Discrete Mathematics

Lecture 22

Finite Automata

2024-02-09

We will be making use of mathematical models of physical systems called *finite state machines* – of which there are many different varieties. Here we use one particular sort, **finite automata** (singular: finite automaton), to recognise whether or not a string is in a particular language. The key features of this abstract notion of machine are as follows and are illustrated by the example on Slide 44.

- There are only finitely many different **states** that a finite automaton can be in. In the example there are four states, labelled q_0, q_1, q_2, and q_3.
- We do not care at all about the internal structure of machine states. All we care about is which **transitions** the machine can make between the states. A symbol from some fixed alphabet Σ is associated with each transition: we think of the elements of Σ as **input symbols**. Thus all the possible transitions of the finite automaton can be specified by giving a finite graph whose vertices are the states and whose edges have both a direction and a label (an element of Σ). In the example $\Sigma = \{a, b\}$ and the only possible transitions from state q_1 are

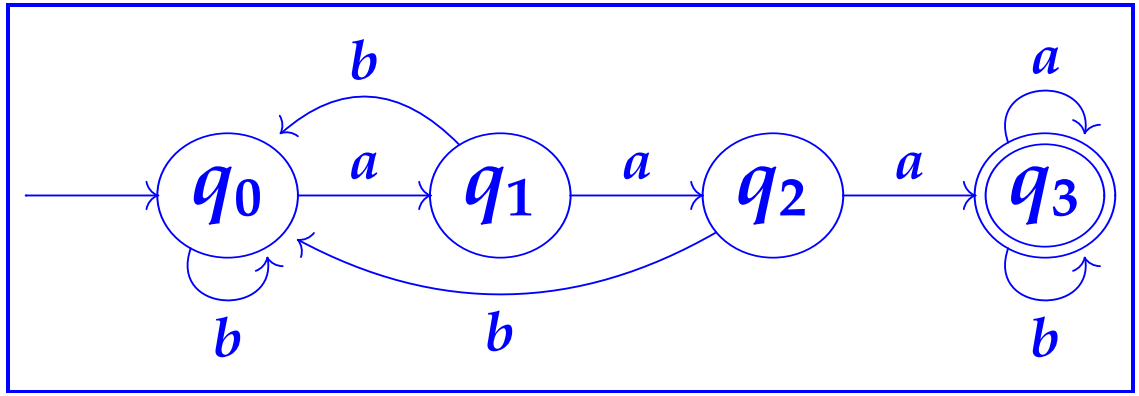
  $$q_1 \xrightarrow{b} q_0 \quad \text{and} \quad q_1 \xrightarrow{a} q_2.$$

  In other words, in state q_1 the machine can either input the symbol b and enter state q_0, or it can input the symbol a and enter state q_2. (Note that transitions from a state back to the same state are allowed: e.g. $q_3 \xrightarrow{a} q_3$ in the example.)
- There is a distinguished **start state** (also known as the **initial state**). In the example it is q_0. In the graphical representation of a finite automaton, the start state is usually indicated by means of a unlabelled arrow.

- The states are partitioned into two kinds: **accepting states** (also know as **final states**) and non-accepting states. In the graphical representation of a finite automaton, the accepting states are indicated by double circles round the name of each such state, and the non-accepting states are indicated using single circles. In the example there is only one accepting state, q_3; the other three states are non-accepting. (The two extreme possibilities that *all* states are accepting, or that *no* states are accepting, are allowed; it is also allowed for the start state to be accepting.)

The reason for the partitioning of the states of a finite automaton into 'accepting' and 'non-accepting' has to do with the use to which one puts finite automata—namely to recognise whether or not a string $u \in \Sigma^*$ is in a particular language (= subset of Σ^*). Given u we begin in the start state of the automaton and traverse its graph of transitions, using up the symbols in u in the correct order reading the string from left to right. If we can use up all the symbols in u in this way and reach an accepting state, then u is in the language 'accepted' (or 'recognised') by this particular automaton. On the other hand, if there is no path in the graph of transitions from the start state to some accepting state with string of labels equal to u, then u is not in the language accepted by the automaton. This is summed up on Slide 45.

Example of a finite automaton

$M \triangleq$ 

- set of states: $\{q_0, q_1, q_2, q_3\}$
- input alphabet: $\{a, b\}$
- transitions, labelled by input symbols: as indicated by the above directed graph
- start state: q_0
- accepting state(s): q_3

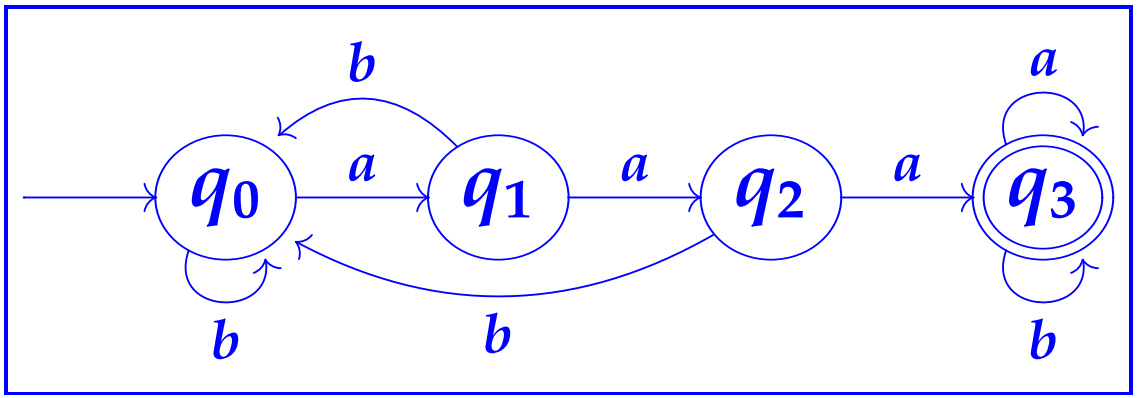
Language accepted by a finite automaton M

- Look at paths in the transition graph from the start state to *some* accepting state.
- Each such path gives a string of input symbols, namely the string of labels on each transition in the path.
- The set of all such strings is by definition **the language accepted by** M, written $L(M)$.

Notation: write $q \xrightarrow{u}{}^{*} q'$ to mean that in the automaton there is a path from state q to state q' whose labels form the string u.

(**N.B.** $q \xrightarrow{\varepsilon}{}^{*} q'$ means $q = q'$.)

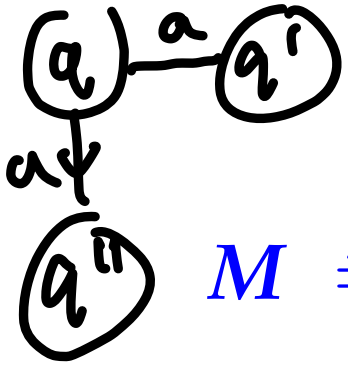
Example of an accepted language

$M \triangleq$ 

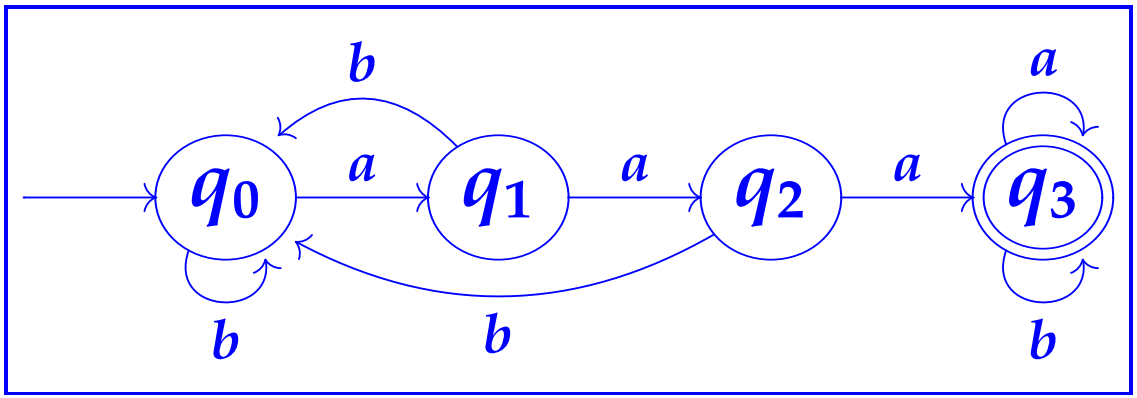
For example

- $aaab \in L(M)$, because $q_0 \xrightarrow{aaab}{}^* q_3$
- $abaa \notin L(M)$, because $\forall q(q_0 \xrightarrow{abaa}{}^* q \Leftrightarrow q = q_2)$

Example of an accepted language



$M \triangleq$



Claim:

$$L(M) = L((a|b)^*aaa(a|b)^*)$$

set of all strings matching the regular expression $(a|b)^*aaa(a|b)^*$

(q_i (for $i = 0, 1, 2$) represents the state in the process of reading a string in which the last i symbols read were all as)

Determinism and non-determinism

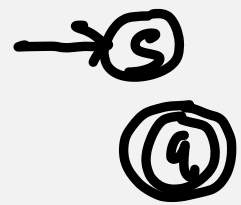
Slide 49 gives a formal definition of the notion of finite automaton. The reason for the qualification 'non-deterministic' is because in general, for each state $q \in Q$ and each input symbol $a \in \Sigma$, there may be no, one, or many states that can be reached in a single transition labelled a from q; see the example on Slide 50.

We single out as particularly important the case when there is always exactly one next state for a given input symbol in any given state and call such automata **deterministic**: see Slide 51. The finite automaton pictured on Slide 52 is deterministic. But note that if we took the same graph of transitions but insisted that the alphabet of input symbols was $\{a, b, c\}$ say, then we have specified an NFA not a DFA – see Slide 53. The moral of this is: *when specifying an NFA, as well as giving the graph of state transitions, it is important to say what is the alphabet of input symbols* (because some input symbols may not appear in the graph at all).

Non-deterministic finite automaton (NFA)

is by definition a 5-tuple $M = (Q, \Sigma, \Delta, s, F)$, where:

- Q is a finite set (of **states**)
- Σ is a finite set (the alphabet of **input symbols**)
- Δ is a subset of $Q \times \Sigma \times Q$ (the **transition relation**)
- s is an element of Q (the **start state**)
- F is a subset of Q (the **accepting states**)



Notation: write "$q \xrightarrow{a} q'$ in M" to mean $(q, a, q') \in \Delta$.

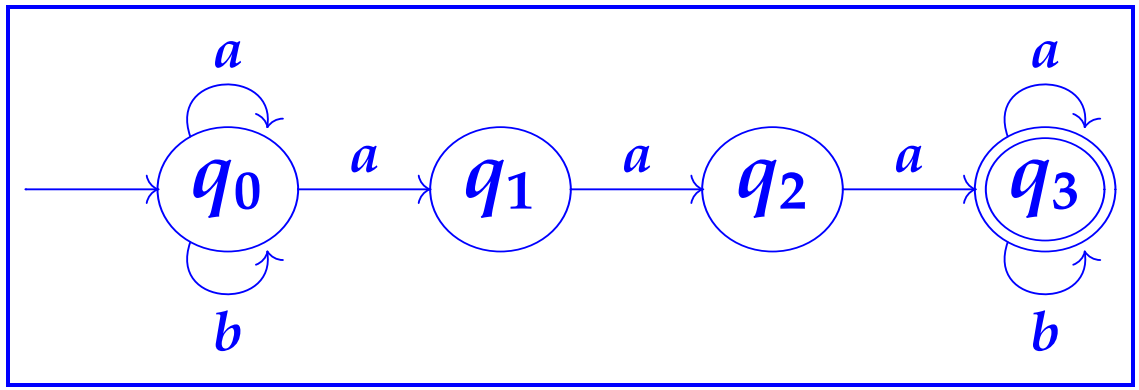
$\Delta : Q \times \Sigma \to Q$

$(q, a)\ \Delta\ q' \iff$

Example of an NFA

Input alphabet: $\{a, b\}$.

States, transitions, start state, and accepting states as shown:



For example $\{q \mid q_1 \xrightarrow{a} q\} = \{q_2\}$

$\{q \mid q_1 \xrightarrow{b} q\} = \emptyset$

$\{q \mid q_0 \xrightarrow{a} q\} = \{q_0, q_1\}$.

The language accepted by this automaton is the same as for the automaton on Slide 44, namely $\{u \in \{a, b\}^* \mid u$ contains three consecutive a's$\}$.

Deterministic finite automaton (DFA)

A **deterministic finite automaton** (DFA) is an NFA $M = (Q, \Sigma, \Delta, s, F)$ with the property that for each state $q \in Q$ and each input symbol $a \in \Sigma_M$, there is a unique state $q' \in Q$ satisfying $q \xrightarrow{a} q'$.

In a DFA $\Delta \subseteq Q \times \Sigma \times Q$ is the graph of a function $Q \times \Sigma \to Q$, which we write as δ and call the **next-state function**.

Thus for each (state, input symbol)-pair (q, a), $\delta(q, a)$ is the unique state that can be reached from q by a transition labelled a:

$$\forall q'(q \xrightarrow{a} q' \Leftrightarrow q' = \delta(q, a))$$

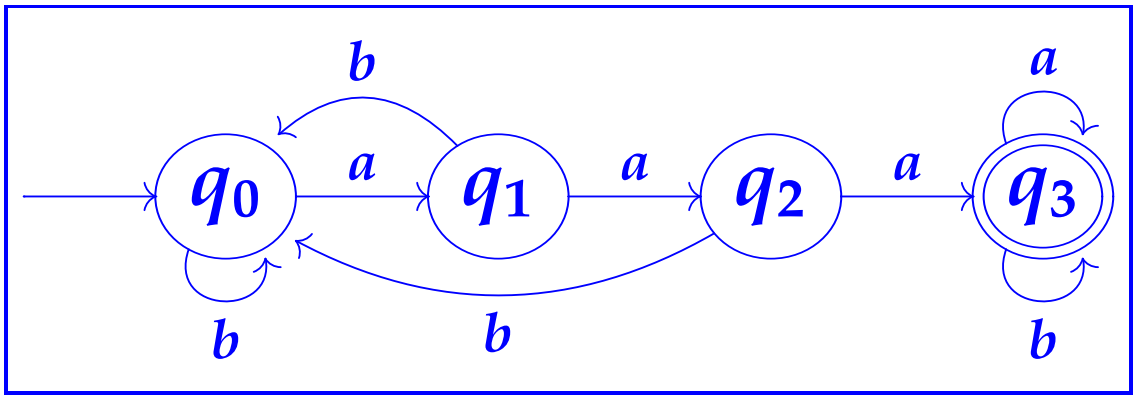
$\Delta : Q \times \Sigma \to \mathcal{P}Q$ is a functional

$\delta : Q \times \Sigma \to Q$

Example of a DFA

with input alphabet $\{a, b\}$

$M \triangleq$

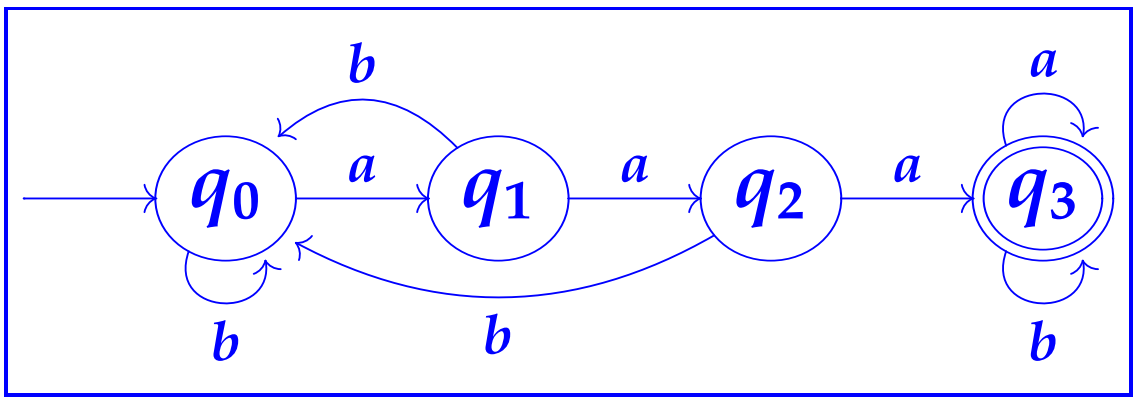


next-state function:

δ	a	b
q_0	q_1	q_0
q_1	q_2	q_0
q_2	q_3	q_0
q_3	q_3	q_3

Example of an NFA

with input alphabet $\{a, b, c\}$

$M \triangleq$ 

M is non-deterministic, because for example $\{q \mid q_0 \xrightarrow{c} q\} = \emptyset$.

ε-Transitions

When constructing machines for matching strings with regular expressions (as we will do later), it is useful to consider finite state machines exhibiting an 'internal' form of non-determinism in which the machine is allowed to change state without consuming any input symbol. One calls such transitions ε-**transitions** and writes them as $q \xrightarrow{\varepsilon} q'$. This leads to the definition on Slide 55.

When using an NFA$^\varepsilon$ M to accept a string $u \in \Sigma^*$ of input symbols, we are interested in sequences of transitions in which the symbols in u occur in the correct order, but with zero or more ε-transitions before or after each one. We write $q \overset{u}{\Rightarrow} q'$ to indicate that such a sequence exists from state q to state q' in the NFA$^\varepsilon$. Equivalently, $\{(q,u,q') \mid q \overset{u}{\Rightarrow} q'\}$ is the subset of $Q \times \Sigma^* \times Q$ inductively defined by

axioms: $\dfrac{}{(q,\varepsilon,q)}$ and rules: $\dfrac{(q,u,q')}{(q,u,q'')}$ if $q' \xrightarrow{\varepsilon} q''$, $\dfrac{(q,u,q')}{(q,ua,q'')}$ if $q' \xrightarrow{a} q''$ (see Exercise 7)

Slide 56 uses the relation $q \overset{u}{\Rightarrow} q'$ to define the language accepted by an NFA$^\varepsilon$. For example, for the NFA$^\varepsilon$ on Slide 55 it is not too hard to see that the language accepted consists of all strings which either contain two consecutive a's or contain two consecutive b's, i.e. the language determined by the regular expression $(a|b)^*(aa|bb)(a|b)^*$.

An **NFA with ε-transitions** (NFA$^\varepsilon$)

$$M = (Q, \Sigma, \Delta, s, F, T)$$

is an NFA $(Q, \Sigma, \Delta, s, F)$ together with a subset $T \subseteq Q \times Q$, called the **ε-transition relation**.

Example:

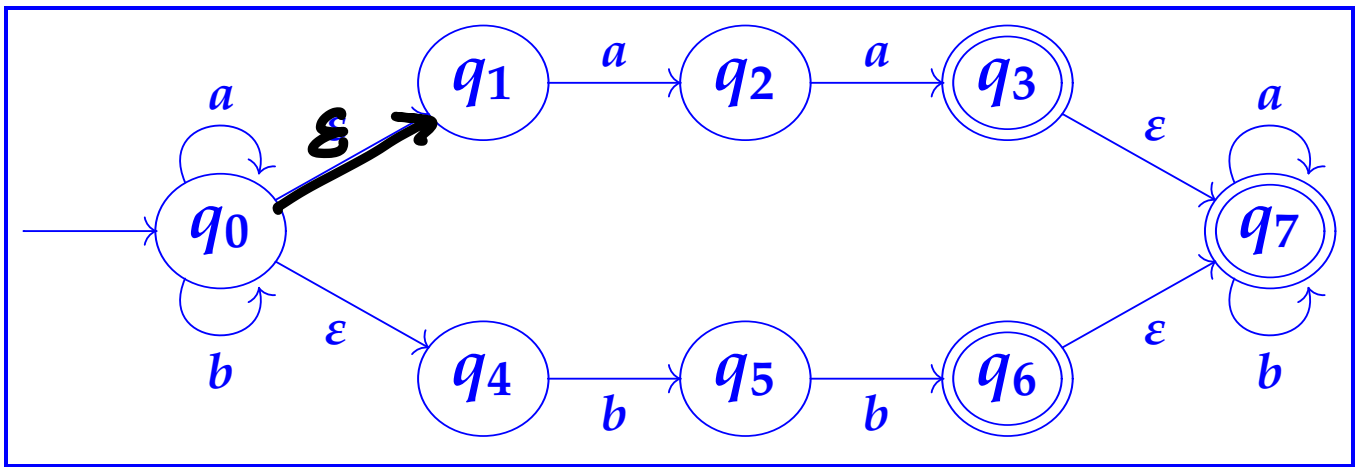


abb
$q_0 \Rightarrow q_6$

Notation: write "$q \xrightarrow{\varepsilon} q'$ in M" to mean $(q, q') \in T$.

(**N.B.** for NFA$^\varepsilon$s, we always assume $\varepsilon \notin \Sigma$.)

$T : Q \leftrightarrow Q$ $\quad q \; T \; q' \iff$ 

Language accepted by an NFA$^\varepsilon$

$M = (Q, \Sigma, \Delta, s, F, T)$

- Look at paths in the transition graph (including ε-transitions) from start state to *some* accepting state.
- Each such path gives a string in Σ^*, namely the string of non-ε labels that occur along the path.
- The set of all such strings is by definition **the language accepted by** M, written $L(M)$.

Notation: write $q \stackrel{u}{\Rightarrow} q'$ to mean that there is a path in M from state q to state q' whose non-ε labels form the string $u \in \Sigma^*$.

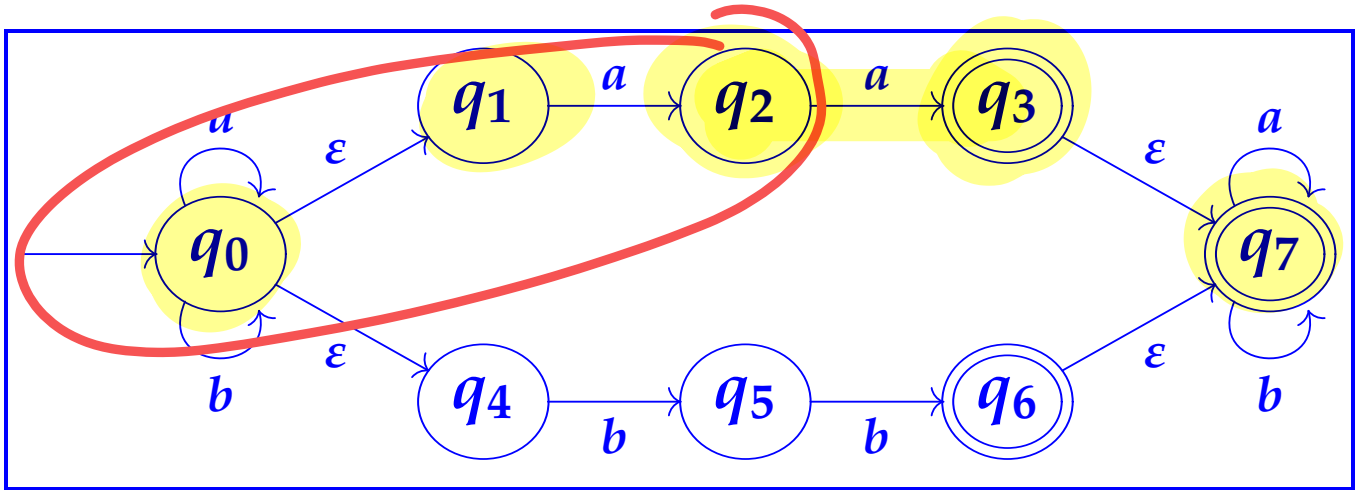
NFA: $q \stackrel{u}{\Rightarrow}^* q'$ NFA$^\varepsilon$: $q \stackrel{u}{\Rightarrow} q'$

An **NFA with ε-transitions** (NFA$^\varepsilon$)

$M = (Q, \Sigma, \Delta, s, F, T)$

is an NFA $(Q, \Sigma, \Delta, s, F)$ together with a subset $T \subseteq Q \times Q$, called the **ε-transition relation**.

Example:



For this NFA$^\varepsilon$ we have, e.g.: $q_0 \stackrel{aa}{\Rightarrow} q_2$, $q_0 \stackrel{aa}{\Rightarrow} q_3$ and $q_0 \stackrel{aa}{\Rightarrow} q_7$.

In fact the language of accepted strings is equal to the set of strings matching the regular expression $(a|b)^*(aa|bb)(a|b)^*$.

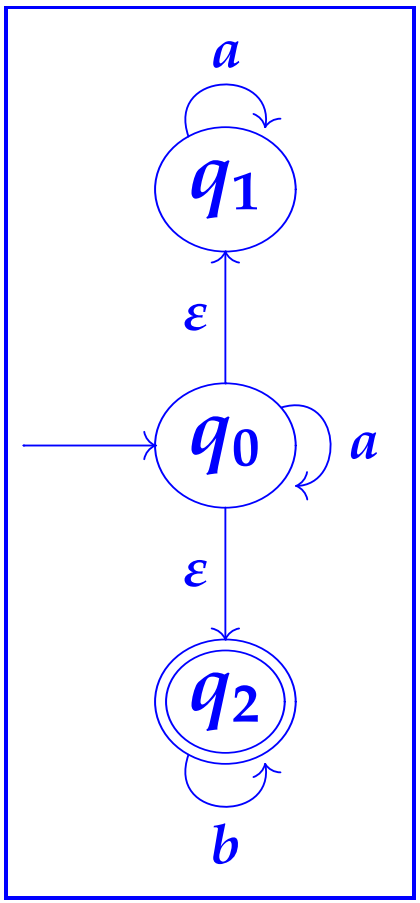
The subset construction for NFA$^\varepsilon$s

Note that every DFA is an NFA (whose transition relation is deterministic) and that every NFA is an NFA$^\varepsilon$ (whose ε-transition relation is empty). It might seem that non-determinism and ε-transitions allow a greater range of languages to be characterised as sets of strings accepted by a finite automaton, but this is not so. We can use a construction, called the **subset construction**, to convert an NFA$^\varepsilon$ M into a DFA PM accepting the same language (at the expense of increasing the number of states, possibly exponentially). Slide 59 gives an example of this construction.

The name 'subset construction' refers to the fact that there is one state of PM for each subset of the set of states of M. Given two such subsets, S and S' say, there is a transition $S \xrightarrow{a} S'$ in PM just in case S' consists of all the M-states q' reachable from states q in S via the $\cdot \stackrel{a}{\Rightarrow} \cdot$ relation defined on Slide 56, i.e. such that we can get from q to q' in M via finitely many ε-transitions followed by an a-transition followed by finitely many ε-transitions.

Example of the subset construction

M



next-state function for PM

	a	b
$\emptyset$	$\emptyset$	$\emptyset$
$\{q_0\}$	$\{q_0, q_1, q_2\}$	$\{q_2\}$
$\{q_1\}$	$\{q_1\}$	$\emptyset$
$\{q_2\}$	$\emptyset$	$\{q_2\}$
$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$	$\{q_2\}$
$\{q_0, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_2\}$
$\{q_1, q_2\}$	$\{q_1\}$	$\{q_2\}$
$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_2\}$

Theorem. For each NFA$^\varepsilon$ $M = (Q, \Sigma, \Delta, s, F, T)$ there is a DFA $PM = (\mathcal{P}(Q), \Sigma, \delta, s', F')$ accepting exactly the same strings as M, i.e. with $L(PM) = L(M)$.

Definition of PM:

- set of states is the powerset $\mathcal{P}(Q) = \{S \mid S \subseteq Q\}$ of the set Q of states of M
- same input alphabet Σ as for M
- next-state function maps each $(S, a) \in \mathcal{P}(Q) \times \Sigma$ to $\delta(S, a) \triangleq \{q' \in Q \mid \exists q \in S.\ q \stackrel{a}{\Rightarrow} q' \text{ in } M\}$
- start state is $s' \triangleq \{q' \in Q \mid s \stackrel{\varepsilon}{\Rightarrow} q'\}$
- subset of accepting states is $F' \triangleq \{S \in \mathcal{P}(Q) \mid S \cap F \neq \emptyset\}$

To prove the theorem we show that $L(M) \subseteq L(PM)$ and $L(PM) \subseteq L(M)$.

Proof that $L(M) \subseteq L(PM)$

Consider the case of ε first: if $\varepsilon \in L(M)$, then $s \stackrel{\varepsilon}{\Rightarrow} q$ for some $q \in F$, hence $s' \in F'$ and thus $\varepsilon \in L(PM)$.

Now given any non-null string $u = a_1 a_2 \ldots a_n$, if $u \in L(M)$, then there is a sequence of transitions in M of the form

$$s \stackrel{a_1}{\Rightarrow} q_1 \stackrel{a_2}{\Rightarrow} \cdots \stackrel{a_n}{\Rightarrow} q_n \in F \tag{1}$$

Since PM is deterministic, feeding $a_1 a_2 \ldots a_n$ to it results in the sequence of transitions

$$s' \stackrel{a_1}{\rightarrow} S_1 \stackrel{a_2}{\rightarrow} \cdots \stackrel{a_n}{\rightarrow} S_n \tag{2}$$

where $S_1 = \delta(s', a_1)$, $S_2 = \delta(S_1, a_2)$, etc. By definition of δ (Slide 60), from (1) we deduce

$q_1 \in \delta(s', a_1) = S_1$, hence $q_2 \in \delta(S_1, a_2) = S_2$, ..., hence $q_n \in \delta(S_{n-1}, a_n) = S_n$.

Therefore $S_n \in F'$ (because $q_n \in S_n \cap F$). So (2) shows that u is accepted by PM. □

Proof that $L(PM) \subseteq L(M)$

Consider the case of ε first: if $\varepsilon \in L(PM)$, then $s' \in F'$ and so there is some $q \in s'$ with $q \in F$, i.e. $s \stackrel{\varepsilon}{\Rightarrow} q \in F$ and thus $\varepsilon \in L(M)$.

Now given any non-null string $u = a_1 a_2 \ldots a_n$, if $u \in L(PM)$, then there is a sequence of transitions in PM of the form (2) with $S_n \in F'$, i.e. with S_n containing some $q_n \in F$. Now since $q_n \in S_n = \delta(S_{n-1}, a_n)$, by definition of δ there is some $q_{n-1} \in S_{n-1}$ with $q_{n-1} \stackrel{a_n}{\Rightarrow} q_n$ in M. Then since $q_{n-1} \in S_{n-1} = \delta(S_{n-2}, a_{n-1})$, there is some $q_{n-2} \in S_{n-2}$ with $q_{n-2} \stackrel{a_{n-1}}{\Rightarrow} q_{n-1}$. Working backwards in this way we can build up a sequence of transitions like (1) until, at the last step, from the fact that $q_1 \in S_1 = \delta(s', a_1)$ we deduce that $s \stackrel{a_1}{\Rightarrow} q_1$. So we get a sequence of transitions (1) with $q_n \in F$, and hence u is accepted by M. □