Nondeterministic Finite Automata

Last Updated: October 7th, 2025

1 Nondeterministic Finite Automata

To prove the closure of regular languages under union, concatenation, and star, it will be of great help to use the concept of a nondeterministic finite automata (NFA). An NFA has almost the same definition as a DFA, with one exception: upon reading an input symbol, a state transition allows for more than one possible next state. More formally, the codomain of function δ is no longer Q, but rather $\mathcal{P}(Q)$, the set of all subsets of Q.

A **nondeterministic finite automaton (NFA)** consists of a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

Q is a finite set with each member of Q called a **state**

Σ a finite alphabet that is used to represent an input word

δ a transition function that determines the set of possible next states of the automaton given the current state and the current input symbol being read. In other words,

$$\delta : Q \times \Sigma \to \mathcal{P}(Q),$$

$\mathcal{P}(Q)$ = power set of Q i.e. set of all subsets of Q

is a mapping from (current) state-input pairs to a *subset* of possible next states.

$q_0 \in Q$ the **initial state**.

$F \subseteq Q$ a member of F is called an **accepting state**.

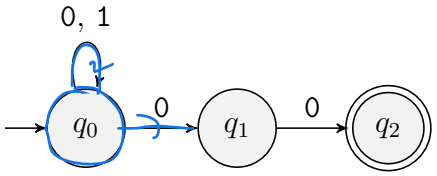
Note that every DFA is technically an NFA for which the transition function maps each state-input pair to a subset of size 1. Indeed, DFA transition function statement $\delta(q, s) = r$ can be expressed like an NFA statement by writing

$$\delta(q, s) = \{r\} \in \mathcal{P}(Q).$$

Example 1.1. The following state diagram provides an example of an NFA N that is not a DFA, since

1. state q_0 has two different outgoing edges that are labeled with 0
2. states q_1 and q_2 are both missing outgoing edges. This is done to simplify the state diagram.

The convention is, if N is in state q, reading input symbol s, and there is no outgoing edge from state q that is labeled with s, then the default is $\delta(q, s) = \emptyset$.



The following table represents transition function δ.

$q_i \backslash s$	0	1
q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\emptyset$
q_2	$\emptyset$	$\emptyset$

1.1 NFA computation

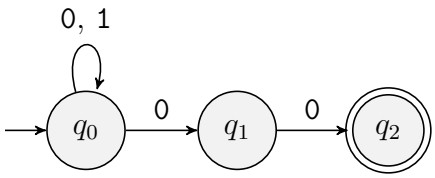
Let $N = (Q, \Sigma, \delta, q_0, F)$ be an NFA, and $w = w_1 w_2 \cdots w_n$ be a word in Σ^*. Then the **computation of N on input** w is a sequence $S_0, S_1, S_2, \ldots, S_n$ of subsets of states that are recursively defined as follows.

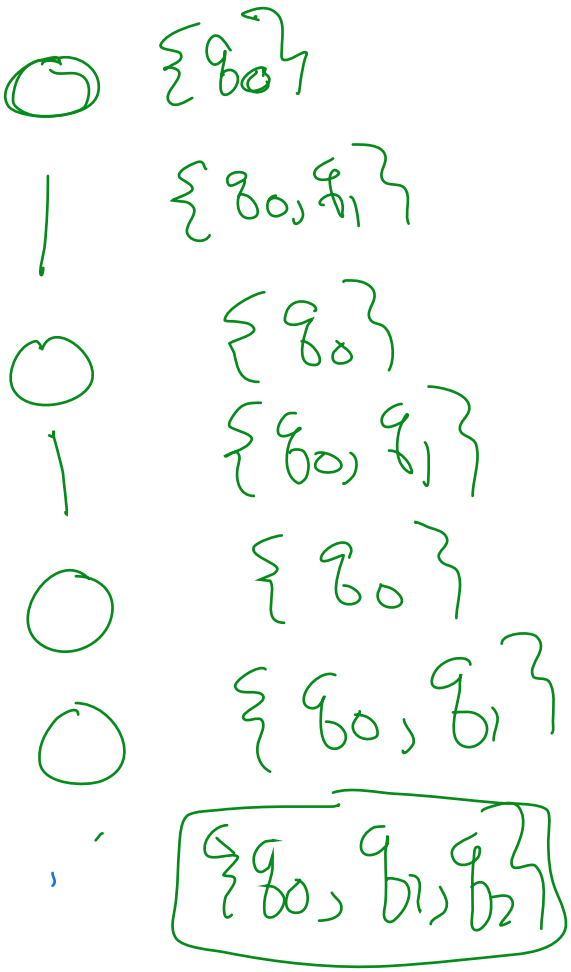
1. $S_0 = \{q_0\}$ is the singleton set consisting of the initial state q_0 of M
2. $S_i = \bigcup_{q \in S_{i-1}} \delta(q, w_i)$, for every $1 \leq i \leq n$.

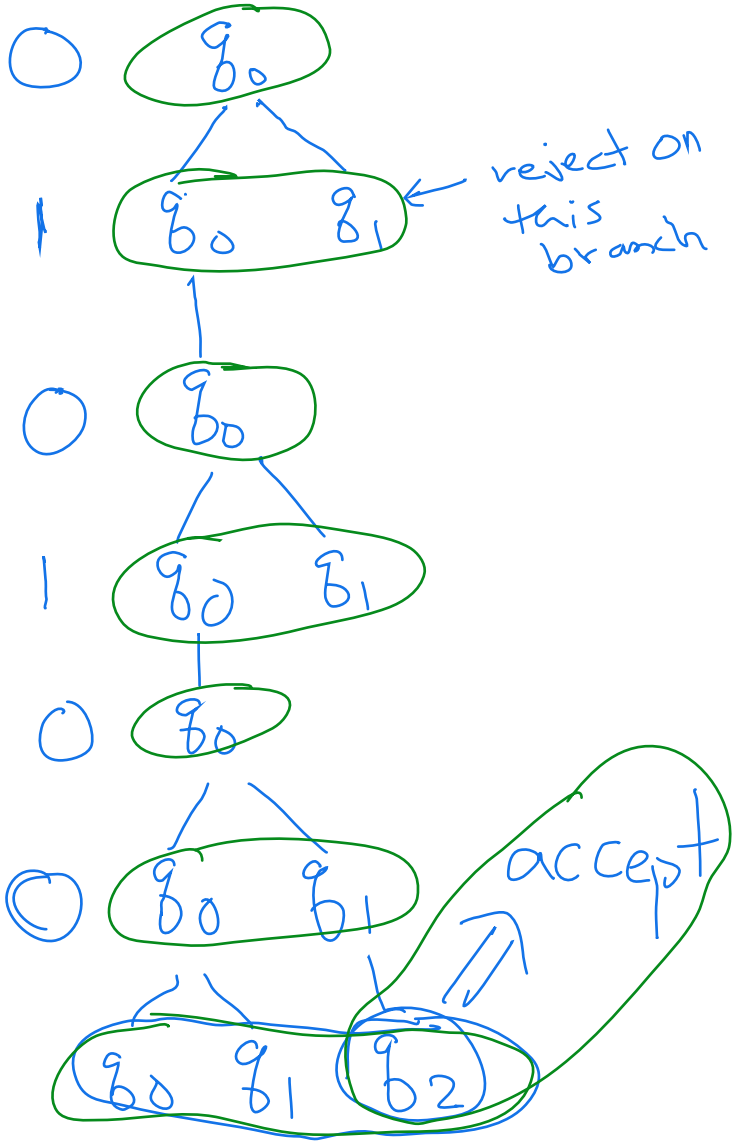
In words, the computation of N on input w is the sequence of state subsets $S_0, S_1, S_2, \ldots, S_n$ that N moves through when successively reading input symbols $w_1, w_2, \ldots, w_n$. The computation of N on input w is said to be **accepting** iff $S_n \cap F \neq \emptyset$. In other words, at least one accepting state belongs in S_n. In this case we say N **accepts** w. Otherwise it is called a **rejecting** computation, and N **rejects** w.

Rather than say that an NFA is in a particular state, we instead say that it is in a particular subset S of states.

Example 1.2. For the NFA N shown below, show the computation of N on inputs 010100 and 11010.







Accept since $q_2 \in \{q_0, q_1, q_2\}$ is accepting

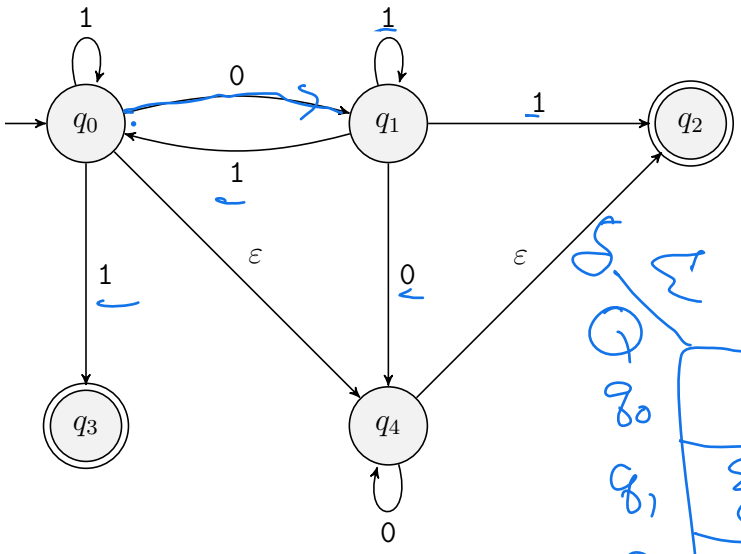
1.2 ε transitions

Another difference between an NFA state diagram and a DFA diagram is that an NFA node may have an edge leaving it that is labeled with ε. This means the following statements are true.

1. If there is an ε-labeled edge from q_1 and q_2, then q_2 is in a state subset S of the transition table whenever $q_1 \in S$.
2. As a corollary to the above, if q_0 is the designated initial state for N, the the initial subset state S_0 consists of

$$\{q_0\} \cup \{q \mid q \text{ is reachable from } q_0 \text{ via an } \varepsilon \text{ path.}\}.$$

Example 1.3. Provide the table of the δ-transition function for the following NFA, and show the computation on i) $w = 010$ and ii) $w = 101$.



initial state for a computation is $\{q_0, q_4, q_2\}$

Q	0	1
q_0	$\{q_1\}$	$\{q_0, q_2, q_3, q_4\}$
q_1	$\{q_2, q_4\}$	$\{q_1, q_2, q_0, q_4\}$
q_2	$\emptyset$	$\emptyset$
q_3	$\emptyset$	$\emptyset$
q_4	$\{q_2, q_4\}$	$\emptyset$

1.3 Equivalence between DFA's and NFA's

Since every DFA is also an NFA, it follows that every regular language L is accepted by some NFA. The converse is also true! Every language L that is accepted by an NFA is regular.

This is because an NFA is actually a DFA in disguise. The reason why an NFA is not a DFA is that it does not meet the transition-function definition for DFA's. That definition states

that

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

must map a state-input pair to a state. But an NFA N maps a state-input pair to a subset of states. Recalling that N's formal definition is

$$N = (Q, \Sigma, \delta : Q \times \Sigma \to \mathcal{P}(Q), q_0, F),$$

we can make the following changes to this definition to create a DFA N'.

Changing Q to $\mathcal{P}(Q)$: every state is now a subset of the original set of states

Changing the domain of δ: instead of $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ we now have

$$\delta' : \mathcal{P}(Q) \times \Sigma \to \mathcal{P}(Q),$$

which is now a DFA transition function because it maps a subset state-input pair (S, a) to a subset state T.

Changing the initial state: instead of q_0, the initial state is now the singleton set $\{q_0\}$.

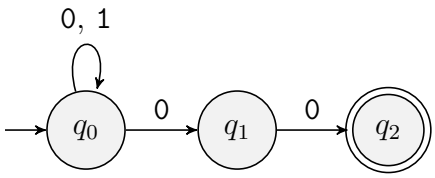
Changing the subset of accepting states: instead of F, we have F' where subset $S \in F'$ iff $S \cap F \neq \emptyset$. In other words, a subset is an accepting state iff it contains some accepting state of N.

The new DFA N' may now be formally expressed as

$$N' = (\mathcal{P}(Q), \Sigma, \delta', \{q_0\}, F').$$

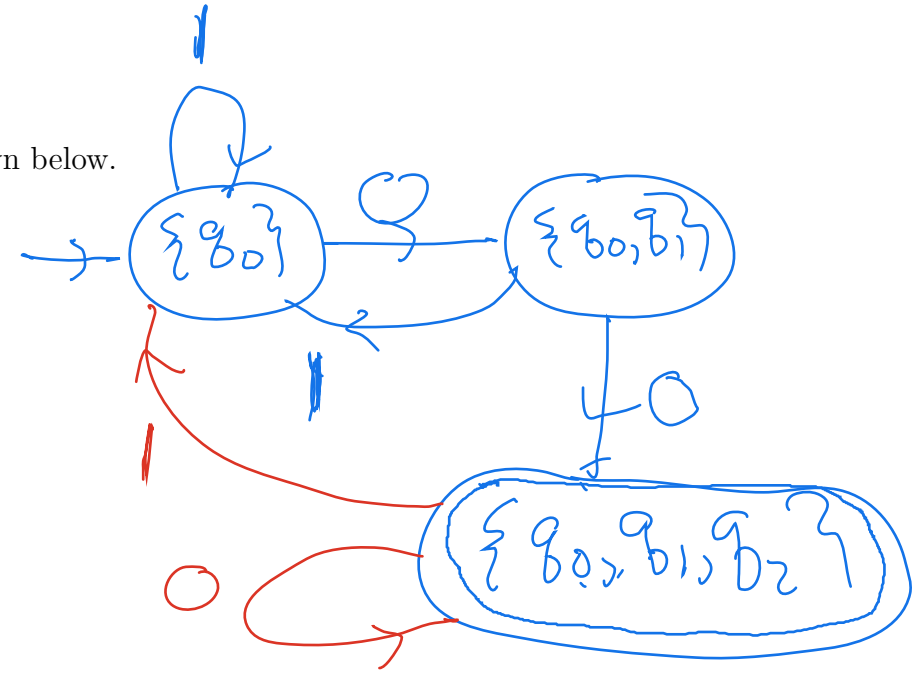
Moreover, the computation of N' on some input word w is defined *exactly* in the same way as the computation of N on w. In other words, N accepts w iff N' accepts w. To see this, as an exercise, write out the definition of what it means to be an accepting computation for a DFA, but within the context of machine N'. Then show that it is identical to the definition of what it means to be an accepting computation for NFA N. Both definitions should appear identical.

Example 1.4. Convert the following NFA to a DFA

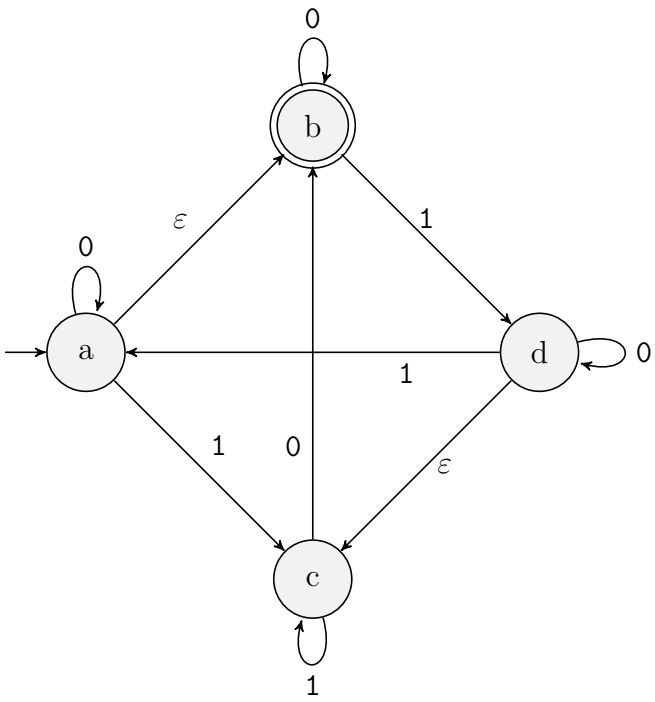


whose transition function δ is shown below.

$q_i \backslash s$	0	1
q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\emptyset$
q_2	$\emptyset$	$\emptyset$

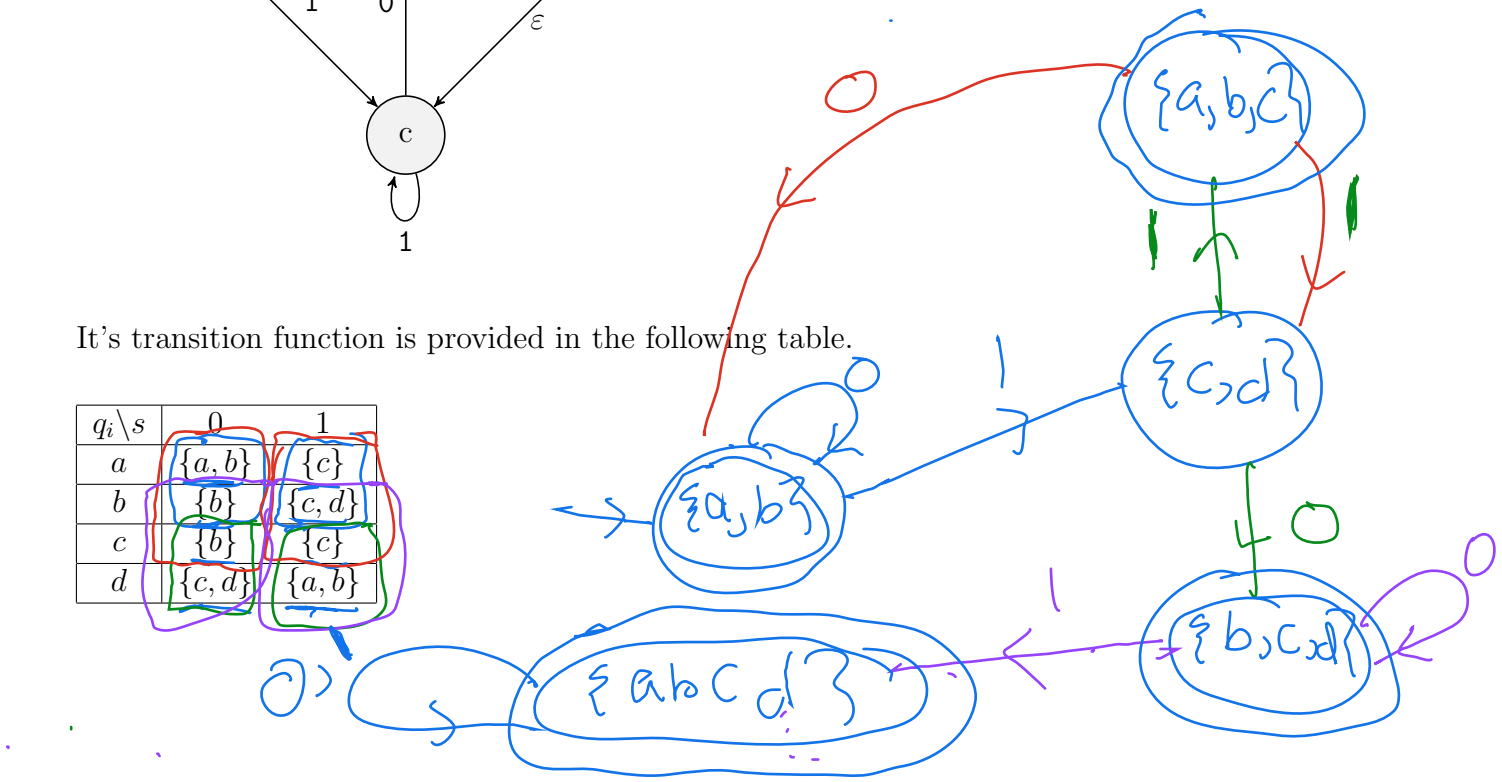


Example 1.5. Convert the following NFA to a DFA by using the method of subset states.



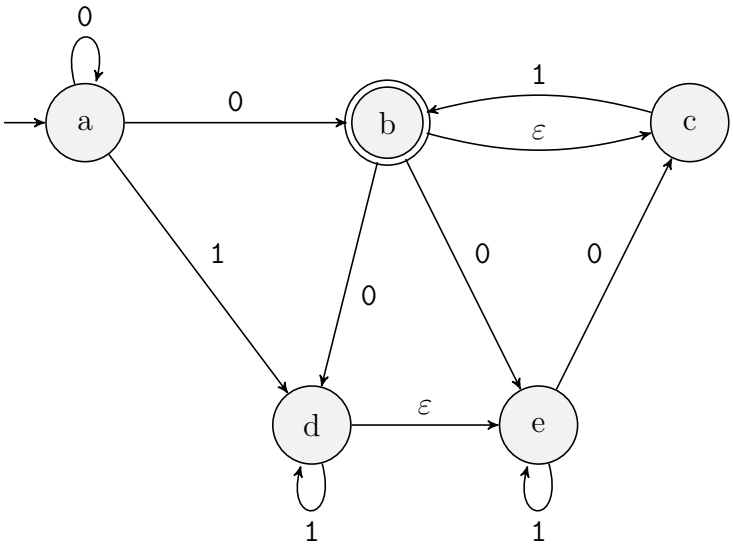
It's transition function is provided in the following table.

$q_i \backslash s$	0	1
a	$\{a, b\}$	$\{c\}$
b	$\{b\}$	$\{c, d\}$
c	$\{b\}$	$\{c\}$
d	$\{c, d\}$	$\{a, b\}$



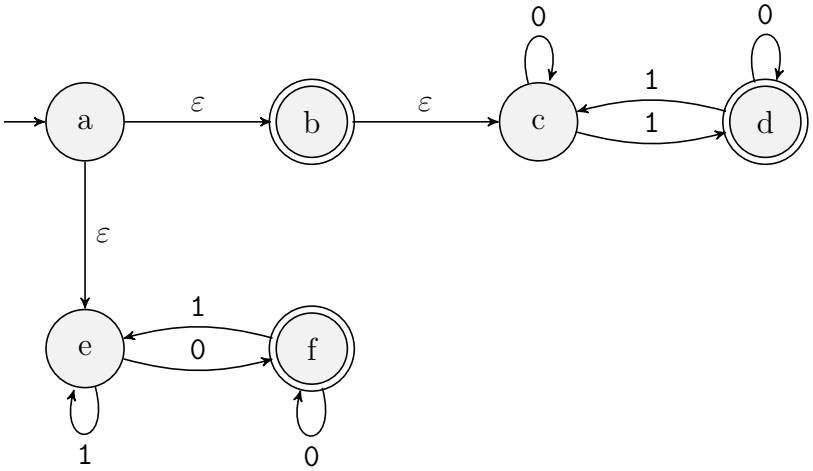
NFA Core Exercises

1. Provide the δ-transition function for the NFA whose state diagram is shown below.



2. For the NFA from the previous exercise, provide the computation on inputs 0010 and 10111. Which of the two are accepted?

3. Provide the δ-transition function for the NFA N whose state diagram is shown below.



4. For the NFA from the previous exercise, provide the computation on inputs ε, 0101 and 000. Verify that all three are accepted by N. Can you find an input that does not get accepted?

5. Provide the state diagram of an NFA that accepts the language

   (a) $\{w|w$ ends with $00\}$ and uses only three states.

   (b) $\{w|w$ contains the substring $0101\}$ and uses only five states.

   (c) $\{w|w$ contains an even number of zeros and exactly two ones $\}$ and uses only six states.

   (d) $\{0\}$ and uses only two states.

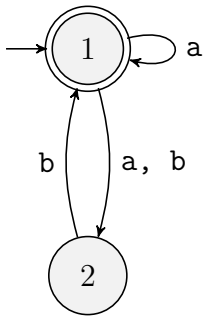
   (e) $0^*1^*0^+$ and uses only three states.

(f) $1^*(001^+)^*$ and uses only three states.

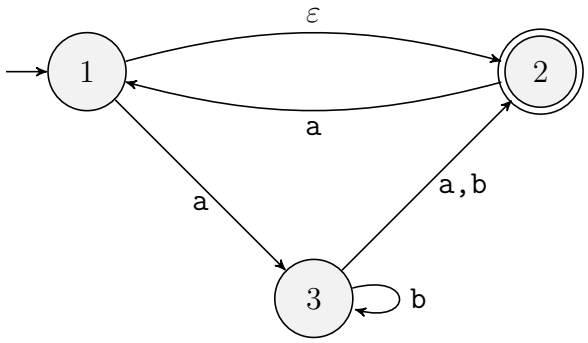
(g) ε and uses only one state.

(h) 0^* and uses only one state.

6. Provide a DFA M that accepts the same language as that accepted by the NFA N that is defined in the following state diagram. Do so by defining the states of M to be subsets of the states of N.



7. Provide a DFA M that accepts the same language as that accepted by the NFA N that is defined in the following state diagram. Do so by defining the states of M to be subsets of the states of N.



Solutions to NFA Core Exercises

1. The following table provides the $\delta(Q, \Sigma)$ transition function.

$Q \backslash \Sigma$	0	1
a	{a,b,c}	{d,e}
b	{d,e}	$\emptyset$
c	$\emptyset$	{b,c}
d	$\emptyset$	{d,e}
e	{c}	{e}

2. We have the following computations.

Input Symbol Read	**Current Subset State**
0	{a}
0	{a,b,c}
1	{a,b,c,d,e}
0	{b,c,d,e}
Rejecting State:	{c,d,e}

Input Symbol Read	**Current Subset State**
1	{a}
0	{d,e}
1	{c}
1	{b,c}
1	{b,c}
Accepting State:	{b,c}

3. The following table provides the $\delta(Q, \Sigma)$ transition function.

$Q \backslash \Sigma$	0	1
a	$\emptyset$	$\emptyset$
b	$\emptyset$	$\emptyset$
c	{c}	{d}
d	{d}	{c}
e	{f}	{e}
f	{f}	{e}

4. We have the following computations. For ε, the final state equals {a,b,c,e} which is accepting since b is an accepting state.

Input Symbol Read	**Current Subset State**
0	{a, b, c, e}
1	{c,f}
0	{d,e}
1	{d,f}
Rejecting State:	{c,e}

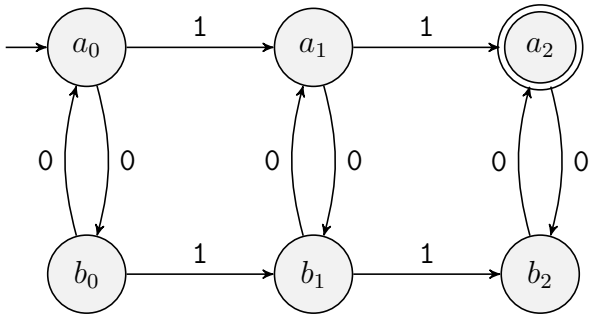
Input Symbol Read	Current Subset State
0	$\{a, b, c, e\}$
0	{c,f}
0	{c,f}
Accepting State:	{c,f}

5. We have the following.

   (a)

   (b)

   (c) We have the following NFA.



   (d)

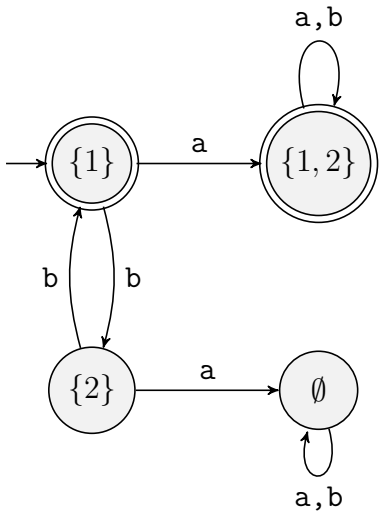
   (e)

   (f)

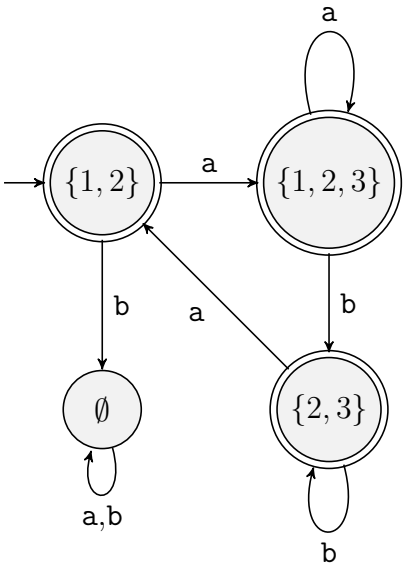
   (g)

   (h)

6. We have the following DFA.



7. We have the following DFA. Notice that the initial state is $\{1, 2\}$, since, becuase of the ε edge, (initially) entering state 1 triggers the entering of state 2.


a
{1, 2}
a
{1, 2, 3}
b
a
b
∅
{2, 3}
a,b
b