1, q_2\} \cup \epsilon\text{-DFS}(q3, \{q_0, q_1\}) \cup \epsilon\text{-DFS}(q_3, \{q_0, q_2\}) \\
&= \{q_0, q_1, q_2, q_3, \} \cup (\{q_3\} \cup Q') \cup (\{q_3\} \cup Q') \\
&= \{q_0, q_1, q_2, q_3, \} \cup Q'
\end{aligned}
$$

We compute $\{q_3\} \cup Q'$ two times, that is, we traverse two times q_3 and all the states of $\mathcal{E}'$, which can be inefficient if Q' is big.

ϵ-NFA/ϵ-closure/Optimisation (cont)

The way to avoid duplicating traversals is to change the definitions of ϵ-close and $\overline{\epsilon\text{-close}}$.
Dually, we need a new definition of ϵ-DFS and create function $\overline{\epsilon\text{-DFS}}$ which is similar to ϵ-DFS except that it applies to set of states instead of one state:

$$
\begin{aligned}
\epsilon\text{-close}(q) &= \epsilon\text{-DFS}(q, \varnothing) && \text{if } q \in Q_E \\
\overline{\epsilon\text{-close}}(Q) &= \overline{\epsilon\text{-DFS}}(Q, \varnothing) && \text{if } Q \subseteq Q_E
\end{aligned}
$$

We interpret Q' in ϵ-DFS(q, Q') and $\overline{\epsilon\text{-DFS}}(Q, Q')$ as the set of states that have already been visited in the depth-first search.

Variables q and Q denote, respectively, a state and a set of states that have to be explored.

ϵ-NFA/ϵ-closure/Optimisation (cont)

In the first definition we computed the *new reachable states*, but in the new one we compute the *currently reached states*. Then let us redefine ϵ-DFS this way:

$$\epsilon\text{-DFS}(q, Q') = Q' \qquad \text{if } q \in Q' \qquad (1')$$

$$\epsilon\text{-DFS}(q, Q') = \overline{\epsilon\text{-DFS}}(\delta_E(q, \epsilon), Q' \cup \{q\}) \qquad \text{if } q \notin Q' \qquad (2')$$

Contrast with the first definition

$$\epsilon\text{-DFS}(q, Q') = \varnothing \qquad \text{if } q \in Q' \qquad (1)$$

$$\epsilon\text{-DFS}(q, Q') = \{q\} \cup \bigcup_{p \in \delta_E(q,\epsilon)} \epsilon\text{-DFS}(p, Q' \cup \{q\}) \qquad \text{if } q \notin Q' \qquad (2)$$

Hence, in (1) we return $\varnothing$ because there is no new state, i.e., none not already in Q', whereas in (1') we return Q' itself.

ϵ-NFA/ϵ-closure/Optimisation (cont)

The new definition of $\overline{\epsilon\text{-DFS}}$ is not more difficult than the first one:

$$\overline{\epsilon\text{-DFS}}(\varnothing, Q') = Q' \tag{5}$$

$$\overline{\epsilon\text{-DFS}}(\{q\} \cup Q, Q') = \overline{\epsilon\text{-DFS}}(Q, \epsilon\text{-DFS}(q, Q')) \qquad \text{if } q \notin Q \tag{6}$$

Notice that the definitions of ϵ-DFS and $\overline{\epsilon\text{-DFS}}$ are **mutually recursive**, i.e., they depend on each other.

In (2) we traverse states in *parallel* (consider the union operator), starting from each element in $\delta_E(q, \epsilon)$, whereas in (2') and (6), we traverse them *sequentially* so we can use the information collected (currently reached states) in the previous searches.

ϵ-NFA/ϵ-closure/Optimisation (cont)

Coming back to our example page 149, we find

$$
\begin{aligned}
\epsilon\text{-close}(q_0) &= \epsilon\text{-DFS}(q_0, \varnothing) && q_0 \in Q_E \\
&= \overline{\epsilon\text{-DFS}}(\{q_1, q_2\}, \{q_0\}) && \text{by eq. (2')} \\
&= \overline{\epsilon\text{-DFS}}(\{q_2\}, \epsilon\text{-DFS}(q_1, \{q_0\})) && \text{by eq. (4)} \\
&= \overline{\epsilon\text{-DFS}}(\{q_2\}, \overline{\epsilon\text{-DFS}}(\{q_3\}, \{q_0, q_1\})) && \text{by eq. (2')} \\
&= \overline{\epsilon\text{-DFS}}(\{q_2\}, \overline{\epsilon\text{-DFS}}(\varnothing, \epsilon\text{-DFS}(q_3, \{q_0, q_1\}))) && \text{by eq. (4)} \\
&= \overline{\epsilon\text{-DFS}}(\{q_2\}, \epsilon\text{-DFS}(q_3, \{q_0, q_1\})) && \text{by eq. (3)} \\
&= \overline{\epsilon\text{-DFS}}(\{q_2\}, \{q_0, q_1, q_3\} \cup Q') \\
&= \overline{\epsilon\text{-DFS}}(\varnothing, \epsilon\text{-DFS}(q_2, \{q_0, q_1, q_3\} \cup Q')) && \text{by eq. (4)}
\end{aligned}
$$

ϵ-NFA/ϵ-closure/Optimisation (cont)

$$
\begin{aligned}
\epsilon\text{-close}(q_0) &= \epsilon\text{-DFS}(q_2, \{q_0, q_1, q_3\} \cup Q') && \text{by eq. (3)}\\
&= \overline{\epsilon\text{-DFS}}(\{q_3\}, \{q_0, q_1, q_2, q_3\} \cup Q') && \text{by eq. (2')}\\
&= \overline{\epsilon\text{-DFS}}(\varnothing, \epsilon\text{-DFS}(q_3, \{q_0, q_1, q_2, q_3\} \cup Q')) && \text{by eq. (4)}\\
&= \epsilon\text{-DFS}(q_3, \{q_0, q_1, q_2, q_3\} \cup Q') && \text{by eq. (3)}\\
&= \{q_0, q_1, q_2, q_3\} \cup Q' && \text{by eq. (1')}
\end{aligned}
$$

The important thing here is that we did not compute (traverse) several times Q'. Note that some equations can be used in a different order and q can be chosen arbitrarily in equation (4), but the result is always the same.

Extended transition functions for ϵ-NFAs

The ϵ-closure allows to explain how a ϵ-NFA recognises or rejects a given input word.

In (2) we traverse states in *parallel* (consider the union operator), starting from each element in $\delta_E(q, \epsilon)$, whereas in (2') and (6), we traverse them *sequentially* so we can use the information collected (currently reached states) in the previous searches.

ϵ-NFA/ϵ-closure/Optimisation (cont)

Coming back to our example page 149, we find

$$
\begin{aligned}
\epsilon\text{-close}(q_0) &= \epsilon\text{-DFS}(q_0, \varnothing) && q_0 \in Q_E \\
&= \overline{\epsilon\text{-DFS}}(\{q_1, q_2\}, \{q_0\}) && \text{by eq. (2')} \\
&= \overline{\epsilon\text{-DFS}}(\{q_2\}, \epsilon\text{-DFS}(q_1, \{q_0\})) && \text{by eq. (4)} \\
&= \overline{\epsilon\text{-DFS}}(\{q_2\}, \overline{\epsilon\text{-DFS}}(\{q_3\}, \{q_0, q_1\})) && \text{by eq. (2')} \\
&= \overline{\epsilon\text{-DFS}}(\{q_2\}, \overline{\epsilon\text{-DFS}}(\varnothing, \epsilon\text{-DFS}(q_3, \{q_0, q_1\}))) && \text{by eq. (4)} \\
&= \overline{\epsilon\text{-DFS}}(\{q_2\}, \epsilon\text{-DFS}(q_3, \{q_0, q_1\})) && \text{by eq. (3)} \\
&= \overline{\epsilon\text{-DFS}}(\{q_2\}, \{q_0, q_1, q_3\} \cup Q') \\
&= \overline{\epsilon\text{-DFS}}(\varnothing, \epsilon\text{-DFS}(q_2, \{q_0, q_1, q_3\} \cup Q')) && \text{by eq. (4)}
\end{aligned}
$$

ϵ-NFA/ϵ-closure/Optimisation (cont)

$$
\begin{aligned}
\epsilon\text{-close}(q_0) &= \epsilon\text{-DFS}(q_2, \{q_0, q_1, q_3\} \cup Q') && \text{by eq. (3)}\\
&= \overline{\epsilon\text{-DFS}}(\{q_3\}, \{q_0, q_1, q_2, q_3\} \cup Q') && \text{by eq. (2')}\\
&= \overline{\epsilon\text{-DFS}}(\varnothing, \epsilon\text{-DFS}(q_3, \{q_0, q_1, q_2, q_3\} \cup Q')) && \text{by eq. (4)}\\
&= \epsilon\text{-DFS}(q_3, \{q_0, q_1, q_2, q_3\} \cup Q') && \text{by eq. (3)}\\
&= \{q_0, q_1, q_2, q_3\} \cup Q' && \text{by eq. (1')}
\end{aligned}
$$

The important thing here is that we did not compute (traverse) several times Q'. Note that some equations can be used in a different order and q can be chosen arbitrarily in equation (4), but the result is always the same.

Extended transition functions for ϵ-NFAs

The ϵ-closure allows to explain how a ϵ-NFA recognises or rejects a given input word.

In (2) we traverse states in *parallel* (consider the union operator), starting from each element in $\delta_E(q, \epsilon)$, whereas in (2') and (6), we traverse them *sequentially* so we can use the information collected (currently reached states) in the previous searches.

ϵ-NFA/ϵ-closure/Optimisation (cont)

Coming back to our example page 149, we find

$$
\begin{aligned}
\epsilon\text{-close}(q_0) &= \epsilon\text{-DFS}(q_0, \varnothing) && q_0 \in Q_E \\
&= \overline{\epsilon\text{-DFS}}(\{q_1, q_2\}, \{q_0\}) && \text{by eq. (2')} \\
&= \overline{\epsilon\text{-DFS}}(\{q_2\}, \epsilon\text{-DFS}(q_1, \{q_0\})) && \text{by eq. (4)} \\
&= \overline{\epsilon\text{-DFS}}(\{q_2\}, \overline{\epsilon\text{-DFS}}(\{q_3\}, \{q_0, q_1\})) && \text{by eq. (2')} \\
&= \overline{\epsilon\text{-DFS}}(\{q_2\}, \overline{\epsilon\text{-DFS}}(\varnothing, \epsilon\text{-DFS}(q_3, \{q_0, q_1\}))) && \text{by eq. (4)} \\
&= \overline{\epsilon\text{-DFS}}(\{q_2\}, \epsilon\text{-DFS}(q_3, \{q_0, q_1\})) && \text{by eq. (3)} \\
&= \overline{\epsilon\text{-DFS}}(\{q_2\}, \{q_0, q_1, q_3\} \cup Q') \\
&= \overline{\epsilon\text{-DFS}}(\varnothing, \epsilon\text{-DFS}(q_2, \{q_0, q_1, q_3\} \cup Q')) && \text{by eq. (4)}
\end{aligned}
$$

ϵ-NFA/ϵ-closure/Optimisation (cont)

$$
\begin{aligned}
\epsilon\text{-close}(q_0) &= \epsilon\text{-DFS}(q_2, \{q_0, q_1, q_3\} \cup Q') && \text{by eq. (3)}\\
&= \overline{\epsilon\text{-DFS}}(\{q_3\}, \{q_0, q_1, q_2, q_3\} \cup Q') && \text{by eq. (2')}\\
&= \overline{\epsilon\text{-DFS}}(\varnothing, \epsilon\text{-DFS}(q_3, \{q_0, q_1, q_2, q_3\} \cup Q')) && \text{by eq. (4)}\\
&= \epsilon\text{-DFS}(q_3, \{q_0, q_1, q_2, q_3\} \cup Q') && \text{by eq. (3)}\\
&= \{q_0, q_1, q_2, q_3\} \cup Q' && \text{by eq. (1')}
\end{aligned}
$$

The important thing here is that we did not compute (traverse) several times Q'. Note that some equations can be used in a different order and q can be chosen arbitrarily in equation (4), but the result is always the same.

Extended transition functions for ϵ-NFAs

The ϵ-closure allows to explain how a ϵ-NFA recognises or rejects a given input word.

Let $\mathcal{E} = (Q_E, \Sigma, \delta_E, q_0, F_E)$.

We want $\hat{\delta}_E(q, w)$ be the set of states reachable from q along a path whose labels, when concatenated, for the string w. The difference here with NFA's is that several ϵ can be present along this path, despite not contributing to w. For all state $q \in Q_E$, let

$$\hat{\delta}_E(q, \epsilon) = \epsilon\text{-close}(q)$$

$$\hat{\delta}_E(q, wa) = \overline{\epsilon\text{-close}}\left(\bigcup_{p \in \hat{\delta}_E(q,w)} \delta_N(p, a)\right) \qquad \text{for all } a \in \Sigma, w \in \Sigma^*$$

This definition is based on the regular identity $wa = ((w\epsilon^*)a)\epsilon^*$.

Extended transition functions for ϵ-NFAs/Example

Let us consider again the ϵ-NFA recognising natural and decimal numbers, at page 139, and compute the states reached on the input `5.6`:

$$
\begin{aligned}
\hat{\delta}_E(q_0, \epsilon) &= \epsilon\text{-close}(q_0) = \{q_0, q_1\} \\
\hat{\delta}_E(q_0, 5) &= \overline{\epsilon\text{-close}}\left(\bigcup_{p \in \hat{\delta}_E(q_0,\epsilon)} \delta_N(p, 5) \right) \\
&= \overline{\epsilon\text{-close}}(\delta_N(q_0, 5) \cup \delta_N(q_1, 5)) = \overline{\epsilon\text{-close}}(\varnothing \cup \{q_1, q_4\}) \\
&= \{q_1, q_3, q_4, q_5\} \\
\hat{\delta}_E(q_0, 5.) &= \overline{\epsilon\text{-close}}\left(\bigcup_{p \in \hat{\delta}_N(q_0,5)} \delta_N(p, .) \right) \\
&= \overline{\epsilon\text{-close}}(\delta_N(q_1, .) \cup \delta_N(q_3, .) \cup \delta_N(q_4, .) \cup \delta_N(q_5, .))
\end{aligned}
$$

Extended transition functions for ϵ-NFAs/Example (cont)

$$
\begin{aligned}
\hat{\delta}_E(q_0, 5.) &= \overline{\epsilon\text{-close}}(\{q_2\} \cup \varnothing \cup \varnothing \cup \varnothing) = \{q_2\} \\
\hat{\delta}_N(q_0, 5.6) &= \overline{\epsilon\text{-close}}\left(\bigcup_{p \in \hat{\delta}_E(q_0, 5.)} \delta_N(p, 6) \right) \\
&= \overline{\epsilon\text{-close}}(\delta_N(q_2, 6)) \\
&= \overline{\epsilon\text{-close}}(\{q_3\}) \\
&= \{q_3, q_5\} \ni q_5
\end{aligned}
$$

Since q_5 is a final state, the string `5.6` is recognised as a number.

Subset construction for ϵ-NFAs

Let us present now how to construct a DFA from a ϵ-NFA such that both recognise the same language.

The method is a variation of the subset construction we presented for NFA: we must take into account the states reachable through ϵ-transitions, with help of ϵ-closures.

Subset construction for ϵ-NFAs (cont)

Assume that $\mathcal{E} = (Q, \Sigma, \delta, q_0, F)$ is an ϵ-NFA. Let us define as follows the equivalent DFA $\mathcal{D} = (Q_D, \Sigma, \delta_D, q_D, F_D)$.

1. Q_D is the set of subsets of Q_E. More precisely, all accessible states of $\mathcal{D}$ are ϵ-closed subsets of Q_E, i.e., sets $Q \subseteq Q_E$ such that $Q = \epsilon\text{-close}(Q)$.
2. $q_D = \epsilon\text{-close}(q_0)$, i.e., we get the start state of $\mathcal{D}$ by ϵ-closing the set made of only the start state of $\mathcal{E}$.
3. F_D is those sets of states that contain at least one final state of $\mathcal{E}$, that is to say $F_D = \{Q \mid Q \in Q_D \text{ and } Q \cap F_E \neq \varnothing\}$.
4. For all $a \in \Sigma$ and $Q \in Q_D$, let $\delta_D(Q, a) = \overline{\epsilon\text{-close}}\left(\bigcup_{q \in Q} \delta_E(q, a)\right)$

Subset construction for ϵ-NFAs/Example

Let us consider again the ϵ-NFA page 139. Its transition table is

$\mathcal{E}$	$+$	$-$	$0,\ldots,9$	$.$	ϵ
$\rightarrow q_0$	$\{q_1\}$	$\{q_1\}$	$\varnothing$	$\varnothing$	$\{q_1\}$
q_1	$\varnothing$	$\varnothing$	$\{q_1, q_4\}$	$\{q_2\}$	$\varnothing$
q_2	$\varnothing$	$\varnothing$	$\{q_3\}$	$\varnothing$	$\varnothing$
q_3	$\varnothing$	$\varnothing$	$\{q_3\}$	$\varnothing$	$\{q_5\}$
q_4	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\{q_3\}$
$\# q_5$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$

Subset construction for ϵ-NFAs/Example (cont)

By applying the subset construction to this ϵ-NFA, we get the table

$\mathcal{D}$	$+$	$-$	$0,\ldots,9$	.
$\rightarrow\{q_0, q_1\}$	$\{q_1\}$	$\{q_1\}$	$\{q_1, q_3, q_4, q_5\}$	$\{q_2\}$
$\{q_1\}$	$\varnothing$	$\varnothing$	$\{q_1, q_3, q_4, q_5\}$	$\{q_2\}$
$\#\{q_1, q_3, q_4, q_5\}$	$\varnothing$	$\varnothing$	$\{q_1, q_3, q_4, q_5\}$	$\{q_2\}$
$\{q_2\}$	$\varnothing$	$\varnothing$	$\{q_3, q_5\}$	$\varnothing$
$\#\{q_3, q_5\}$	$\varnothing$	$\varnothing$	$\{q_3, q_5\}$	$\varnothing$

Subset construction for ϵ-NFAs/Example (cont)

Let us rename the states of $\mathcal{D}$ and get rid of the empty sets:

$\mathcal{D}$	$+$	$-$	$0, \ldots, 9$	.
$\rightarrow A$	B	B	C	D
B			C	D
$\#C$			C	D
D			E	
$\#E$			E	

Subset construction for ϵ-NFAs/Example (cont)

The transition diagram of $\mathcal{D}$ is therefore