Formal Languages, Automata and Codes

Oleg Gutik

Lecture 7

2.4 Reduction of the Number of States in Finite Automata

Any DFA defines a unique language, but the converse is not true. For a given language, there are many DFA's that accept it. There may be a considerable difference in the number of states of such equivalent automata. In terms of the questions we have considered so far, all solutions are equally satisfactory, but if the results are to be applied in a practical setting, there may be reasons for preferring one over another.

Example 2.14

The two DFA's depicted in (a) and (b) are equivalent, as a few test strings will quickly reveal.

(a) (b)

Any DFA defines a unique language, but the converse is not true. For a given language, there are many DFA's that accept it. There may be a considerable difference in the number of states of such equivalent automata. In terms of the questions we have considered so far, all solutions are equally satisfactory, but if the results are to be applied in a practical setting, there may be reasons for preferring one over another.

Example 2.14

The two DFA's depicted in (a) and (b) are equivalent, as a few test strings will quickly reveal.

2.4 Reduction of the Number of States in Finite Automata

Any DFA defines a unique language, but the converse is not true. For a given language, there are many DFA's that accept it. There may be a considerable difference in the number of states of such equivalent automata. In terms of the questions we have considered so far, all solutions are equally satisfactory, but if the results are to be applied in a practical setting, there may be reasons for preferring one over another.

Example 2.14

The two DFA's depicted in (a) and (b) are equivalent, as a few test strings will quickly reveal.

(a) (b)

2.4 Reduction of the Number of States in Finite Automata

Any DFA defines a unique language, but the converse is not true. For a given language, there are many DFA's that accept it. There may be a considerable difference in the number of states of such equivalent automata. In terms of the questions we have considered so far, all solutions are equally satisfactory, but if the results are to be applied in a practical setting, there may be reasons for preferring one over another.

Example 2.14

The two DFA's depicted in (a) and (b) are equivalent, as a few test strings will quickly reveal.

(a) (b)

2.4 Reduction of the Number of States in Finite Automata

Any DFA defines a unique language, but the converse is not true. For a given language, there are many DFA's that accept it. There may be a considerable difference in the number of states of such equivalent automata. In terms of the questions we have considered so far, all solutions are equally satisfactory, but if the results are to be applied in a practical setting, there may be reasons for preferring one over another.

Example 2.14

The two DFA's depicted in (a) and (b) are equivalent, as a few test strings will quickly reveal.

(a) (b)

2.4 Reduction of the Number of States in Finite Automata

Any DFA defines a unique language, but the converse is not true. For a given language, there are many DFA's that accept it. There may be a considerable difference in the number of states of such equivalent automata. In terms of the questions we have considered so far, all solutions are equally satisfactory, but if the results are to be applied in a practical setting, there may be reasons for preferring one over another.

Example 2.14

The two DFA's depicted in (a) and (b) are equivalent, as a few test strings will quickly reveal.

(a) (b)

Any DFA defines a unique language, but the converse is not true. For a given language, there are many DFA's that accept it. There may be a considerable difference in the number of states of such equivalent automata. In terms of the questions we have considered so far, all solutions are equally satisfactory, but if the results are to be applied in a practical setting, there may be reasons for preferring one over another.

Example 2.14

The two DFA's depicted in (a) and (b) are equivalent, as a few test strings will quickly reveal.

(a) (b)

Any DFA defines a unique language, but the converse is not true. For a given language, there are many DFA's that accept it. There may be a considerable difference in the number of states of such equivalent automata. In terms of the questions we have considered so far, all solutions are equally satisfactory, but if the results are to be applied in a practical setting, there may be reasons for preferring one over another.

Example 2.14

The two DFA's depicted in (a) and (b) are equivalent, as a few test strings will quickly reveal.

(a)

(b)

Any DFA defines a unique language, but the converse is not true. For a given language, there are many DFA's that accept it. There may be a considerable difference in the number of states of such equivalent automata. In terms of the questions we have considered so far, all solutions are equally satisfactory, but if the results are to be applied in a practical setting, there may be reasons for preferring one over another.

Example 2.14

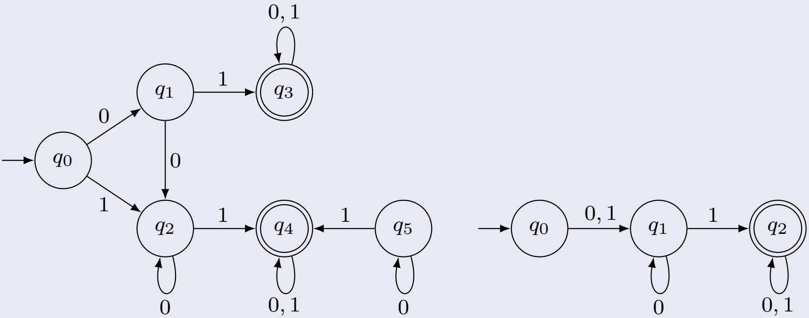
The two DFA's depicted in (a) and (b) are equivalent, as a few test strings will quickly reveal.

(a) (b)

Any DFA defines a unique language, but the converse is not true. For a given language, there are many DFA's that accept it. There may be a considerable difference in the number of states of such equivalent automata. In terms of the questions we have considered so far, all solutions are equally satisfactory, but if the results are to be applied in a practical setting, there may be reasons for preferring one over another.

Example 2.14

The two DFA's depicted in (a) and (b) are equivalent, as a few test strings will quickly reveal.



(a)

(b)

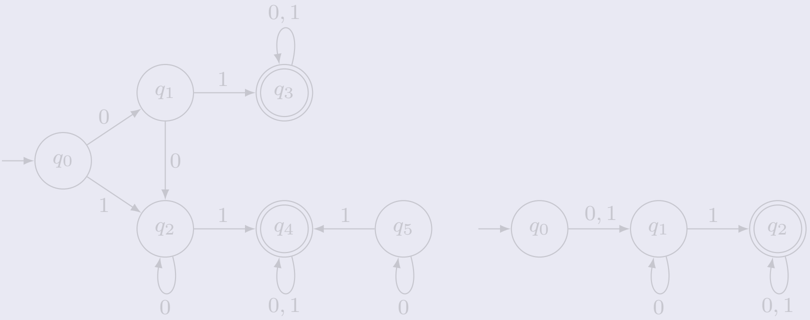
Example 2.14 (continuation)

We notice some obviously unnecessary features of Figure (a). The state q_5 plays absolutely no role in the automaton since it can never be reached from the initial state q_0. Such a state is inaccessible, and it can be removed (along with all transitions relating to it) without affecting the language accepted by the automaton. But even after the removal of q_5, the first automaton has some redundant parts. The states reachable subsequent to the first move $\delta(q_0, 0)$ mirror those reachable from a first move $\delta(q_0, 1)$. The second automaton combines these two options.

(a) (b)

Example 2.14 (continuation)

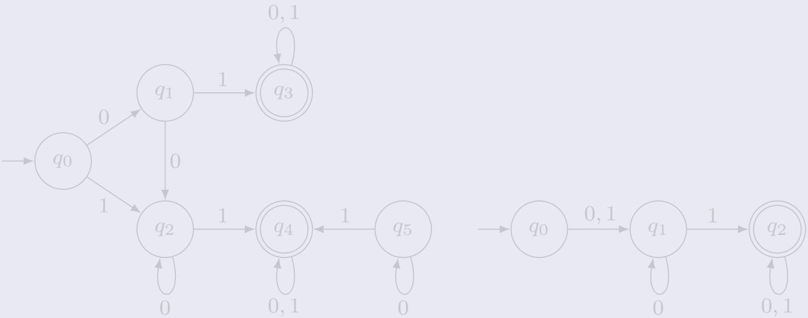
We notice some obviously unnecessary features of Figure (a). The state q_5 plays absolutely no role in the automaton since it can never be reached from the initial state q_0. Such a state is inaccessible, and it can be removed (along with all transitions relating to it) without affecting the language accepted by the automaton. But even after the removal of q_5, the first automaton has some redundant parts. The states reachable subsequent to the first move $\delta(q_0, 0)$ mirror those reachable from a first move $\delta(q_0, 1)$. The second automaton combines these two options.



(a) (b)

Example 2.14 (continuation)

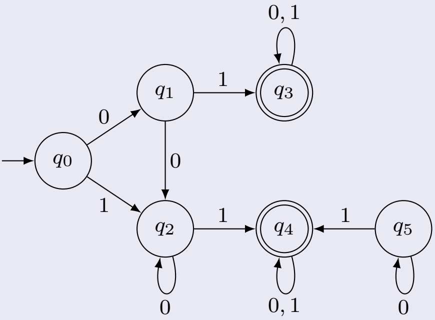
We notice some obviously unnecessary features of Figure (a). The state q_5 plays absolutely no role in the automaton since it can never be reached from the initial state q_0. Such a state is inaccessible, and it can be removed (along with all transitions relating to it) without affecting the language accepted by the automaton. But even after the removal of q_5, the first automaton has some redundant parts. The states reachable subsequent to the first move $\delta(q_0, 0)$ mirror those reachable from a first move $\delta(q_0, 1)$. The second automaton combines these two options.



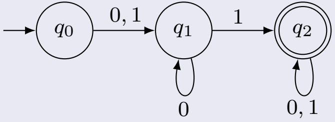
(a) (b)

Example 2.14 (continuation)

We notice some obviously unnecessary features of Figure (a). The state q_5 plays absolutely no role in the automaton since it can never be reached from the initial state q_0. Such a state is inaccessible, and it can be removed (along with all transitions relating to it) without affecting the language accepted by the automaton. But even after the removal of q_5, the first automaton has some redundant parts. The states reachable subsequent to the first move $\delta(q_0, 0)$ mirror those reachable from a first move $\delta(q_0, 1)$. The second automaton combines these two options.



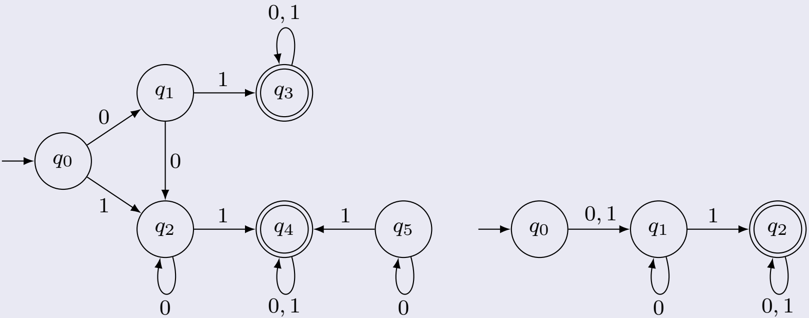
(a)



(b)

Example 2.14 (continuation)

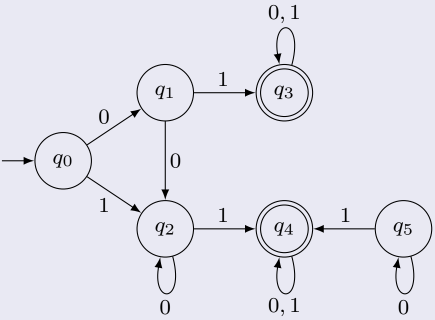
We notice some obviously unnecessary features of Figure (a). The state q_5 plays absolutely no role in the automaton since it can never be reached from the initial state q_0. Such a state is inaccessible, and it can be removed (along with all transitions relating to it) without affecting the language accepted by the automaton. But even after the removal of q_5, the first automaton has some redundant parts. The states reachable subsequent to the first move $\delta(q_0, 0)$ mirror those reachable from a first move $\delta(q_0, 1)$. The second automaton combines these two options.



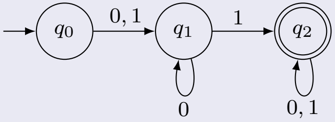
(a) (b)

Example 2.14 (continuation)

We notice some obviously unnecessary features of Figure (a). The state q_5 plays absolutely no role in the automaton since it can never be reached from the initial state q_0. Such a state is inaccessible, and it can be removed (along with all transitions relating to it) without affecting the language accepted by the automaton. But even after the removal of q_5, the first automaton has some redundant parts. The states reachable subsequent to the first move $\delta(q_0, 0)$ mirror those reachable from a first move $\delta(q_0, 1)$. The second automaton combines these two options.



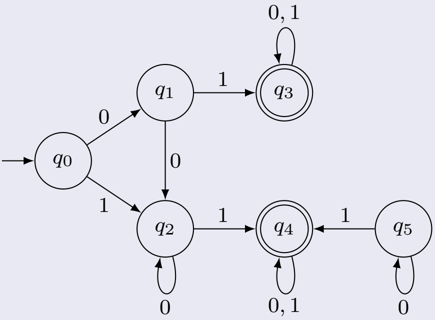
(a)



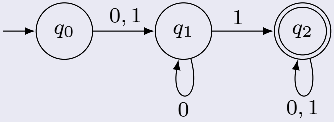
(b)

Example 2.14 (continuation)

We notice some obviously unnecessary features of Figure (a). The state q_5 plays absolutely no role in the automaton since it can never be reached from the initial state q_0. Such a state is inaccessible, and it can be removed (along with all transitions relating to it) without affecting the language accepted by the automaton. But even after the removal of q_5, the first automaton has some redundant parts. The states reachable subsequent to the first move $\delta(q_0, 0)$ mirror those reachable from a first move $\delta(q_0, 1)$. The second automaton combines these two options.



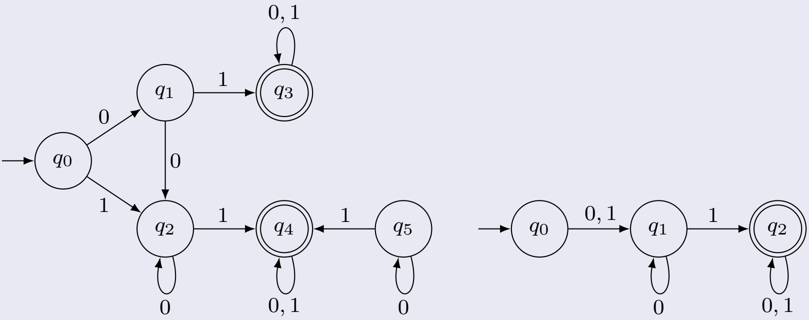
(a)



(b)

Example 2.14 (continuation)

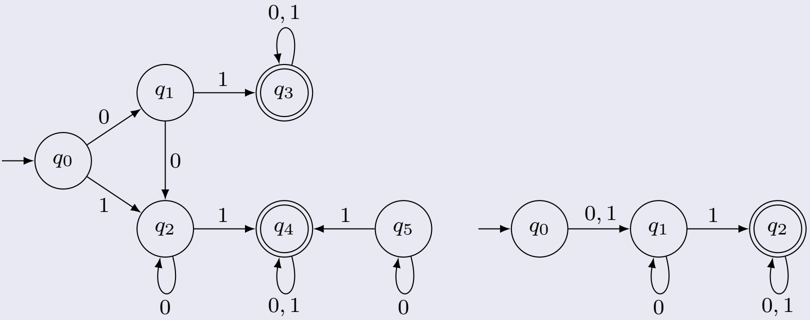
We notice some obviously unnecessary features of Figure (a). The state q_5 plays absolutely no role in the automaton since it can never be reached from the initial state q_0. Such a state is inaccessible, and it can be removed (along with all transitions relating to it) without affecting the language accepted by the automaton. But even after the removal of q_5, the first automaton has some redundant parts. The states reachable subsequent to the first move $\delta(q_0, 0)$ mirror those reachable from a first move $\delta(q_0, 1)$. The second automaton combines these two options.



(a) (b)

Example 2.14 (continuation)

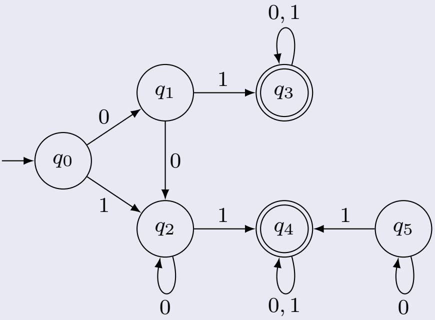
We notice some obviously unnecessary features of Figure (a). The state q_5 plays absolutely no role in the automaton since it can never be reached from the initial state q_0. Such a state is inaccessible, and it can be removed (along with all transitions relating to it) without affecting the language accepted by the automaton. But even after the removal of q_5, the first automaton has some redundant parts. The states reachable subsequent to the first move $\delta(q_0, 0)$ mirror those reachable from a first move $\delta(q_0, 1)$. The second automaton combines these two options.



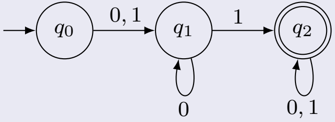
(a) (b)

Example 2.14 (continuation)

We notice some obviously unnecessary features of Figure (a). The state q_5 plays absolutely no role in the automaton since it can never be reached from the initial state q_0. Such a state is inaccessible, and it can be removed (along with all transitions relating to it) without affecting the language accepted by the automaton. But even after the removal of q_5, the first automaton has some redundant parts. The states reachable subsequent to the first move $\delta(q_0, 0)$ mirror those reachable from a first move $\delta(q_0, 1)$. The second automaton combines these two options.



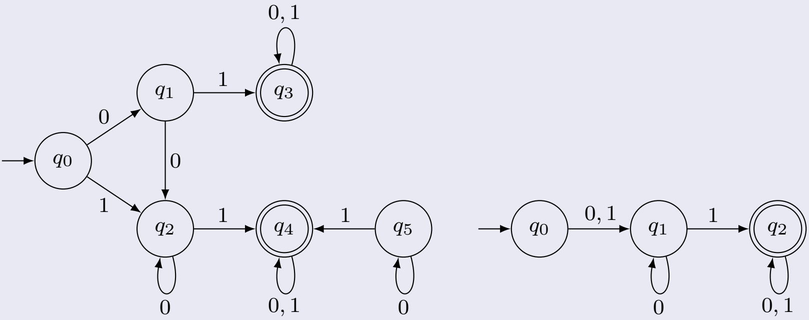
(a)



(b)

Example 2.14 (continuation)

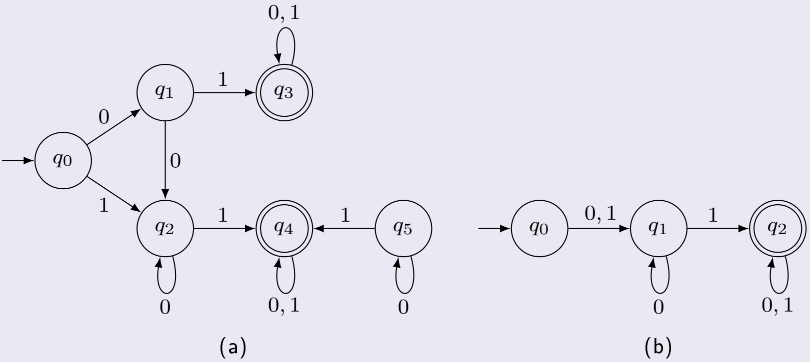
We notice some obviously unnecessary features of Figure (a). The state q_5 plays absolutely no role in the automaton since it can never be reached from the initial state q_0. Such a state is inaccessible, and it can be removed (along with all transitions relating to it) without affecting the language accepted by the automaton. But even after the removal of q_5, the first automaton has some redundant parts. The states reachable subsequent to the first move $\delta(q_0, 0)$ mirror those reachable from a first move $\delta(q_0, 1)$. The second automaton combines these two options.



(a) (b)

Example 2.14 (continuation)

We notice some obviously unnecessary features of Figure (a). The state q_5 plays absolutely no role in the automaton since it can never be reached from the initial state q_0. Such a state is inaccessible, and it can be removed (along with all transitions relating to it) without affecting the language accepted by the automaton. But even after the removal of q_5, the first automaton has some redundant parts. The states reachable subsequent to the first move $\delta(q_0, 0)$ mirror those reachable from a first move $\delta(q_0, 1)$. The second automaton combines these two options.



From a strictly theoretical point of view, there is little reason for preferring the automaton in Figure (b) over that in Figure (a).

(a)

(b)

However, in terms of simplicity, the second alternative is clearly preferable. Representation of an automaton for the purpose of computation requires space proportional to the number of states. For storage efficiency, it is desirable to reduce the number of states as far as possible. We now describe an algorithm that accomplishes this.

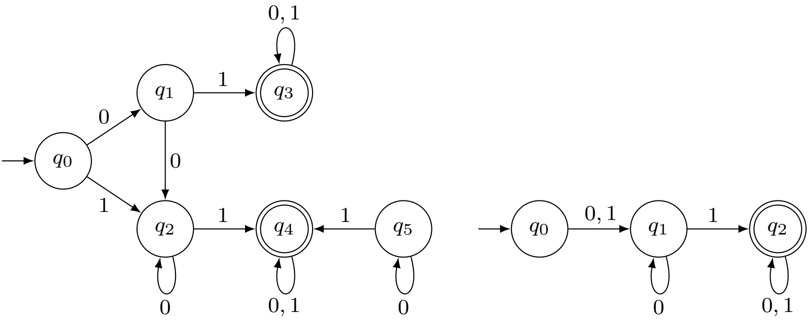
From a strictly theoretical point of view, there is little reason for preferring the automaton in Figure (b) over that in Figure (a).

(a)

(b)

However, in terms of simplicity, the second alternative is clearly preferable. Representation of an automaton for the purpose of computation requires space proportional to the number of states. For storage efficiency, it is desirable to reduce the number of states as far as possible. We now describe an algorithm that accomplishes this.

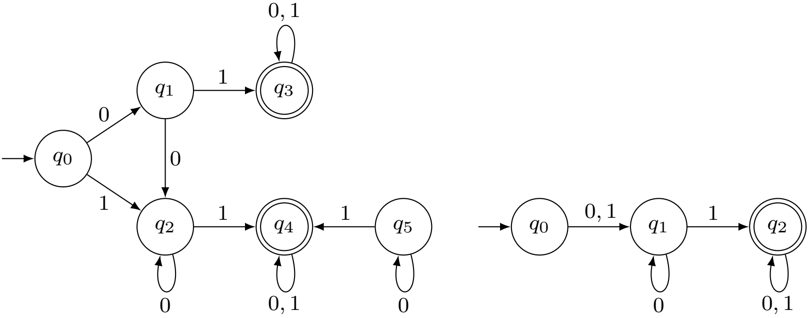
From a strictly theoretical point of view, there is little reason for preferring the automaton in Figure (b) over that in Figure (a).



(a) (b)

However, in terms of simplicity, the second alternative is clearly preferable. Representation of an automaton for the purpose of computation requires space proportional to the number of states. For storage efficiency, it is desirable to reduce the number of states as far as possible. We now describe an algorithm that accomplishes this.

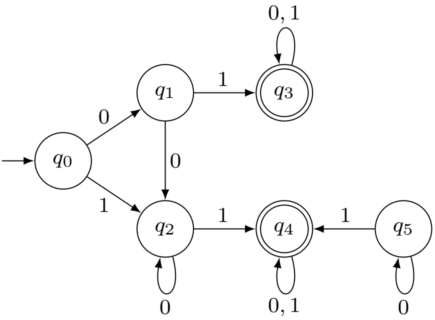
From a strictly theoretical point of view, there is little reason for preferring the automaton in Figure (b) over that in Figure (a).



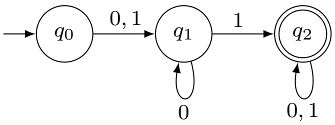
(a) (b)

However, in terms of simplicity, the second alternative is clearly preferable. Representation of an automaton for the purpose of computation requires space proportional to the number of states. For storage efficiency, it is desirable to reduce the number of states as far as possible. We now describe an algorithm that accomplishes this.

From a strictly theoretical point of view, there is little reason for preferring the automaton in Figure (b) over that in Figure (a).



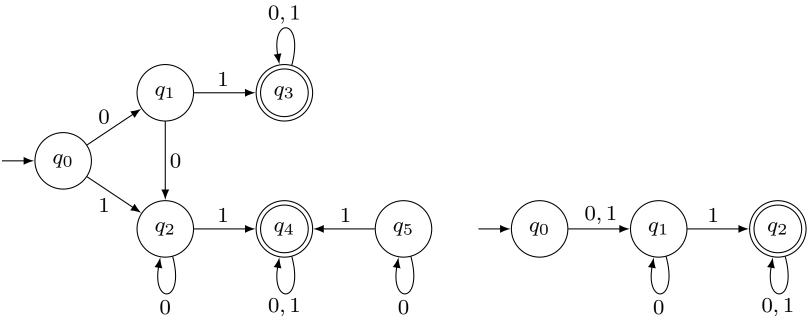
(a)



(b)

However, in terms of simplicity, the second alternative is clearly preferable.
Representation of an automaton for the purpose of computation requires space proportional to the number of states. For storage efficiency, it is desirable to reduce the number of states as far as possible. We now describe an algorithm that accomplishes this.

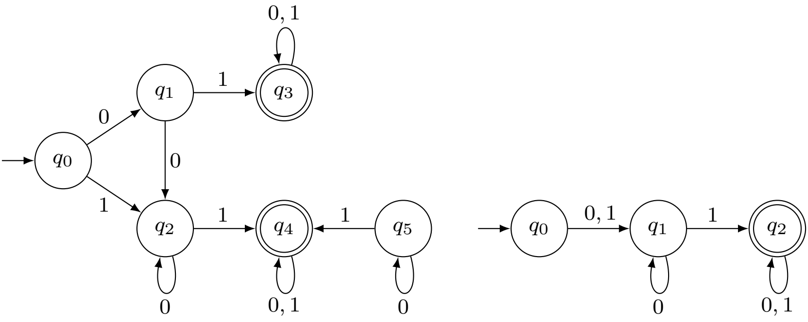
From a strictly theoretical point of view, there is little reason for preferring the automaton in Figure (b) over that in Figure (a).



(a) (b)

However, in terms of simplicity, the second alternative is clearly preferable. Representation of an automaton for the purpose of computation requires space proportional to the number of states. For storage efficiency, it is desirable to reduce the number of states as far as possible. We now describe an algorithm that accomplishes this.

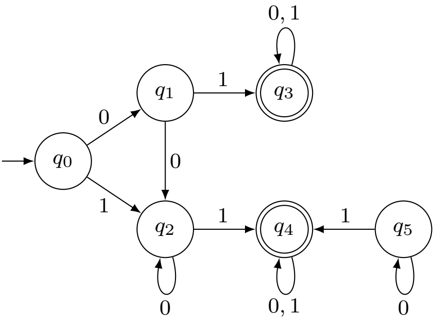
From a strictly theoretical point of view, there is little reason for preferring the automaton in Figure (b) over that in Figure (a).



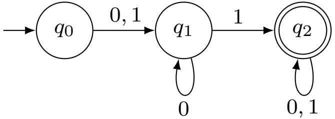
(a) (b)

However, in terms of simplicity, the second alternative is clearly preferable. Representation of an automaton for the purpose of computation requires space proportional to the number of states. For storage efficiency, it is desirable to reduce the number of states as far as possible. We now describe an algorithm that accomplishes this.

From a strictly theoretical point of view, there is little reason for preferring the automaton in Figure (b) over that in Figure (a).



(a)



(b)

However, in terms of simplicity, the second alternative is clearly preferable. Representation of an automaton for the purpose of computation requires space proportional to the number of states. For storage efficiency, it is desirable to reduce the number of states as far as possible. We now describe an algorithm that accomplishes this.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \in F,$$

and

$$\delta^*(p, w) \notin F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \in F,$$

and

$$\delta^*(p, w) \notin F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \in F,$$

and

$$\delta^*(p, w) \notin F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \in F,$$

and

$$\delta^*(p, w) \notin F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \in F,$$

and

$$\delta^*(p, w) \notin F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p,w) \in F \quad \text{implies} \quad \delta^*(q,w) \in F,$$

and

$$\delta^*(p,w) \notin F \quad \text{implies} \quad \delta^*(q,w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p,w) \in F \quad \text{implies} \quad \delta^*(q,w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p,w) \in F \quad \text{implies} \quad \delta^*(q,w) \in F,$$

and

$$\delta^*(p,w) \notin F \quad \text{implies} \quad \delta^*(q,w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p,w) \in F \quad \text{implies} \quad \delta^*(q,w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p,w) \in F \quad \text{implies} \quad \delta^*(q,w) \in F,$$

and

$$\delta^*(p,w) \notin F \quad \text{implies} \quad \delta^*(q,w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p,w) \in F \quad \text{implies} \quad \delta^*(q,w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p,w) \in F \quad \text{implies} \quad \delta^*(q,w) \in F,$$

and

$$\delta^*(p,w) \notin F \quad \text{implies} \quad \delta^*(q,w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p,w) \in F \quad \text{implies} \quad \delta^*(q,w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p,w) \in F \quad \text{implies} \quad \delta^*(q,w) \in F,$$

and

$$\delta^*(p,w) \notin F \quad \text{implies} \quad \delta^*(q,w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p,w) \in F \quad \text{implies} \quad \delta^*(q,w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \in F,$$

and

$$\delta^*(p, w) \notin F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p,w) \in F \quad \text{implies} \quad \delta^*(q,w) \in F,$$

and

$$\delta^*(p,w) \notin F \quad \text{implies} \quad \delta^*(q,w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p,w) \in F \quad \text{implies} \quad \delta^*(q,w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \in F,$$

and

$$\delta^*(p, w) \notin F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \in F,$$

and

$$\delta^*(p, w) \notin F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p,w) \in F \quad \text{implies} \quad \delta^*(q,w) \in F,$$

and

$$\delta^*(p,w) \notin F \quad \text{implies} \quad \delta^*(q,w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p,w) \in F \quad \text{implies} \quad \delta^*(q,w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p,w) \in F \quad \text{implies} \quad \delta^*(q,w) \in F,$$

and

$$\delta^*(p,w) \notin F \quad \text{implies} \quad \delta^*(q,w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p,w) \in F \quad \text{implies} \quad \delta^*(q,w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \in F,$$

and

$$\delta^*(p, w) \notin F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Definition 2.8

Two states p and q of a DFA are called *indistinguishable* if

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \in F,$$

and

$$\delta^*(p, w) \notin F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

for all $w \in \Sigma^*$. If, on the other hand, there exists some string $w \in \Sigma^*$ such that

$$\delta^*(p, w) \in F \quad \text{implies} \quad \delta^*(q, w) \notin F,$$

or vice versa, then the states p and q are said to be *distinguishable* by a string w.

Clearly, two states are either indistinguishable or distinguishable. Indistinguishability has the properties of an equivalence relation: If p and q are indistinguishable and if q and r are also indistinguishable, then so are p and r, and all three states are indistinguishable.

One method for reducing the states of a DFA is based on finding and combining indistinguishable states. We first describe a method for finding pairs of distinguishable states.

Procedure: mark

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = p_a$ and $\delta(q, a) = q_a$. If the pair (p_a, q_a) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: ***mark***

1. Remove all inaccessible states.
2. Consider all pairs of states (p, q).
3. Repeat the following step until no previously unmarked pairs are marked.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: ***mark***

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = pa$ and $\delta(q, a) = qa$. If the pair (pa, qa) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: mark

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = pa$ and $\delta(q, a) = qa$. If the pair (pa, qa) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: mark

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = pa$ and $\delta(q, a) = qa$. If the pair (pa, qa) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: mark

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = p_a$ and $\delta(q, a) = q_a$. If the pair (p_a, q_a) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: mark

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = p_a$ and $\delta(q, a) = q_a$. If the pair (p_a, q_a) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: mark

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = p_a$ and $\delta(q, a) = q_a$. If the pair (p_a, q_a) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: ***mark***

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = pa$ and $\delta(q, a) = qa$. If the pair (pa, qa) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: mark

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = pa$ and $\delta(q, a) = qa$. If the pair (pa, qa) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: mark

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = pa$ and $\delta(q, a) = qa$. If the pair (pa, qa) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: mark

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = pa$ and $\delta(q, a) = qa$. If the pair (pa, qa) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: ***mark***

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = pa$ and $\delta(q, a) = qa$. If the pair (pa, qa) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: mark

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = pa$ and $\delta(q, a) = qa$. If the pair (pa, qa) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: mark

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = pa$ and $\delta(q, a) = qa$. If the pair (pa, qa) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: mark

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = pa$ and $\delta(q, a) = qa$. If the pair (pa, qa) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: mark

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = pa$ and $\delta(q, a) = qa$. If the pair (pa, qa) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: mark

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = pa$ and $\delta(q, a) = qa$. If the pair (pa, qa) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: mark

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = pa$ and $\delta(q, a) = qa$. If the pair (pa, qa) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Procedure: mark

1. Remove all inaccessible states. This can be done by enumerating all simple paths of the graph of the DFA starting at the initial state. Any state not part of some path is inaccessible.
2. Consider all pairs of states (p, q). If $p \in F$ and $q \notin F$ or vice versa, mark the pair (p, q) as distinguishable.
3. Repeat the following step until no previously unmarked pairs are marked. For all pairs (p, q) and all $a \in \Sigma$, compute $\delta(p, a) = pa$ and $\delta(q, a) = qa$. If the pair (pa, qa) is marked as distinguishable, mark (p, q) as distinguishable.

We claim that this procedure constitutes an algorithm for marking all distinguishable pairs.

Theorem 2.3

The procedure *mark*, applied to any DFA $M = (Q, \Sigma, \delta, q_0, F)$, terminates and determines all pairs of distinguishable states.

Proof. Obviously, the procedure terminates, because there are only a finite number of pairs that can be marked. It is also easy to see that the states of any pair so marked are distinguishable. The only claim that requires elaboration is that the procedure finds all distinguishable pairs.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n - 1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n - 1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n - 1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n + 1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n - 1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n - 1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n - 1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n + 1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ∎

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \dots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

2.4 Reduction of the Number of States in Finite Automata

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ∎

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

2.4 Reduction of the Number of States in Finite Automata

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

Note first that states q_i and q_j are distinguishable with a string of length n if and only if there are transitions

$$\delta(q_i, a) = q_k \tag{1}$$

and

$$\delta(q_j, a) = q_l, \tag{2}$$

for some $a \in \Sigma$, with q_k and q_l distinguishable by a string of length $n-1$. We use this first to show that at the completion of the nth pass through the loop in step 3, all states distinguishable by strings of length n or less have been marked. In step 2, we mark all pairs indistinguishable by λ, so we have a basis with $n = 0$ for induction. We now assume that the claim is true for all $i = 0, 1, \ldots, n-1$. By this inductive assumption, at the beginning of the nth pass through the loop, all states distinguishable by strings of length up to $n-1$ have been marked. Because of (1) and (2) above, at the end of this pass, all states distinguishable by strings of length up to n will be marked. By induction then, we can claim that, for any n, at the completion of the nth pass, all pairs distinguishable by strings of length n or less have been marked.

To show that this procedure marks all distinguishable states, assume that the loop terminates after n passes. This means that during the nth pass no new states were marked. From (1) and (2), it then follows that there cannot be any states distinguishable by a string of length n, but not distinguishable by any shorter string. But if there are no states distinguishable only by strings of length n, there cannot be any states distinguishable only by strings of length $n+1$, and so on. As a consequence, when the loop terminates, all distinguishable pairs have been marked. ■

The procedure mark can be implemented by partitioning the states into equivalence classes. Whenever two states are found to be distinguishable, they are immediately put into separate equivalence classes.

2.4 Reduction of the Number of States in Finite Automata

Example 2.15

Consider the automaton in the following Figure.

In the second step of procedure mark we partition the state set into final and nonfinal states to get two equivalence classes $\{q_0, q_1, q_3\}$ and $\{q_2, q_4\}$. In the next step, when we compute

$$\delta(q_0, 0) = q_1$$

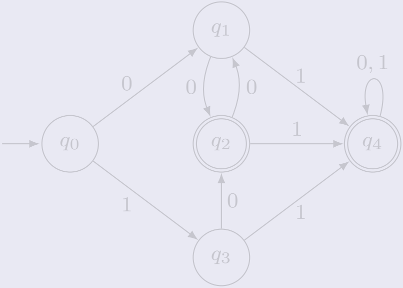
and

$$\delta(q_1, 0) = q_2,$$

we recognize that q_0 and q_1 are distinguishable, so we put them into different sets. So $\{q_0, q_1, q_3\}$ is split into $\{q_0\}$ and $\{q_1, q_3\}$. Also, since $\delta(q_2, 0) = q_3$ and $\delta(q_4, 0) = q_4$, the class $\{q_2, q_4\}$ is split into $\{q_2\}$ and $\{q_4\}$. The rest of the computations show that no further splitting is needed.

Example 2.15

Consider the automaton in the following Figure.



In the second step of procedure mark we partition the state set into final and nonfinal states to get two equivalence classes $\{q_0, q_1, q_3\}$ and $\{q_2, q_4\}$. In the next step, when we compute

$$\delta(q_0, 0) = q_1$$

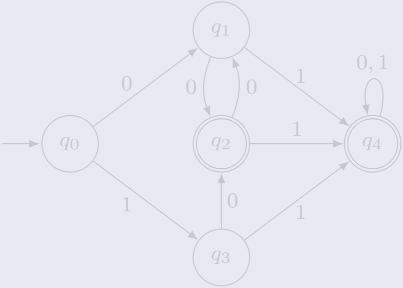
and

$$\delta(q_1, 0) = q_2,$$

we recognize that q_0 and q_1 are distinguishable, so we put them into different sets. So $\{q_0, q_1, q_3\}$ is split into $\{q_0\}$ and $\{q_1, q_3\}$. Also, since $\delta(q_2, 0) = q_3$ and $\delta(q_4, 0) = q_4$, the class $\{q_2, q_4\}$ is split into $\{q_2\}$ and $\{q_4\}$. The rest of the computations show that no further splitting is needed.

Example 2.15

Consider the automaton in the following Figure.



In the second step of procedure mark we partition the state set into final and nonfinal states to get two equivalence classes $\{q_0, q_1, q_3\}$ and $\{q_2, q_4\}$. In the next step, when we compute

$$\delta(q_0, 0) = q_1$$

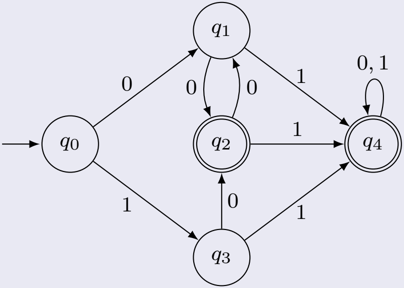
and

$$\delta(q_1, 0) = q_2,$$

we recognize that q_0 and q_1 are distinguishable, so we put them into different sets. So $\{q_0, q_1, q_3\}$ is split into $\{q_0\}$ and $\{q_1, q_3\}$. Also, since $\delta(q_2, 0) = q_3$ and $\delta(q_4, 0) = q_4$, the class $\{q_2, q_4\}$ is split into $\{q_2\}$ and $\{q_4\}$. The rest of the computations show that no further splitting is needed.

Example 2.15

Consider the automaton in the following Figure.



In the second step of procedure mark we partition the state set into final and nonfinal states to get two equivalence classes $\{q_0, q_1, q_3\}$ and $\{q_2, q_4\}$. In the next step, when we compute

$$\delta(q_0, 0) = q_1$$

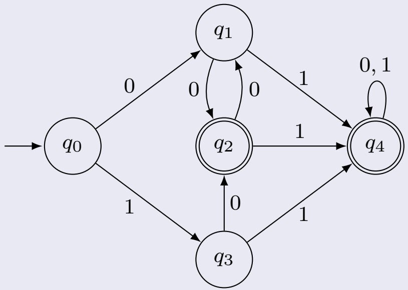
and

$$\delta(q_1, 0) = q_2,$$

we recognize that q_0 and q_1 are distinguishable, so we put them into different sets. So $\{q_0, q_1, q_3\}$ is split into $\{q_0\}$ and $\{q_1, q_3\}$. Also, since $\delta(q_2, 0) = q_3$ and $\delta(q_4, 0) = q_4$, the class $\{q_2, q_4\}$ is split into $\{q_2\}$ and $\{q_4\}$. The rest of the computations show that no further splitting is needed.

Example 2.15

Consider the automaton in the following Figure.



In the second step of procedure mark we partition the state set into final and nonfinal states to get two equivalence classes $\{q_0, q_1, q_3\}$ and $\{q_2, q_4\}$. In the next step, when we compute

$$\delta(q_0, 0) = q_1$$

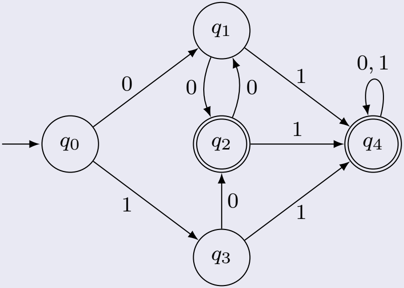
and

$$\delta(q_1, 0) = q_2,$$

we recognize that q_0 and q_1 are distinguishable, so we put them into different sets. So $\{q_0, q_1, q_3\}$ is split into $\{q_0\}$ and $\{q_1, q_3\}$. Also, since $\delta(q_2, 0) = q_3$ and $\delta(q_4, 0) = q_4$, the class $\{q_2, q_4\}$ is split into $\{q_2\}$ and $\{q_4\}$. The rest of the computations show that no further splitting is needed.

Example 2.15

Consider the automaton in the following Figure.



In the second step of procedure mark we partition the state set into final and nonfinal states to get two equivalence classes $\{q_0, q_1, q_3\}$ and $\{q_2, q_4\}$. In the next step, when we compute

$$\delta(q_0, 0) = q_1$$

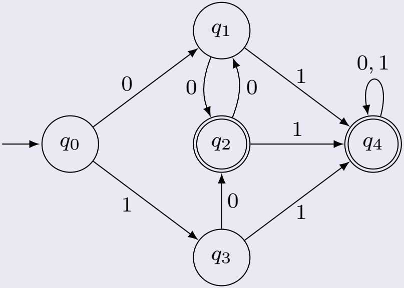
and

$$\delta(q_1, 0) = q_2,$$

we recognize that q_0 and q_1 are distinguishable, so we put them into different sets. So $\{q_0, q_1, q_3\}$ is split into $\{q_0\}$ and $\{q_1, q_3\}$. Also, since $\delta(q_2, 0) = q_3$ and $\delta(q_4, 0) = q_4$, the class $\{q_2, q_4\}$ is split into $\{q_2\}$ and $\{q_4\}$. The rest of the computations show that no further splitting is needed.

Example 2.15

Consider the automaton in the following Figure.



In the second step of procedure mark we partition the state set into final and nonfinal states to get two equivalence classes $\{q_0, q_1, q_3\}$ and $\{q_2, q_4\}$. In the next step, when we compute

$$\delta(q_0, 0) = q_1$$

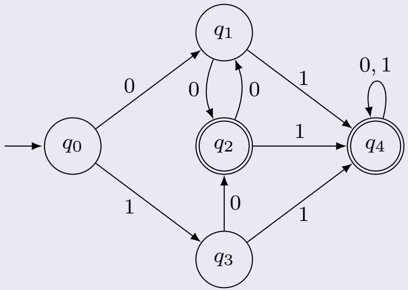
and

$$\delta(q_1, 0) = q_2,$$

we recognize that q_0 and q_1 are distinguishable, so we put them into different sets. So $\{q_0, q_1, q_3\}$ is split into $\{q_0\}$ and $\{q_1, q_3\}$. Also, since $\delta(q_2, 0) = q_3$ and $\delta(q_4, 0) = q_4$, the class $\{q_2, q_4\}$ is split into $\{q_2\}$ and $\{q_4\}$. The rest of the computations show that no further splitting is needed.

Example 2.15

Consider the automaton in the following Figure.



In the second step of procedure mark we partition the state set into final and nonfinal states to get two equivalence classes $\{q_0, q_1, q_3\}$ and $\{q_2, q_4\}$. In the next step, when we compute

$$\delta(q_0, 0) = q_1$$

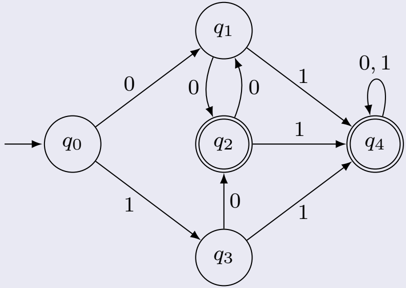
and

$$\delta(q_1, 0) = q_2,$$

we recognize that q_0 and q_1 are distinguishable, so we put them into different sets. So $\{q_0, q_1, q_3\}$ is split into $\{q_0\}$ and $\{q_1, q_3\}$. Also, since $\delta(q_2, 0) = q_3$ and $\delta(q_4, 0) = q_4$, the class $\{q_2, q_4\}$ is split into $\{q_2\}$ and $\{q_4\}$. The rest of the computations show that no further splitting is needed.

Example 2.15

Consider the automaton in the following Figure.



In the second step of procedure mark we partition the state set into final and nonfinal states to get two equivalence classes $\{q_0, q_1, q_3\}$ and $\{q_2, q_4\}$. In the next step, when we compute

$$\delta(q_0, 0) = q_1$$

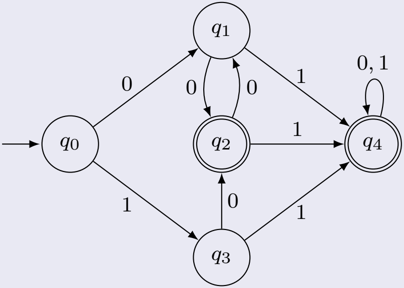
and

$$\delta(q_1, 0) = q_2,$$

we recognize that q_0 and q_1 are distinguishable, so we put them into different sets. So $\{q_0, q_1, q_3\}$ is split into $\{q_0\}$ and $\{q_1, q_3\}$. Also, since $\delta(q_2, 0) = q_3$ and $\delta(q_4, 0) = q_4$, the class $\{q_2, q_4\}$ is split into $\{q_2\}$ and $\{q_4\}$. The rest of the computations show that no further splitting is needed.

Example 2.15

Consider the automaton in the following Figure.



In the second step of procedure mark we partition the state set into final and nonfinal states to get two equivalence classes $\{q_0, q_1, q_3\}$ and $\{q_2, q_4\}$. In the next step, when we compute

$$\delta(q_0, 0) = q_1$$

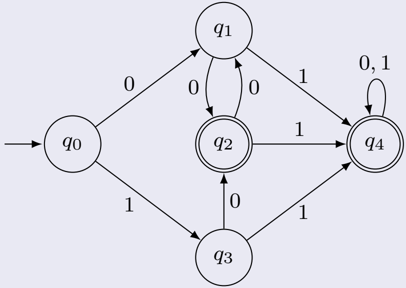
and

$$\delta(q_1, 0) = q_2,$$

we recognize that q_0 and q_1 are distinguishable, so we put them into different sets. So $\{q_0, q_1, q_3\}$ is split into $\{q_0\}$ and $\{q_1, q_3\}$. Also, since $\delta(q_2, 0) = q_3$ and $\delta(q_4, 0) = q_4$, the class $\{q_2, q_4\}$ is split into $\{q_2\}$ and $\{q_4\}$. The rest of the computations show that no further splitting is needed.

Example 2.15

Consider the automaton in the following Figure.



In the second step of procedure mark we partition the state set into final and nonfinal states to get two equivalence classes $\{q_0, q_1, q_3\}$ and $\{q_2, q_4\}$. In the next step, when we compute

$$\delta(q_0, 0) = q_1$$

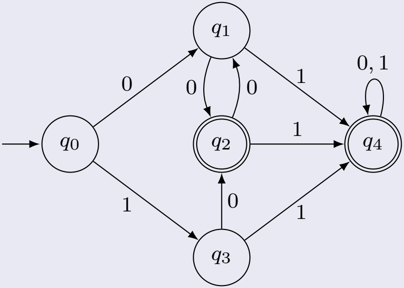
and

$$\delta(q_1, 0) = q_2,$$

we recognize that q_0 and q_1 are distinguishable, so we put them into different sets. So $\{q_0, q_1, q_3\}$ is split into $\{q_0\}$ and $\{q_1, q_3\}$. Also, since $\delta(q_2, 0) = q_3$ and $\delta(q_4, 0) = q_4$, the class $\{q_2, q_4\}$ is split into $\{q_2\}$ and $\{q_4\}$. The rest of the computations show that no further splitting is needed.

Example 2.15

Consider the automaton in the following Figure.



In the second step of procedure mark we partition the state set into final and nonfinal states to get two equivalence classes $\{q_0, q_1, q_3\}$ and $\{q_2, q_4\}$. In the next step, when we compute

$$\delta(q_0, 0) = q_1$$

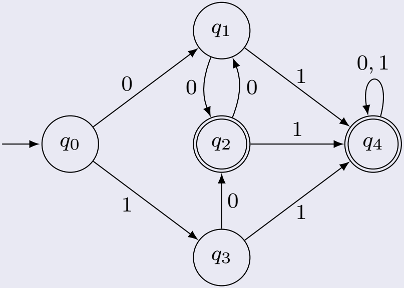
and

$$\delta(q_1, 0) = q_2,$$

we recognize that q_0 and q_1 are distinguishable, so we put them into different sets. So $\{q_0, q_1, q_3\}$ is split into $\{q_0\}$ and $\{q_1, q_3\}$. Also, since $\delta(q_2, 0) = q_3$ and $\delta(q_4, 0) = q_4$, the class $\{q_2, q_4\}$ is split into $\{q_2\}$ and $\{q_4\}$. The rest of the computations show that no further splitting is needed.

Example 2.15

Consider the automaton in the following Figure.



In the second step of procedure mark we partition the state set into final and nonfinal states to get two equivalence classes $\{q_0, q_1, q_3\}$ and $\{q_2, q_4\}$. In the next step, when we compute

$$\delta(q_0, 0) = q_1$$

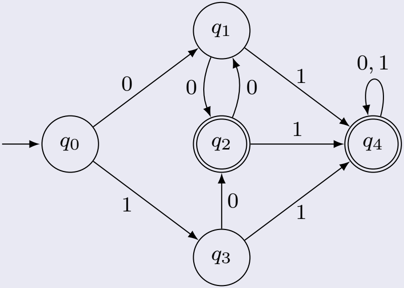
and

$$\delta(q_1, 0) = q_2,$$

we recognize that q_0 and q_1 are distinguishable, so we put them into different sets. So $\{q_0, q_1, q_3\}$ is split into $\{q_0\}$ and $\{q_1, q_3\}$. Also, since $\delta(q_2, 0) = q_3$ and $\delta(q_4, 0) = q_4$, the class $\{q_2, q_4\}$ is split into $\{q_2\}$ and $\{q_4\}$. The rest of the computations show that no further splitting is needed.

Example 2.15

Consider the automaton in the following Figure.



In the second step of procedure mark we partition the state set into final and nonfinal states to get two equivalence classes $\{q_0, q_1, q_3\}$ and $\{q_2, q_4\}$. In the next step, when we compute

$$\delta(q_0, 0) = q_1$$

and

$$\delta(q_1, 0) = q_2,$$

we recognize that q_0 and q_1 are distinguishable, so we put them into different sets. So $\{q_0, q_1, q_3\}$ is split into $\{q_0\}$ and $\{q_1, q_3\}$. Also, since $\delta(q_2, 0) = q_3$ and $\delta(q_4, 0) = q_4$, the class $\{q_2, q_4\}$ is split into $\{q_2\}$ and $\{q_4\}$. The rest of the computations show that no further splitting is needed.

Once the indistinguishability classes are found, the construction of the minimal DFA is straightforward.

Procedure: reduce

Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$, we construct a reduced DFA $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, \widehat{q_0}, \widehat{F})$ as follows.

1. Use procedure *mark* to generate the equivalence classes, say $\{q_i, q_j, \ldots, q_k\}$, as described.
2. For each set $\{q_i, q_j, \ldots, q_k\}$ of such indistinguishable states, create a state labeled by $ij \cdots k$ for $\widehat{M}$.
3. For each transition rule of M of the form
$$\delta(q_r, a) = q_p,$$
find the sets to which q_r and q_p belong. If $q_r \in \{q_i, q_j, \ldots, q_k\}$ and $q_p \in \{q_l, q_m, \ldots, q_n\}$, add to $\widehat{\delta}$ a rule
$$\widehat{\delta}(ij \cdots k, a) = lm \cdots n.$$
4. The initial state $\widehat{q_0}$ is that state of $\widehat{M}$ whose label includes the symbol 0.
5. $\widehat{F}$ is the set of all the states whose label contains i such that $q_i \in F$.

Once the indistinguishability classes are found, the construction of the minimal DFA is straightforward.

Procedure: reduce

Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$, we construct a reduced DFA $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, \widehat{q_0}, \widehat{F})$ as follows.

1. Use procedure *mark* to generate the equivalence classes, say $\{q_i, q_j, \ldots, q_k\}$, as described.
2. For each set $\{q_i, q_j, \ldots, q_k\}$ of such indistinguishable states, create a state labeled by $ij \cdots k$ for $\widehat{M}$.
3. For each transition rule of M of the form
$$\delta(q_r, a) = q_p,$$
find the sets to which q_r and q_p belong. If $q_r \in \{q_i, q_j, \ldots, q_k\}$ and $q_p \in \{q_l, q_m, \ldots, q_n\}$, add to $\widehat{\delta}$ a rule
$$\widehat{\delta}(ij \cdots k, a) = lm \cdots n.$$
4. The initial state $\widehat{q_0}$ is that state of $\widehat{M}$ whose label includes the symbol 0.
5. $\widehat{F}$ is the set of all the states whose label contains i such that $q_i \in F$.

Once the indistinguishability classes are found, the construction of the minimal DFA is straightforward.

Procedure: reduce

Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$, we construct a reduced DFA $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, \widehat{q_0}, \widehat{F})$ as follows.

1. Use procedure *mark* to generate the equivalence classes, say $\{q_i, q_j, \ldots, q_k\}$, as described.
2. For each set $\{q_i, q_j, \ldots, q_k\}$ of such indistinguishable states, create a state labeled by $ij \cdots k$ for $\widehat{M}$.
3. For each transition rule of M of the form
$$\delta(q_r, a) = q_p,$$
find the sets to which q_r and q_p belong. If $q_r \in \{q_i, q_j, \ldots, q_k\}$ and $q_p \in \{q_l, q_m, \ldots, q_n\}$, add to $\widehat{\delta}$ a rule
$$\widehat{\delta}(ij \cdots k, a) = lm \cdots n.$$
4. The initial state $\widehat{q_0}$ is that state of $\widehat{M}$ whose label includes the symbol 0.
5. $\widehat{F}$ is the set of all the states whose label contains i such that $q_i \in F$.

Once the indistinguishability classes are found, the construction of the minimal DFA is straightforward.

Procedure: reduce

Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$, we construct a reduced DFA $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, \widehat{q_0}, \widehat{F})$ as follows.

1. Use procedure *mark* to generate the equivalence classes, say $\{q_i, q_j, \ldots, q_k\}$, as described.
2. For each set $\{q_i, q_j, \ldots, q_k\}$ of such indistinguishable states, create a state labeled by $ij \cdots k$ for $\widehat{M}$.
3. For each transition rule of M of the form
$$\delta(q_r, a) = q_p,$$
find the sets to which q_r and q_p belong. If $q_r \in \{q_i, q_j, \ldots, q_k\}$ and $q_p \in \{q_l, q_m, \ldots, q_n\}$, add to $\widehat{\delta}$ a rule
$$\widehat{\delta}(ij \cdots k, a) = lm \cdots n.$$
4. The initial state $\widehat{q_0}$ is that state of $\widehat{M}$ whose label includes the symbol 0.
5. $\widehat{F}$ is the set of all the states whose label contains i such that $q_i \in F$.

Once the indistinguishability classes are found, the construction of the minimal DFA is straightforward.

Procedure: reduce

Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$, we construct a reduced DFA $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, \widehat{q_0}, \widehat{F})$ as follows.

1. Use procedure *mark* to generate the equivalence classes, say $\{q_i, q_j, \ldots, q_k\}$, as described.
2. For each set $\{q_i, q_j, \ldots, q_k\}$ of such indistinguishable states, create a state labeled by $ij \cdots k$ for $\widehat{M}$.
3. For each transition rule of M of the form
$$\delta(q_r, a) = q_p,$$
find the sets to which q_r and q_p belong. If $q_r \in \{q_i, q_j, \ldots, q_k\}$ and $q_p \in \{q_l, q_m, \ldots, q_n\}$, add to $\widehat{\delta}$ a rule
$$\widehat{\delta}(ij \cdots k, a) = lm \cdots n.$$
4. The initial state $\widehat{q_0}$ is that state of $\widehat{M}$ whose label includes the symbol 0.
5. $\widehat{F}$ is the set of all the states whose label contains i such that $q_i \in F$.

Once the indistinguishability classes are found, the construction of the minimal DFA is straightforward.

Procedure: reduce

Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$, we construct a reduced DFA $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, \widehat{q_0}, \widehat{F})$ as follows.

1. Use procedure *mark* to generate the equivalence classes, say $\{q_i, q_j, \ldots, q_k\}$, as described.
2. For each set $\{q_i, q_j, \ldots, q_k\}$ of such indistinguishable states, create a state labeled by $ij \cdots k$ for $\widehat{M}$.
3. For each transition rule of M of the form
$$\delta(q_r, a) = q_p,$$
find the sets to which q_r and q_p belong. If $q_r \in \{q_i, q_j, \ldots, q_k\}$ and $q_p \in \{q_l, q_m, \ldots, q_n\}$, add to $\widehat{\delta}$ a rule
$$\widehat{\delta}(ij \cdots k, a) = lm \cdots n.$$
4. The initial state $\widehat{q_0}$ is that state of $\widehat{M}$ whose label includes the symbol 0.
5. $\widehat{F}$ is the set of all the states whose label contains i such that $q_i \in F$.

Once the indistinguishability classes are found, the construction of the minimal DFA is straightforward.

Procedure: reduce

Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$, we construct a reduced DFA $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, \widehat{q_0}, \widehat{F})$ as follows.

1. Use procedure *mark* to generate the equivalence classes, say $\{q_i, q_j, \ldots, q_k\}$, as described.
2. For each set $\{q_i, q_j, \ldots, q_k\}$ of such indistinguishable states, create a state labeled by $ij \cdots k$ for $\widehat{M}$.
3. For each transition rule of M of the form
$$\delta(q_r, a) = q_p,$$
find the sets to which q_r and q_p belong. If $q_r \in \{q_i, q_j, \ldots, q_k\}$ and $q_p \in \{q_l, q_m, \ldots, q_n\}$, add to $\widehat{\delta}$ a rule
$$\widehat{\delta}(ij \cdots k, a) = lm \cdots n.$$
4. The initial state $\widehat{q_0}$ is that state of $\widehat{M}$ whose label includes the symbol 0.
5. $\widehat{F}$ is the set of all the states whose label contains i such that $q_i \in F$.

Once the indistinguishability classes are found, the construction of the minimal DFA is straightforward.

Procedure: reduce

Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$, we construct a reduced DFA $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, \widehat{q_0}, \widehat{F})$ as follows.

1. Use procedure *mark* to generate the equivalence classes, say $\{q_i, q_j, \ldots, q_k\}$, as described.
2. For each set $\{q_i, q_j, \ldots, q_k\}$ of such indistinguishable states, create a state labeled by $ij \cdots k$ for $\widehat{M}$.
3. For each transition rule of M of the form
$$\delta(q_r, a) = q_p,$$
find the sets to which q_r and q_p belong. If $q_r \in \{q_i, q_j, \ldots, q_k\}$ and $q_p \in \{q_l, q_m, \ldots, q_n\}$, add to $\widehat{\delta}$ a rule
$$\widehat{\delta}(ij \cdots k, a) = lm \cdots n.$$
4. The initial state $\widehat{q_0}$ is that state of $\widehat{M}$ whose label includes the symbol 0.
5. $\widehat{F}$ is the set of all the states whose label contains i such that $q_i \in F$.

Once the indistinguishability classes are found, the construction of the minimal DFA is straightforward.

Procedure: reduce

Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$, we construct a reduced DFA $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, \widehat{q_0}, \widehat{F})$ as follows.

1. Use procedure *mark* to generate the equivalence classes, say $\{q_i, q_j, \ldots, q_k\}$, as described.
2. For each set $\{q_i, q_j, \ldots, q_k\}$ of such indistinguishable states, create a state labeled by $ij \cdots k$ for $\widehat{M}$.
3. For each transition rule of M of the form
$$\delta(q_r, a) = q_p,$$
find the sets to which q_r and q_p belong. If $q_r \in \{q_i, q_j, \ldots, q_k\}$ and $q_p \in \{q_l, q_m, \ldots, q_n\}$, add to $\widehat{\delta}$ a rule
$$\widehat{\delta}(ij \cdots k, a) = lm \cdots n.$$
4. The initial state $\widehat{q_0}$ is that state of $\widehat{M}$ whose label includes the symbol 0.
5. $\widehat{F}$ is the set of all the states whose label contains i such that $q_i \in F$.

Once the indistinguishability classes are found, the construction of the minimal DFA is straightforward.

Procedure: reduce

Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$, we construct a reduced DFA $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, \widehat{q_0}, \widehat{F})$ as follows.

1. Use procedure *mark* to generate the equivalence classes, say $\{q_i, q_j, \ldots, q_k\}$, as described.
2. For each set $\{q_i, q_j, \ldots, q_k\}$ of such indistinguishable states, create a state labeled by $ij \cdots k$ for $\widehat{M}$.
3. For each transition rule of M of the form
$$\delta(q_r, a) = q_p,$$
find the sets to which q_r and q_p belong. If $q_r \in \{q_i, q_j, \ldots, q_k\}$ and $q_p \in \{q_l, q_m, \ldots, q_n\}$, add to $\widehat{\delta}$ a rule
$$\widehat{\delta}(ij \cdots k, a) = lm \cdots n.$$
4. The initial state $\widehat{q_0}$ is that state of $\widehat{M}$ whose label includes the symbol 0.
5. $\widehat{F}$ is the set of all the states whose label contains i such that $q_i \in F$.

Once the indistinguishability classes are found, the construction of the minimal DFA is straightforward.

Procedure: reduce

Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$, we construct a reduced DFA $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, \widehat{q_0}, \widehat{F})$ as follows.

1. Use procedure *mark* to generate the equivalence classes, say $\{q_i, q_j, \ldots, q_k\}$, as described.
2. For each set $\{q_i, q_j, \ldots, q_k\}$ of such indistinguishable states, create a state labeled by $ij \cdots k$ for $\widehat{M}$.
3. For each transition rule of M of the form
$$\delta(q_r, a) = q_p,$$
find the sets to which q_r and q_p belong. If $q_r \in \{q_i, q_j, \ldots, q_k\}$ and $q_p \in \{q_l, q_m, \ldots, q_n\}$, add to $\widehat{\delta}$ a rule
$$\widehat{\delta}(ij \cdots k, a) = lm \cdots n.$$
4. The initial state $\widehat{q_0}$ is that state of $\widehat{M}$ whose label includes the symbol 0.
5. $\widehat{F}$ is the set of all the states whose label contains i such that $q_i \in F$.

Once the indistinguishability classes are found, the construction of the minimal DFA is straightforward.

Procedure: reduce

Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$, we construct a reduced DFA $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, \widehat{q_0}, \widehat{F})$ as follows.

1. Use procedure *mark* to generate the equivalence classes, say $\{q_i, q_j, \ldots, q_k\}$, as described.
2. For each set $\{q_i, q_j, \ldots, q_k\}$ of such indistinguishable states, create a state labeled by $ij \cdots k$ for $\widehat{M}$.
3. For each transition rule of M of the form
$$\delta(q_r, a) = q_p,$$
find the sets to which q_r and q_p belong. If $q_r \in \{q_i, q_j, \ldots, q_k\}$ and $q_p \in \{q_l, q_m, \ldots, q_n\}$, add to $\widehat{\delta}$ a rule
$$\widehat{\delta}(ij \cdots k, a) = lm \cdots n.$$
4. The initial state $\widehat{q_0}$ is that state of $\widehat{M}$ whose label includes the symbol 0.
5. $\widehat{F}$ is the set of all the states whose label contains i such that $q_i \in F$.

Once the indistinguishability classes are found, the construction of the minimal DFA is straightforward.

Procedure: reduce

Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$, we construct a reduced DFA $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, \widehat{q_0}, \widehat{F})$ as follows.

1. Use procedure *mark* to generate the equivalence classes, say $\{q_i, q_j, \ldots, q_k\}$, as described.
2. For each set $\{q_i, q_j, \ldots, q_k\}$ of such indistinguishable states, create a state labeled by $ij \cdots k$ for $\widehat{M}$.
3. For each transition rule of M of the form
$$\delta(q_r, a) = q_p,$$
find the sets to which q_r and q_p belong. If $q_r \in \{q_i, q_j, \ldots, q_k\}$ and $q_p \in \{q_l, q_m, \ldots, q_n\}$, add to $\widehat{\delta}$ a rule
$$\widehat{\delta}(ij \cdots k, a) = lm \cdots n.$$
4. The initial state $\widehat{q_0}$ is that state of $\widehat{M}$ whose label includes the symbol 0.
5. $\widehat{F}$ is the set of all the states whose label contains i such that $q_i \in F$.

Once the indistinguishability classes are found, the construction of the minimal DFA is straightforward.

Procedure: reduce

Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$, we construct a reduced DFA $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, \widehat{q_0}, \widehat{F})$ as follows.

1. Use procedure *mark* to generate the equivalence classes, say $\{q_i, q_j, \ldots, q_k\}$, as described.
2. For each set $\{q_i, q_j, \ldots, q_k\}$ of such indistinguishable states, create a state labeled by $ij \cdots k$ for $\widehat{M}$.
3. For each transition rule of M of the form
$$\delta(q_r, a) = q_p,$$
find the sets to which q_r and q_p belong. If $q_r \in \{q_i, q_j, \ldots, q_k\}$ and $q_p \in \{q_l, q_m, \ldots, q_n\}$, add to $\widehat{\delta}$ a rule
$$\widehat{\delta}(ij \cdots k, a) = lm \cdots n.$$
4. The initial state $\widehat{q_0}$ is that state of $\widehat{M}$ whose label includes the symbol 0.
5. $\widehat{F}$ is the set of all the states whose label contains i such that $q_i \in F$.

Example 2.16

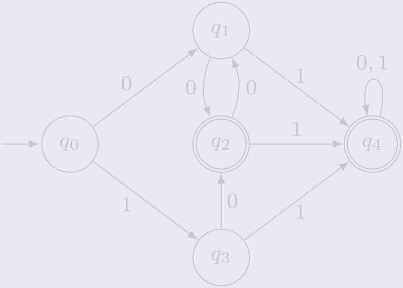
Continuing with Example 2.15,

we create the states in the following Figure.

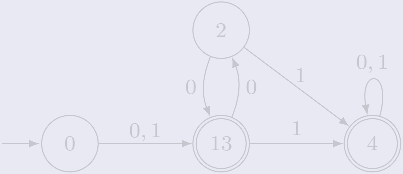
Since, for example, $\delta(q_1, 0) = q_2$, there is an edge labeled 0 from state 13 to state 2. The rest of the transitions are easily found, giving the minimal DFA in the Figure.

Example 2.16

Continuing with Example 2.15,



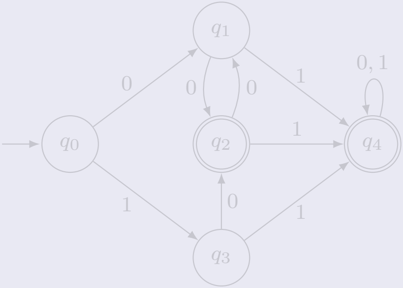
we create the states in the following Figure.



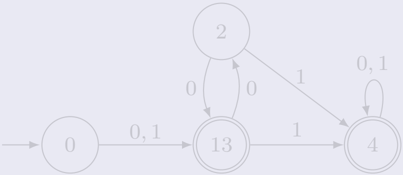
Since, for example, $\delta(q_1, 0) = q_2$, there is an edge labeled 0 from state 13 to state 2. The rest of the transitions are easily found, giving the minimal DFA in the Figure.

Example 2.16

Continuing with Example 2.15,



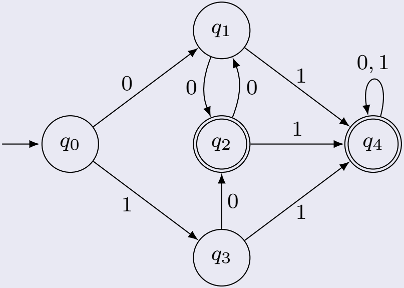
we create the states in the following Figure.



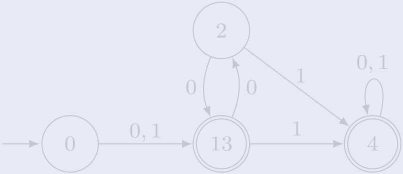
Since, for example, $\delta(q_1, 0) = q_2$, there is an edge labeled 0 from state 13 to state 2. The rest of the transitions are easily found, giving the minimal DFA in the Figure.

Example 2.16

Continuing with Example 2.15,



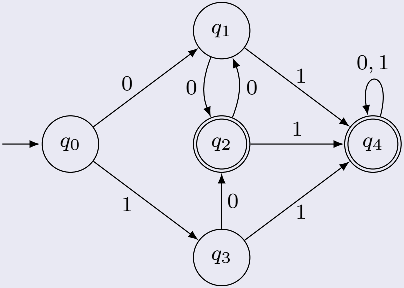
we create the states in the following Figure.



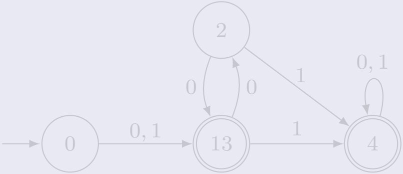
Since, for example, $\delta(q_1, 0) = q_2$, there is an edge labeled 0 from state 13 to state 2. The rest of the transitions are easily found, giving the minimal DFA in the Figure.

Example 2.16

Continuing with Example 2.15,



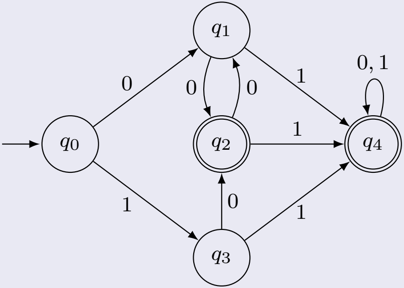
we create the states in the following Figure.



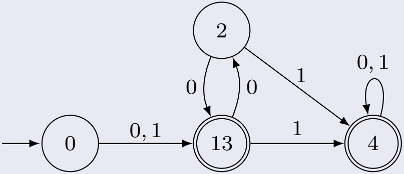
Since, for example, $\delta(q_1, 0) = q_2$, there is an edge labeled 0 from state 13 to state 2. The rest of the transitions are easily found, giving the minimal DFA in the Figure.

Example 2.16

Continuing with Example 2.15,



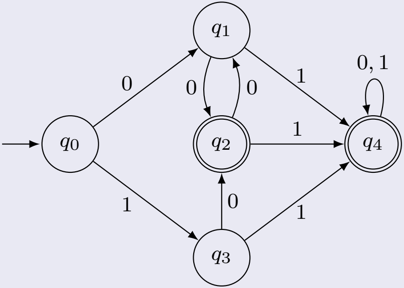
we create the states in the following Figure.



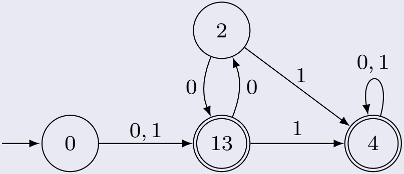
Since, for example, $\delta(q_1, 0) = q_2$, there is an edge labeled 0 from state 13 to state 2. The rest of the transitions are easily found, giving the minimal DFA in the Figure.

Example 2.16

Continuing with Example 2.15,



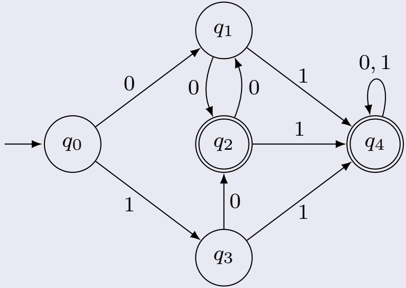
we create the states in the following Figure.



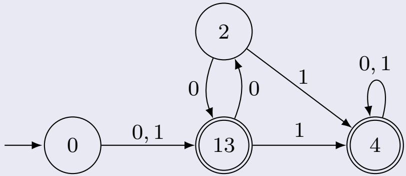
Since, for example, $\delta(q_1, 0) = q_2$, there is an edge labeled 0 from state 13 to state 2. The rest of the transitions are easily found, giving the minimal DFA in the Figure.

Example 2.16

Continuing with Example 2.15,



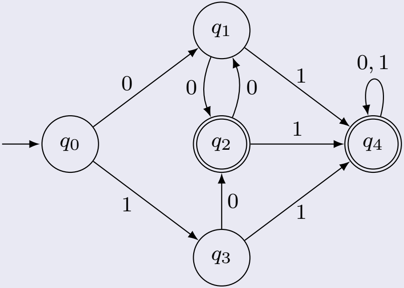
we create the states in the following Figure.



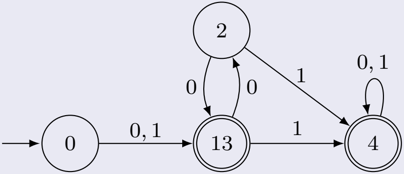
Since, for example, $\delta(q_1, 0) = q_2$, there is an edge labeled 0 from state 13 to state 2. The rest of the transitions are easily found, giving the minimal DFA in the Figure.

Example 2.16

Continuing with Example 2.15,



we create the states in the following Figure.



Since, for example, $\delta(q_1, 0) = q_2$, there is an edge labeled 0 from state 13 to state 2. The rest of the transitions are easily found, giving the minimal DFA in the Figure.

2.4 Reduction of the Number of States in Finite Automata

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that
$$L(M) = L(\widehat{M}).$$
Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that
$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$
But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that
$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that
$$L(M) = L(\widehat{M}).$$
Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that
$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$
But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that
$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta^*}(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta^*}(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta^*}(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta^*}(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Theorem 2.4

Given any DFA M, application of the procedure reduce yields another DFA $\widehat{M}$ such that

$$L(M) = L(\widehat{M}).$$

Furthermore, $\widehat{M}$ is minimal in the sense that there is no other DFA with a smaller number of states that also accepts $L(M)$.

Proof. There are two parts. The first is to show that the DFA created by reduce is equivalent to the original DFA. This is relatively easy and we can use inductive arguments similar to those used in establishing the equivalence of DFA's and NFA's. All we have to do is to show that $\delta^*(q_i, w) = q_j$ if and only if the label of $\widehat{\delta}^*(q_i, w)$ is of the form $\ldots j \ldots$. We shall leave this as an exercise.

The second part, to show that $\widehat{M}$ is minimal, is harder. Suppose $\widehat{M}$ has states $\{p_0, p_1, p_2, \ldots, p_m\}$, with p_0 the initial state. Assume that there is an equivalent DFA M_1, with transition function δ_1 and initial state q_0, which is equivalent to $\widehat{M}$, but with fewer states. Since there are no inaccessible states in $\widehat{M}$, there must be distinct strings $w_1, w_2, \ldots, w_m$ such that

$$\widehat{\delta}^*(p_0, w_i) = p_i, \quad i = 1, 2, \ldots, m.$$

But since M_1 has fewer states than $\widehat{M}$, there must be at least two of these strings, say w_k and w_l, such that

$$\delta_1^*(q_0, w_k) = \delta_1^*(q_0, w_l).$$

Since p_k and p_l are distinguishable, there must be some string x such that $\widehat{\delta}^*(p_0, w_k x) = \widehat{\delta}^*(p_k, x)$ is a final state, and $\widehat{\delta}^*(q_0, w_l x) = \widehat{\delta}^*(p_l, x)$ is a nonfinal state (or vice versa). In other words, $w_k x$ is accepted by $\widehat{M}$ and $w_l x$ is not. But note that

$$\begin{aligned}\widehat{\delta}_1^*(q_0, w_k x) &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_k), x) = \\ &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_l), x) = \\ &= \widehat{\delta}_1^*(q_0, w_l x).\end{aligned}$$

Thus, M_1 either accepts both $w_k x$ and $w_l x$ or rejects both, contradicting the assumption that $\widehat{M}$ and M_1 are equivalent. This contradiction proves that M_1 cannot exist. ■

Since p_k and p_l are distinguishable, there must be some string x such that $\widehat{\delta}^*(p_0, w_k x) = \widehat{\delta}^*(p_k, x)$ is a final state, and $\widehat{\delta}^*(q_0, w_l x) = \widehat{\delta}^*(p_l, x)$ is a nonfinal state (or vice versa). In other words, $w_k x$ is accepted by $\widehat{M}$ and $w_l x$ is not. But note that

$$\begin{aligned}\widehat{\delta}_1^*(q_0, w_k x) &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_k), x) = \\ &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_l), x) = \\ &= \widehat{\delta}_1^*(q_0, w_l x).\end{aligned}$$

Thus, M_1 either accepts both $w_k x$ and $w_l x$ or rejects both, contradicting the assumption that $\widehat{M}$ and M_1 are equivalent. This contradiction proves that M_1 cannot exist. ■

Since p_k and p_l are distinguishable, there must be some string x such that $\widehat{\delta}^*(p_0, w_k x) = \widehat{\delta}^*(p_k, x)$ is a final state, and $\widehat{\delta}^*(q_0, w_l x) = \widehat{\delta}^*(p_l, x)$ is a nonfinal state (or vice versa). In other words, $w_k x$ is accepted by $\widehat{M}$ and $w_l x$ is not. But note that

$$\begin{aligned}\widehat{\delta}_1^*(q_0, w_k x) &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_k), x) = \\ &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_l), x) = \\ &= \widehat{\delta}_1^*(q_0, w_l x).\end{aligned}$$

Thus, M_1 either accepts both $w_k x$ and $w_l x$ or rejects both, contradicting the assumption that $\widehat{M}$ and M_1 are equivalent. This contradiction proves that M_1 cannot exist. $\blacksquare$

Since p_k and p_l are distinguishable, there must be some string x such that $\widehat{\delta}^*(p_0, w_k x) = \widehat{\delta}^*(p_k, x)$ is a final state, and $\widehat{\delta}^*(q_0, w_l x) = \widehat{\delta}^*(p_l, x)$ is a nonfinal state (or vice versa). In other words, $w_k x$ is accepted by $\widehat{M}$ and $w_l x$ is not. But note that

$$\begin{aligned}\widehat{\delta}_1^*(q_0, w_k x) &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_k), x) = \\ &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_l), x) = \\ &= \widehat{\delta}_1^*(q_0, w_l x).\end{aligned}$$

Thus, M_1 either accepts both $w_k x$ and $w_l x$ or rejects both, contradicting the assumption that $\widehat{M}$ and M_1 are equivalent. This contradiction proves that M_1 cannot exist. ∎

Since p_k and p_l are distinguishable, there must be some string x such that $\widehat{\delta}^*(p_0, w_k x) = \widehat{\delta}^*(p_k, x)$ is a final state, and $\widehat{\delta}^*(q_0, w_l x) = \widehat{\delta}^*(p_l, x)$ is a nonfinal state (or vice versa). In other words, $w_k x$ is accepted by $\widehat{M}$ and $w_l x$ is not. But note that

$$\begin{aligned}\widehat{\delta}_1^*(q_0, w_k x) &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_k), x) = \\ &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_l), x) = \\ &= \widehat{\delta}_1^*(q_0, w_l x).\end{aligned}$$

Thus, M_1 either accepts both $w_k x$ and $w_l x$ or rejects both, contradicting the assumption that $\widehat{M}$ and M_1 are equivalent. This contradiction proves that M_1 cannot exist. ■

Since p_k and p_l are distinguishable, there must be some string x such that $\widehat{\delta}^*(p_0, w_k x) = \widehat{\delta}^*(p_k, x)$ is a final state, and $\widehat{\delta}^*(q_0, w_l x) = \widehat{\delta}^*(p_l, x)$ is a nonfinal state (or vice versa). In other words, $w_k x$ is accepted by $\widehat{M}$ and $w_l x$ is not. But note that

$$\begin{aligned}\widehat{\delta}_1^*(q_0, w_k x) &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_k), x) = \\ &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_l), x) = \\ &= \widehat{\delta}_1^*(q_0, w_l x).\end{aligned}$$

Thus, M_1 either accepts both $w_k x$ and $w_l x$ or rejects both, contradicting the assumption that $\widehat{M}$ and M_1 are equivalent. This contradiction proves that M_1 cannot exist. ∎

Since p_k and p_l are distinguishable, there must be some string x such that $\widehat{\delta}^*(p_0, w_k x) = \widehat{\delta}^*(p_k, x)$ is a final state, and $\widehat{\delta}^*(q_0, w_l x) = \widehat{\delta}^*(p_l, x)$ is a nonfinal state (or vice versa). In other words, $w_k x$ is accepted by $\widehat{M}$ and $w_l x$ is not. But note that

$$\begin{aligned} \widehat{\delta}_1^*(q_0, w_k x) &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_k), x) = \\ &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_l), x) = \\ &= \widehat{\delta}_1^*(q_0, w_l x). \end{aligned}$$

Thus, M_1 either accepts both $w_k x$ and $w_l x$ or rejects both, contradicting the assumption that $\widehat{M}$ and M_1 are equivalent. This contradiction proves that M_1 cannot exist. ■

Since p_k and p_l are distinguishable, there must be some string x such that $\widehat{\delta}^*(p_0, w_k x) = \widehat{\delta}^*(p_k, x)$ is a final state, and $\widehat{\delta}^*(q_0, w_l x) = \widehat{\delta}^*(p_l, x)$ is a nonfinal state (or vice versa). In other words, $w_k x$ is accepted by $\widehat{M}$ and $w_l x$ is not. But note that

$$\begin{aligned}\widehat{\delta}_1^*(q_0, w_k x) &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_k), x) = \\ &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_l), x) = \\ &= \widehat{\delta}_1^*(q_0, w_l x).\end{aligned}$$

Thus, M_1 either accepts both $w_k x$ and $w_l x$ or rejects both, contradicting the assumption that $\widehat{M}$ and M_1 are equivalent. This contradiction proves that M_1 cannot exist. ∎

Since p_k and p_l are distinguishable, there must be some string x such that $\widehat{\delta}^*(p_0, w_k x) = \widehat{\delta}^*(p_k, x)$ is a final state, and $\widehat{\delta}^*(q_0, w_l x) = \widehat{\delta}^*(p_l, x)$ is a nonfinal state (or vice versa). In other words, $w_k x$ is accepted by $\widehat{M}$ and $w_l x$ is not. But note that

$$\begin{aligned}\widehat{\delta}_1^*(q_0, w_k x) &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_k), x) = \\ &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_l), x) = \\ &= \widehat{\delta}_1^*(q_0, w_l x).\end{aligned}$$

Thus, M_1 either accepts both $w_k x$ and $w_l x$ or rejects both, contradicting the assumption that $\widehat{M}$ and M_1 are equivalent. This contradiction proves that M_1 cannot exist. $\blacksquare$

Since p_k and p_l are distinguishable, there must be some string x such that $\widehat{\delta}^*(p_0, w_k x) = \widehat{\delta}^*(p_k, x)$ is a final state, and $\widehat{\delta}^*(q_0, w_l x) = \widehat{\delta}^*(p_l, x)$ is a nonfinal state (or vice versa). In other words, $w_k x$ is accepted by $\widehat{M}$ and $w_l x$ is not. But note that

$$\begin{aligned} \widehat{\delta}_1^*(q_0, w_k x) &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_k), x) = \\ &= \widehat{\delta}_1^*(\widehat{\delta}_l^*(q_0, w_l), x) = \\ &= \widehat{\delta}_1^*(q_0, w_l x). \end{aligned}$$

Thus, M_1 either accepts both $w_k x$ and $w_l x$ or rejects both, contradicting the assumption that $\widehat{M}$ and M_1 are equivalent. This contradiction proves that M_1 cannot exist. ■

Thank You for attention!