Formal Languages, Automata and Codes

Oleg Gutik

Lecture 11

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Consider the following question: Given two regular languages L_1 and L_2, is their union also regular? In specific instances, the answer may be obvious, but here we want to address the problem in general. Is it true for all regular L_1 and L_2? It turns out that the answer is yes, a fact we express by saying that the family of regular languages is **closed** under union. We can ask similar questions about other types of operations on languages; this leads us to the study of the closure properties of languages in general.

Closure properties of various language families under different operations are of considerable theoretical interest. At first sight, it may not be clear what practical significance these properties have. Admittedly, some of them have very little, but many results are useful. By giving us insight into the general nature of language families, closure properties help us answer other, more practical questions. We shall see instances of this later in this couse of lectures.

Closure under Simple Set Operations

We begin by looking at the closure of regular languages under the common set operations, such as union and intersection.

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, L_1L_2, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, r_1r_2, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, L_1L_2, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in L$.

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 L_2$, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, $r_1 r_2$, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, $L_1 L_2$, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in L$.

4.1 Closure Properties of Regular Languages

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 L_2$, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, $r_1 r_2$, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, $L_1 L_2$, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in L$.

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 L_2$, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, $r_1 r_2$, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, $L_1 L_2$, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in L$.

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, L_1L_2, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, r_1r_2, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, L_1L_2, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in \overline{L}$.

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 L_2$, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, $r_1 r_2$, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, $L_1 L_2$, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in \overline{L}$.

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 L_2$, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, $r_1 r_2$, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, $L_1 L_2$, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in L$.

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 L_2$, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, $r_1 r_2$, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, $L_1 L_2$, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in L$.

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 L_2$, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, $r_1 r_2$, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, $L_1 L_2$, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in L$.

4.1 Closure Properties of Regular Languages

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, L_1L_2, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, r_1r_2, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, L_1L_2, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in L$.

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 L_2$, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, $r_1 r_2$, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, $L_1 L_2$, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in L$.

4.1 Closure Properties of Regular Languages

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 L_2$, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, $r_1 r_2$, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, $L_1 L_2$, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in L$.

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 L_2$, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, $r_1 r_2$, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, $L_1 L_2$, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in L$.

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, L_1L_2, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, r_1r_2, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, L_1L_2, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in L$.

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 L_2$, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, $r_1 r_2$, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, $L_1 L_2$, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in \overline{L}$.

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, L_1L_2, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, r_1r_2, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, L_1L_2, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in \overline{L}$.

Theorem 4.1

If L_1 and L_2 are regular languages, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, $L_1 L_2$, $\overline{L_1}$, and L_1^*. We say that the family of regular languages is closed under union, intersection, concatenation, complementation, and star-closure.

Proof. If L_1 and L_2 are regular, then there exist regular expressions r_1 and r_2 such that $L_1 = L(r_1)$ and $L_2 = L(r_2)$. By definition, $r_1 + r_2$, $r_1 r_2$, and r_1^* are regular expressions denoting the languages $L_1 \cup L_2$, $L_1 L_2$, and L_1^*, respectively. Thus, closure under union, concatenation, and star-closure is immediate.

To show closure under complementation, let $M = (Q, \Sigma, \delta, q_0, F)$ be a dfa that accepts L_1. Then the dfa

$$\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$$

accepts $\overline{L_1}$. This is rather straightforward. Note that in the definition of a dfa, we assumed δ^* to be a total function, so that $\delta^*(q_0, w)$ is defined for all $w \in \Sigma^*$. Consequently either $\delta^*(q_0, w)$ is a final state, in which case $w \in L$, or $\delta^*(q_0, w) \in Q - F$ and $w \in L$.

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ■

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ■

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ∎

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ■

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ■

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ■

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ■

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ■

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ■

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ■

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ■

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ■

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ■

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ■

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ■

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ∎

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ■

Demonstrating closure under intersection takes a little more work. Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$, where $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$ and $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$ are dfa's. We construct from M_1 and M_2 a combined automaton $\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\delta}, (q_0, p_0), \widehat{F})$, whose state set $\widehat{Q} = Q \times P$ consists of pairs (q_i, p_j), and whose transition function $\widehat{\delta}$ is such that $\widehat{M}$ is in state (q_i, p_j) whenever M_1 is in state q_i and M_2 is in state p_j. This is achieved by taking

$$\widehat{\delta}((q_i, p_j), a) = (q_k, p_l),$$

whenever

$$\delta_1(q_i, a) = q_k$$

and

$$\delta_2(p_j, a) = p_l.$$

$\widehat{F}$ is defined as the set of all (q_i, pj), such that $q_i \in F_1$ and $p_j \in F_2$. Then it is a simple matter to show that $w \in L_1 \cap L_2$ if and only if it is accepted by $\widehat{M}$. Consequently, $L_1 \cap L_2$ is regular. ■

The proof of closure under intersection is a good example of a constructive proof. Not only does it establish the desired result, but it also shows explicitly how to construct a finite accepter for the intersection of two regular languages. Constructive proofs occur throughout this course of lectures; they are important because they give us insight into the results and often serve as the starting point for practical algorithms. Here, as in many cases, there are shorter but nonconstructive (or at least not so obviously constructive) arguments. For closure under intersection, we start with DeMorgan's law, taking the complement of both sides. Then

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

for any languages L_1 and L_2. Now, if L_1 and L_2 are regular, then by closure under complementation, so are $\overline{L_1}$ and $\overline{L_2}$. Using closure under union, we next get that $\overline{L_1} \cup \overline{L_2}$ is regular. Using closure under complementation once more, we see that

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$

is regular.

The proof of closure under intersection is a good example of a constructive proof. Not only does it establish the desired result, but it also shows explicitly how to construct a finite accepter for the intersection of two regular languages. Constructive proofs occur throughout this course of lectures; they are important because they give us insight into the results and often serve as the starting point for practical algorithms. Here, as in many cases, there are shorter but nonconstructive (or at least not so obviously constructive) arguments. For closure under intersection, we start with DeMorgan's law, taking the complement of both sides. Then

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

for any languages L_1 and L_2. Now, if L_1 and L_2 are regular, then by closure under complementation, so are $\overline{L_1}$ and $\overline{L_2}$. Using closure under union, we next get that $\overline{L_1} \cup \overline{L_2}$ is regular. Using closure under complementation once more, we see that

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$

is regular.

The proof of closure under intersection is a good example of a constructive proof. Not only does it establish the desired result, but it also shows explicitly how to construct a finite accepter for the intersection of two regular languages. Constructive proofs occur throughout this course of lectures; they are important because they give us insight into the results and often serve as the starting point for practical algorithms. Here, as in many cases, there are shorter but nonconstructive (or at least not so obviously constructive) arguments. For closure under intersection, we start with DeMorgan's law, taking the complement of both sides. Then

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

for any languages L_1 and L_2. Now, if L_1 and L_2 are regular, then by closure under complementation, so are $\overline{L_1}$ and $\overline{L_2}$. Using closure under union, we next get that $\overline{L_1} \cup \overline{L_2}$ is regular. Using closure under complementation once more, we see that

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$

is regular.

The proof of closure under intersection is a good example of a constructive proof. Not only does it establish the desired result, but it also shows explicitly how to construct a finite accepter for the intersection of two regular languages. Constructive proofs occur throughout this course of lectures; they are important because they give us insight into the results and often serve as the starting point for practical algorithms. Here, as in many cases, there are shorter but nonconstructive (or at least not so obviously constructive) arguments. For closure under intersection, we start with DeMorgan's law, taking the complement of both sides. Then

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

for any languages L_1 and L_2. Now, if L_1 and L_2 are regular, then by closure under complementation, so are $\overline{L_1}$ and $\overline{L_2}$. Using closure under union, we next get that $\overline{L_1} \cup \overline{L_2}$ is regular. Using closure under complementation once more, we see that

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$

is regular.

The proof of closure under intersection is a good example of a constructive proof. Not only does it establish the desired result, but it also shows explicitly how to construct a finite accepter for the intersection of two regular languages. Constructive proofs occur throughout this course of lectures; they are important because they give us insight into the results and often serve as the starting point for practical algorithms. Here, as in many cases, there are shorter but nonconstructive (or at least not so obviously constructive) arguments. For closure under intersection, we start with DeMorgan's law, taking the complement of both sides. Then

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

for any languages L_1 and L_2. Now, if L_1 and L_2 are regular, then by closure under complementation, so are $\overline{L_1}$ and $\overline{L_2}$. Using closure under union, we next get that $\overline{L_1} \cup \overline{L_2}$ is regular. Using closure under complementation once more, we see that

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$

is regular.

The proof of closure under intersection is a good example of a constructive proof. Not only does it establish the desired result, but it also shows explicitly how to construct a finite accepter for the intersection of two regular languages. Constructive proofs occur throughout this course of lectures; they are important because they give us insight into the results and often serve as the starting point for practical algorithms. Here, as in many cases, there are shorter but nonconstructive (or at least not so obviously constructive) arguments. For closure under intersection, we start with DeMorgan's law, taking the complement of both sides. Then

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

for any languages L_1 and L_2. Now, if L_1 and L_2 are regular, then by closure under complementation, so are $\overline{L_1}$ and $\overline{L_2}$. Using closure under union, we next get that $\overline{L_1} \cup \overline{L_2}$ is regular. Using closure under complementation once more, we see that

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$

is regular.

The proof of closure under intersection is a good example of a constructive proof. Not only does it establish the desired result, but it also shows explicitly how to construct a finite accepter for the intersection of two regular languages. Constructive proofs occur throughout this course of lectures; they are important because they give us insight into the results and often serve as the starting point for practical algorithms. Here, as in many cases, there are shorter but nonconstructive (or at least not so obviously constructive) arguments. For closure under intersection, we start with DeMorgan's law, taking the complement of both sides. Then

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

for any languages L_1 and L_2. Now, if L_1 and L_2 are regular, then by closure under complementation, so are $\overline{L_1}$ and $\overline{L_2}$. Using closure under union, we next get that $\overline{L_1} \cup \overline{L_2}$ is regular. Using closure under complementation once more, we see that

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$

is regular.

The proof of closure under intersection is a good example of a constructive proof. Not only does it establish the desired result, but it also shows explicitly how to construct a finite accepter for the intersection of two regular languages. Constructive proofs occur throughout this course of lectures; they are important because they give us insight into the results and often serve as the starting point for practical algorithms. Here, as in many cases, there are shorter but nonconstructive (or at least not so obviously constructive) arguments. For closure under intersection, we start with DeMorgan's law, taking the complement of both sides. Then

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

for any languages L_1 and L_2. Now, if L_1 and L_2 are regular, then by closure under complementation, so are $\overline{L_1}$ and $\overline{L_2}$. Using closure under union, we next get that $\overline{L_1} \cup \overline{L_2}$ is regular. Using closure under complementation once more, we see that

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$

is regular.

The proof of closure under intersection is a good example of a constructive proof. Not only does it establish the desired result, but it also shows explicitly how to construct a finite accepter for the intersection of two regular languages. Constructive proofs occur throughout this course of lectures; they are important because they give us insight into the results and often serve as the starting point for practical algorithms. Here, as in many cases, there are shorter but nonconstructive (or at least not so obviously constructive) arguments. For closure under intersection, we start with DeMorgan's law, taking the complement of both sides. Then

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

for any languages L_1 and L_2. Now, if L_1 and L_2 are regular, then by closure under complementation, so are $\overline{L_1}$ and $\overline{L_2}$. Using closure under union, we next get that $\overline{L_1} \cup \overline{L_2}$ is regular. Using closure under complementation once more, we see that

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$

is regular.

The proof of closure under intersection is a good example of a constructive proof. Not only does it establish the desired result, but it also shows explicitly how to construct a finite accepter for the intersection of two regular languages. Constructive proofs occur throughout this course of lectures; they are important because they give us insight into the results and often serve as the starting point for practical algorithms. Here, as in many cases, there are shorter but nonconstructive (or at least not so obviously constructive) arguments. For closure under intersection, we start with DeMorgan's law, taking the complement of both sides. Then

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

for any languages L_1 and L_2. Now, if L_1 and L_2 are regular, then by closure under complementation, so are $\overline{L_1}$ and $\overline{L_2}$. Using closure under union, we next get that $\overline{L_1} \cup \overline{L_2}$ is regular. Using closure under complementation once more, we see that

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$

is regular.

The proof of closure under intersection is a good example of a constructive proof. Not only does it establish the desired result, but it also shows explicitly how to construct a finite accepter for the intersection of two regular languages. Constructive proofs occur throughout this course of lectures; they are important because they give us insight into the results and often serve as the starting point for practical algorithms. Here, as in many cases, there are shorter but nonconstructive (or at least not so obviously constructive) arguments. For closure under intersection, we start with DeMorgan's law, taking the complement of both sides. Then

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

for any languages L_1 and L_2. Now, if L_1 and L_2 are regular, then by closure under complementation, so are $\overline{L_1}$ and $\overline{L_2}$. Using closure under union, we next get that $\overline{L_1} \cup \overline{L_2}$ is regular. Using closure under complementation once more, we see that

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$

is regular.

The proof of closure under intersection is a good example of a constructive proof. Not only does it establish the desired result, but it also shows explicitly how to construct a finite accepter for the intersection of two regular languages. Constructive proofs occur throughout this course of lectures; they are important because they give us insight into the results and often serve as the starting point for practical algorithms. Here, as in many cases, there are shorter but nonconstructive (or at least not so obviously constructive) arguments. For closure under intersection, we start with DeMorgan's law, taking the complement of both sides. Then

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

for any languages L_1 and L_2. Now, if L_1 and L_2 are regular, then by closure under complementation, so are $\overline{L_1}$ and $\overline{L_2}$. Using closure under union, we next get that $\overline{L_1} \cup \overline{L_2}$ is regular. Using closure under complementation once more, we see that

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$

is regular.

The proof of closure under intersection is a good example of a constructive proof. Not only does it establish the desired result, but it also shows explicitly how to construct a finite accepter for the intersection of two regular languages. Constructive proofs occur throughout this course of lectures; they are important because they give us insight into the results and often serve as the starting point for practical algorithms. Here, as in many cases, there are shorter but nonconstructive (or at least not so obviously constructive) arguments. For closure under intersection, we start with DeMorgan's law, taking the complement of both sides. Then

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

for any languages L_1 and L_2. Now, if L_1 and L_2 are regular, then by closure under complementation, so are $\overline{L_1}$ and $\overline{L_2}$. Using closure under union, we next get that $\overline{L_1} \cup \overline{L_2}$ is regular. Using closure under complementation once more, we see that

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$

is regular.

The proof of closure under intersection is a good example of a constructive proof. Not only does it establish the desired result, but it also shows explicitly how to construct a finite accepter for the intersection of two regular languages. Constructive proofs occur throughout this course of lectures; they are important because they give us insight into the results and often serve as the starting point for practical algorithms. Here, as in many cases, there are shorter but nonconstructive (or at least not so obviously constructive) arguments. For closure under intersection, we start with DeMorgan's law, taking the complement of both sides. Then

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

for any languages L_1 and L_2. Now, if L_1 and L_2 are regular, then by closure under complementation, so are $\overline{L_1}$ and $\overline{L_2}$. Using closure under union, we next get that $\overline{L_1} \cup \overline{L_2}$ is regular. Using closure under complementation once more, we see that

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$

is regular.

The proof of closure under intersection is a good example of a constructive proof. Not only does it establish the desired result, but it also shows explicitly how to construct a finite accepter for the intersection of two regular languages. Constructive proofs occur throughout this course of lectures; they are important because they give us insight into the results and often serve as the starting point for practical algorithms. Here, as in many cases, there are shorter but nonconstructive (or at least not so obviously constructive) arguments. For closure under intersection, we start with DeMorgan's law, taking the complement of both sides. Then

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

for any languages L_1 and L_2. Now, if L_1 and L_2 are regular, then by closure under complementation, so are $\overline{L_1}$ and $\overline{L_2}$. Using closure under union, we next get that $\overline{L_1} \cup \overline{L_2}$ is regular. Using closure under complementation once more, we see that

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$

is regular.

The proof of closure under intersection is a good example of a constructive proof. Not only does it establish the desired result, but it also shows explicitly how to construct a finite accepter for the intersection of two regular languages. Constructive proofs occur throughout this course of lectures; they are important because they give us insight into the results and often serve as the starting point for practical algorithms. Here, as in many cases, there are shorter but nonconstructive (or at least not so obviously constructive) arguments. For closure under intersection, we start with DeMorgan's law, taking the complement of both sides. Then

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

for any languages L_1 and L_2. Now, if L_1 and L_2 are regular, then by closure under complementation, so are $\overline{L_1}$ and $\overline{L_2}$. Using closure under union, we next get that $\overline{L_1} \cup \overline{L_2}$ is regular. Using closure under complementation once more, we see that

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$$

is regular.

The following example is a variation on the same idea.

Example 4.1

Show that the family of regular languages is closed under difference. In other words, we want to show that if L_1 and L_2 are regular, then $L_1 - L_2$ is necessarily regular also.

The needed set identity is immediately obvious from the definition of a set difference, namely

$$L_1 - L_2 = L_1 \cap \overline{L_2}.$$

The fact that L_2 is regular implies that $\overline{L_2}$ is also regular. Then, because of the closure of regular languages under intersection, we know that $L_1 \cap \overline{L_2}$ is regular, and the argument is complete.

A variety of other closure properties can be derived directly by elementary arguments.

The following example is a variation on the same idea.

Example 4.1

Show that the family of regular languages is closed under difference. In other words, we want to show that if L_1 and L_2 are regular, then $L_1 - L_2$ is necessarily regular also.

The needed set identity is immediately obvious from the definition of a set difference, namely

$$L_1 - L_2 = L_1 \cap \overline{L_2}.$$

The fact that L_2 is regular implies that $\overline{L_2}$ is also regular. Then, because of the closure of regular languages under intersection, we know that $L_1 \cap \overline{L_2}$ is regular, and the argument is complete.

A variety of other closure properties can be derived directly by elementary arguments.

The following example is a variation on the same idea.

Example 4.1

Show that the family of regular languages is closed under difference. In other words, we want to show that if L_1 and L_2 are regular, then $L_1 - L_2$ is necessarily regular also.

The needed set identity is immediately obvious from the definition of a set difference, namely

$$L_1 - L_2 = L_1 \cap \overline{L_2}.$$

The fact that L_2 is regular implies that $\overline{L_2}$ is also regular. Then, because of the closure of regular languages under intersection, we know that $L_1 \cap \overline{L_2}$ is regular, and the argument is complete.

A variety of other closure properties can be derived directly by elementary arguments.

The following example is a variation on the same idea.

Example 4.1

Show that the family of regular languages is closed under difference. In other words, we want to show that if L_1 and L_2 are regular, then $L_1 - L_2$ is necessarily regular also.

The needed set identity is immediately obvious from the definition of a set difference, namely

$$L_1 - L_2 = L_1 \cap \overline{L_2}.$$

The fact that L_2 is regular implies that $\overline{L_2}$ is also regular. Then, because of the closure of regular languages under intersection, we know that $L_1 \cap \overline{L_2}$ is regular, and the argument is complete.

A variety of other closure properties can be derived directly by elementary arguments.

The following example is a variation on the same idea.

Example 4.1

Show that the family of regular languages is closed under difference. In other words, we want to show that if L_1 and L_2 are regular, then $L_1 - L_2$ is necessarily regular also.

The needed set identity is immediately obvious from the definition of a set difference, namely

$$L_1 - L_2 = L_1 \cap \overline{L_2}.$$

The fact that L_2 is regular implies that $\overline{L_2}$ is also regular. Then, because of the closure of regular languages under intersection, we know that $L_1 \cap \overline{L_2}$ is regular, and the argument is complete.

A variety of other closure properties can be derived directly by elementary arguments.

The following example is a variation on the same idea.

Example 4.1

Show that the family of regular languages is closed under difference. In other words, we want to show that if L_1 and L_2 are regular, then $L_1 - L_2$ is necessarily regular also.

The needed set identity is immediately obvious from the definition of a set difference, namely

$$L_1 - L_2 = L_1 \cap \overline{L_2}.$$

The fact that L_2 is regular implies that $\overline{L_2}$ is also regular. Then, because of the closure of regular languages under intersection, we know that $L_1 \cap \overline{L_2}$ is regular, and the argument is complete.

A variety of other closure properties can be derived directly by elementary arguments.

The following example is a variation on the same idea.

Example 4.1

Show that the family of regular languages is closed under difference. In other words, we want to show that if L_1 and L_2 are regular, then $L_1 - L_2$ is necessarily regular also.

The needed set identity is immediately obvious from the definition of a set difference, namely

$$L_1 - L_2 = L_1 \cap \overline{L_2}.$$

The fact that L_2 is regular implies that $\overline{L_2}$ is also regular. Then, because of the closure of regular languages under intersection, we know that $L_1 \cap \overline{L_2}$ is regular, and the argument is complete.

A variety of other closure properties can be derived directly by elementary arguments.

The following example is a variation on the same idea.

Example 4.1

Show that the family of regular languages is closed under difference. In other words, we want to show that if L_1 and L_2 are regular, then $L_1 - L_2$ is necessarily regular also.

The needed set identity is immediately obvious from the definition of a set difference, namely

$$L_1 - L_2 = L_1 \cap \overline{L_2}.$$

The fact that L_2 is regular implies that $\overline{L_2}$ is also regular. Then, because of the closure of regular languages under intersection, we know that $L_1 \cap \overline{L_2}$ is regular, and the argument is complete.

A variety of other closure properties can be derived directly by elementary arguments.

The following example is a variation on the same idea.

Example 4.1

Show that the family of regular languages is closed under difference. In other words, we want to show that if L_1 and L_2 are regular, then $L_1 - L_2$ is necessarily regular also.

The needed set identity is immediately obvious from the definition of a set difference, namely

$$L_1 - L_2 = L_1 \cap \overline{L_2}.$$

The fact that L_2 is regular implies that $\overline{L_2}$ is also regular. Then, because of the closure of regular languages under intersection, we know that $L_1 \cap \overline{L_2}$ is regular, and the argument is complete.

A variety of other closure properties can be derived directly by elementary arguments.

The following example is a variation on the same idea.

Example 4.1

Show that the family of regular languages is closed under difference. In other words, we want to show that if L_1 and L_2 are regular, then $L_1 - L_2$ is necessarily regular also.

The needed set identity is immediately obvious from the definition of a set difference, namely

$$L_1 - L_2 = L_1 \cap \overline{L_2}.$$

The fact that L_2 is regular implies that $\overline{L_2}$ is also regular. Then, because of the closure of regular languages under intersection, we know that $L_1 \cap \overline{L_2}$ is regular, and the argument is complete.

A variety of other closure properties can be derived directly by elementary arguments.

The following example is a variation on the same idea.

Example 4.1

Show that the family of regular languages is closed under difference. In other words, we want to show that if L_1 and L_2 are regular, then $L_1 - L_2$ is necessarily regular also.

The needed set identity is immediately obvious from the definition of a set difference, namely

$$L_1 - L_2 = L_1 \cap \overline{L_2}.$$

The fact that L_2 is regular implies that $\overline{L_2}$ is also regular. Then, because of the closure of regular languages under intersection, we know that $L_1 \cap \overline{L_2}$ is regular, and the argument is complete.

A variety of other closure properties can be derived directly by elementary arguments.

The following example is a variation on the same idea.

Example 4.1

Show that the family of regular languages is closed under difference. In other words, we want to show that if L_1 and L_2 are regular, then $L_1 - L_2$ is necessarily regular also.

The needed set identity is immediately obvious from the definition of a set difference, namely

$$L_1 - L_2 = L_1 \cap \overline{L_2}.$$

The fact that L_2 is regular implies that $\overline{L_2}$ is also regular. Then, because of the closure of regular languages under intersection, we know that $L_1 \cap \overline{L_2}$ is regular, and the argument is complete.

A variety of other closure properties can be derived directly by elementary arguments.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ■

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ∎

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ■

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ∎

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ■

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ∎

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ∎

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ■

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ∎

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ■

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ■

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ∎

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ■

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ■

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ■

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ■

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ■

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Theorem 4.2

The family of regular languages is closed under reversal.

Proof. Suppose that L is a regular language. We then construct an nfa with a single final state for it. In the previous lectures we show that this is always possible. In the transition graph for this nfa we make the initial vertex a final vertex, the final vertex the initial vertex, and reverse the direction on all the edges. It is a fairly straightforward matter to show that the modified nfa accepts w^R if and only if the original nfa accepts w. Therefore, the modified nfa accepts L^R, proving closure under reversal. ■

Closure under Other Operations

In addition to the standard operations on languages, one can define other operations and investigate closure properties for them. There are many such results; we select only two typical ones.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w) \colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w)\colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w) \colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1) h(a_2) \cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w)\colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w) \colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w)\colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w)\colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w) \colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w) \colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

4.1 Closure Properties of Regular Languages

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w) \colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w)\colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w)\colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w) \colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w)\colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w)\colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w)\colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w)\colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w)\colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function
$$h\colon \Sigma \to \Gamma^*$$
is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if
$$w = a_1 a_2 \cdots a_n,$$
then
$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$
If L is a language on Σ, then its homomorphic image is defined as
$$h(L) = \{h(w)\colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by
$$h(a) = ab,$$
$$h(b) = bbc.$$
Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

Definition 4.1

Suppose Σ and Γ are alphabets. Then a function

$$h\colon \Sigma \to \Gamma^*$$

is called a *homomorphism*. In words, a homomorphism is a substitution in which a single letter is replaced with a string. The domain of the function h is extended to strings in an obvious fashion; if

$$w = a_1 a_2 \cdots a_n,$$

then

$$h(w) = h(a_1)h(a_2)\cdots h(a_n).$$

If L is a language on Σ, then its homomorphic image is defined as

$$h(L) = \{h(w)\colon w \in L\}.$$

Example 4.2

Let $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, c\}$ and define h by

$$h(a) = ab,$$

$$h(b) = bbc.$$

Then $h(aba) = abbbcab$. The homomorphic image of $L = \{aa, aba\}$ is the language $h(L) = \{abab, abbbcab\}$.

If we have a regular expression r for a language L, then a regular expression for $h(L)$ can be obtained by simply applying the homomorphism to each Σ symbol of r.

Example 4.3

Take $\Sigma = \{a, b\}$ and $\Gamma = \{b, c, d\}$. Define h by

$$h(a) = dbcc,$$

$$h(b) = bdc.$$

If L is the regular language denoted by

$$r = (a + b^*)(aa)^*,$$

then

$$r_1 = (dbcc + (bdc)^*)(dbccdbcc)^*$$

denotes the regular language $h(L)$.

The general result on the closure of regular languages under any homomorphism follows from this example in an obvious manner.

If we have a regular expression r for a language L, then a regular expression for $h(L)$ can be obtained by simply applying the homomorphism to each Σ symbol of r.

Example 4.3

Take $\Sigma = \{a, b\}$ and $\Gamma = \{b, c, d\}$. Define h by

$$h(a) = dbcc,$$

$$h(b) = bdc.$$

If L is the regular language denoted by

$$r = (a + b^*)(aa)^*,$$

then

$$r_1 = (dbcc + (bdc)^*)\,(dbccdbcc)^*$$

denotes the regular language $h(L)$.

The general result on the closure of regular languages under any homomorphism follows from this example in an obvious manner.

If we have a regular expression r for a language L, then a regular expression for $h(L)$ can be obtained by simply applying the homomorphism to each Σ symbol of r.

Example 4.3

Take $\Sigma = \{a, b\}$ and $\Gamma = \{b, c, d\}$. Define h by

$$h(a) = dbcc,$$

$$h(b) = bdc.$$

If L is the regular language denoted by

$$r = (a + b^*)(aa)^*,$$

then

$$r_1 = (dbcc + (bdc)^*)\,(dbccdbcc)^*$$

denotes the regular language $h(L)$.

The general result on the closure of regular languages under any homomorphism follows from this example in an obvious manner.

If we have a regular expression r for a language L, then a regular expression for $h(L)$ can be obtained by simply applying the homomorphism to each Σ symbol of r.

Example 4.3

Take $\Sigma = \{a, b\}$ and $\Gamma = \{b, c, d\}$. Define h by

$$h(a) = dbcc,$$
$$h(b) = bdc.$$

If L is the regular language denoted by

$$r = (a + b^*)(aa)^*,$$

then

$$r_1 = (dbcc + (bdc)^*)\,(dbccdbcc)^*$$

denotes the regular language $h(L)$.

The general result on the closure of regular languages under any homomorphism follows from this example in an obvious manner.

If we have a regular expression r for a language L, then a regular expression for $h(L)$ can be obtained by simply applying the homomorphism to each Σ symbol of r.

Example 4.3

Take $\Sigma = \{a, b\}$ and $\Gamma = \{b, c, d\}$. Define h by

$$h(a) = dbcc,$$

$$h(b) = bdc.$$

If L is the regular language denoted by

$$r = (a + b^*)(aa)^*,$$

then

$$r_1 = (dbcc + (bdc)^*)\,(dbccdbcc)^*$$

denotes the regular language $h(L)$.

The general result on the closure of regular languages under any homomorphism follows from this example in an obvious manner.

If we have a regular expression r for a language L, then a regular expression for $h(L)$ can be obtained by simply applying the homomorphism to each Σ symbol of r.

Example 4.3

Take $\Sigma = \{a, b\}$ and $\Gamma = \{b, c, d\}$. Define h by

$$h(a) = dbcc,$$
$$h(b) = bdc.$$

If L is the regular language denoted by

$$r = (a + b^*)(aa)^*,$$

then

$$r_1 = (dbcc + (bdc)^*)\,(dbccdbcc)^*$$

denotes the regular language $h(L)$.

The general result on the closure of regular languages under any homomorphism follows from this example in an obvious manner.

If we have a regular expression r for a language L, then a regular expression for $h(L)$ can be obtained by simply applying the homomorphism to each Σ symbol of r.

Example 4.3

Take $\Sigma = \{a, b\}$ and $\Gamma = \{b, c, d\}$. Define h by

$$h(a) = dbcc,$$

$$h(b) = bdc.$$

If L is the regular language denoted by

$$r = (a + b^*)(aa)^*,$$

then

$$r_1 = (dbcc + (bdc)^*)\,(dbccdbcc)^*$$

denotes the regular language $h(L)$.

The general result on the closure of regular languages under any homomorphism follows from this example in an obvious manner.

If we have a regular expression r for a language L, then a regular expression for $h(L)$ can be obtained by simply applying the homomorphism to each Σ symbol of r.

Example 4.3

Take $\Sigma = \{a, b\}$ and $\Gamma = \{b, c, d\}$. Define h by

$$h(a) = dbcc,$$
$$h(b) = bdc.$$

If L is the regular language denoted by

$$r = (a + b^*)(aa)^*,$$

then

$$r_1 = (dbcc + (bdc)^*)\,(dbccdbcc)^*$$

denotes the regular language $h(L)$.

The general result on the closure of regular languages under any homomorphism follows from this example in an obvious manner.

If we have a regular expression r for a language L, then a regular expression for $h(L)$ can be obtained by simply applying the homomorphism to each Σ symbol of r.

Example 4.3

Take $\Sigma = \{a, b\}$ and $\Gamma = \{b, c, d\}$. Define h by

$$h(a) = dbcc,$$
$$h(b) = bdc.$$

If L is the regular language denoted by

$$r = (a + b^*)(aa)^*,$$

then

$$r_1 = (dbcc + (bdc)^*)\,(dbccdbcc)^*$$

denotes the regular language $h(L)$.

The general result on the closure of regular languages under any homomorphism follows from this example in an obvious manner.

If we have a regular expression r for a language L, then a regular expression for $h(L)$ can be obtained by simply applying the homomorphism to each Σ symbol of r.

Example 4.3

Take $\Sigma = \{a, b\}$ and $\Gamma = \{b, c, d\}$. Define h by

$$h(a) = dbcc,$$

$$h(b) = bdc.$$

If L is the regular language denoted by

$$r = (a + b^*)(aa)^*,$$

then

$$r_1 = (dbcc + (bdc)^*)\,(dbccdbcc)^*$$

denotes the regular language $h(L)$.

The general result on the closure of regular languages under any homomorphism follows from this example in an obvious manner.

If we have a regular expression r for a language L, then a regular expression for $h(L)$ can be obtained by simply applying the homomorphism to each Σ symbol of r.

Example 4.3

Take $\Sigma = \{a, b\}$ and $\Gamma = \{b, c, d\}$. Define h by

$$h(a) = dbcc,$$
$$h(b) = bdc.$$

If L is the regular language denoted by

$$r = (a + b^*)(aa)^*,$$

then

$$r_1 = (dbcc + (bdc)^*)\,(dbccdbcc)^*$$

denotes the regular language $h(L)$.

The general result on the closure of regular languages under any homomorphism follows from this example in an obvious manner.

If we have a regular expression r for a language L, then a regular expression for $h(L)$ can be obtained by simply applying the homomorphism to each Σ symbol of r.

Example 4.3

Take $\Sigma = \{a, b\}$ and $\Gamma = \{b, c, d\}$. Define h by

$$h(a) = dbcc,$$
$$h(b) = bdc.$$

If L is the regular language denoted by

$$r = (a + b^*)(aa)^*,$$

then

$$r_1 = (dbcc + (bdc)^*)\,(dbccdbcc)^*$$

denotes the regular language $h(L)$.

The general result on the closure of regular languages under any homomorphism follows from this example in an obvious manner.

If we have a regular expression r for a language L, then a regular expression for $h(L)$ can be obtained by simply applying the homomorphism to each Σ symbol of r.

Example 4.3

Take $\Sigma = \{a, b\}$ and $\Gamma = \{b, c, d\}$. Define h by

$$h(a) = dbcc,$$

$$h(b) = bdc.$$

If L is the regular language denoted by

$$r = (a + b^*)(aa)^*,$$

then

$$r_1 = (dbcc + (bdc)^*)\,(dbccdbcc)^*$$

denotes the regular language $h(L)$.

The general result on the closure of regular languages under any homomorphism follows from this example in an obvious manner.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ∎

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x : xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ∎

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x : xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ■

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x : xy \in L_1 \text{ for some } y \in L_2\}. \quad (1)$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ∎

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x : xy \in L_1 \text{ for some } y \in L_2\}. \quad (1)$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ∎

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x : xy \in L_1 \text{ for some } y \in L_2\}. \quad (1)$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ∎

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x : xy \in L_1 \text{ for some } y \in L_2\}. \quad (1)$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ∎

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x : xy \in L_1 \text{ for some } y \in L_2\}. \quad (1)$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ∎

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x : xy \in L_1 \text{ for some } y \in L_2\}. \quad (1)$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ∎

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x : xy \in L_1 \text{ for some } y \in L_2\}. \quad (1)$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ■

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x : xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ■

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x : xy \in L_1 \text{ for some } y \in L_2\}. \quad (1)$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ∎

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x : xy \in L_1 \text{ for some } y \in L_2\}. \quad (1)$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

4.1 Closure Properties of Regular Languages

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ■

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x : xy \in L_1 \text{ for some } y \in L_2\}. \quad (1)$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ∎

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x : xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ∎

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x\colon xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ■

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x\colon xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ■

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x\colon xy \in L_1 \text{ for some } y \in L_2\}. \quad (1)$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ■

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x \colon xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ∎

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x \colon xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ■

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x \colon xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Theorem 4.3

Let h be a homomorphism. If L is a regular language, then its homomorphic image $h(L)$ is also regular. The family of regular languages is therefore closed under arbitrary homomorphisms.

Proof. Let L be a regular language denoted by some regular expression r. We find $h(r)$ by substituting $h(a)$ for each symbol $a \in \Sigma$ of r. It can be shown directly by an appeal to the definition of a regular expression that the result is a regular expression. It is equally easy to see that the resulting expression denotes $h(L)$. All we need to do is to show that for every $w \in L(r)$, the corresponding $h(w)$ is in $L(h(r))$ and conversely that for every v in $L(h(r))$ there is a word w in L, such that $v = h(w)$. Leaving the details as an exercise, we claim that $h(L)$ is regular. ∎

Definition 4.2

Let L_1 and L_2 be languages on the same alphabet. Then the right quotient of L_1 with L_2 is defined as

$$L_1/L_2 = \{x \colon xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

To form the right quotient of L_1 with L_2, we take all the strings in L_1 that have a suffix belonging to L_2. Every such string, after removal of this suffix, belongs to L_1/L_2.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If
$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$
and
$$L_2 = \{b^m : m \geqslant 1\},$$
then
$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If
$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$
and
$$L_2 = \{b^m : m \geqslant 1\},$$
then
$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

Example 4.4

If

$$L_1 = \{a^n b^m : n \geqslant 1, m \geqslant 0\} \cup \{ba\}$$

and

$$L_2 = \{b^m : m \geqslant 1\},$$

then

$$L_1/L_2 = \{a^n b^m : n \geqslant 1, m \geqslant 0\}.$$

The strings in L_2 consist of one or more b's. Therefore, we arrive at the answer by removing one or more b's from those strings in L_1 that terminate with at least one b.

Note that here L_1, L_2, and L_1/L_2 are all regular. This suggests that the right quotient of any two regular languages is also regular. We shall prove this in the next theorem by a construction that takes the dfa's for L_1 and L_2 and constructs from them a dfa for L_1/L_2. Before we describe the construction in full, let us see how it applies to this example. We start with a dfa for L_1; say the automaton $M_1 = (Q, \Sigma, \delta, q_0, F)$ in the Figure.

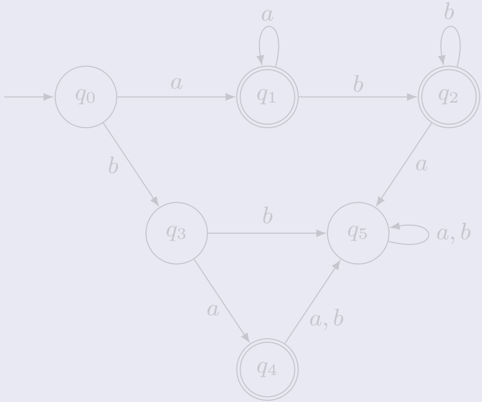
Example 4.4 (continuation)

Since an automaton for L_1/L_2 must accept any prefix of strings in L_1, we shall try to modify M_1 so that it accepts x if there is any y satisfying (1).

$$L_1/L_2 = \{x\colon xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

The difficulty comes in finding whether there is some y such that $xy \in L_1$ and $y \in L_2$. To solve it, we determine, for each $q \in Q$, whether there is a walk to a final state labeled v such that $v \in L_2$. If this is so, any x such that $\delta(q_0, x) = q$ will be in L_1/L_2. We modify the automaton accordingly to make q a final state.

Example 4.4 (continuation)

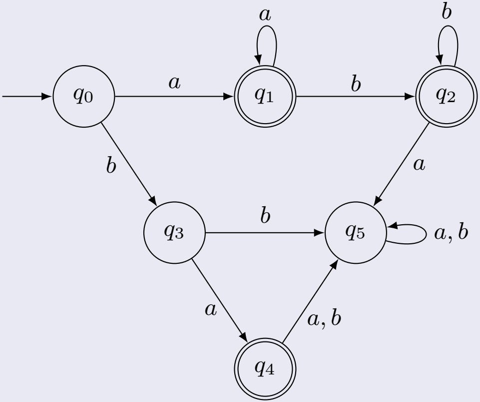


Since an automaton for L_1/L_2 must accept any prefix of strings in L_1, we shall try to modify M_1 so that it accepts x if there is any y satisfying (1).

$$L_1/L_2 = \{x\colon xy \in L_1 \text{ for some } y \in L_2\}. \quad (1)$$

The difficulty comes in finding whether there is some y such that $xy \in L_1$ and $y \in L_2$. To solve it, we determine, for each $q \in Q$, whether there is a walk to a final state labeled v such that $v \in L_2$. If this is so, any x such that $\delta(q_0, x) = q$ will be in L_1/L_2. We modify the automaton accordingly to make q a final state.

Example 4.4 (continuation)

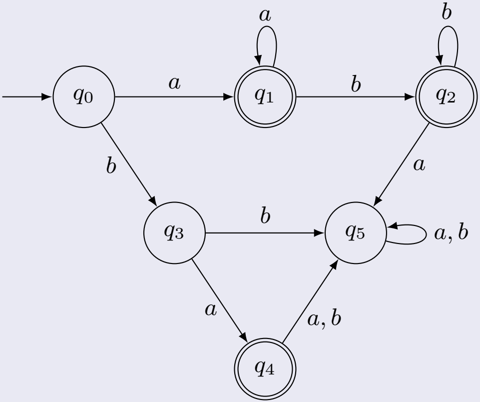


Since an automaton for L_1/L_2 must accept any prefix of strings in L_1, we shall try to modify M_1 so that it accepts x if there is any y satisfying (1).

$$L_1/L_2 = \{x \colon xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

The difficulty comes in finding whether there is some y such that $xy \in L_1$ and $y \in L_2$. To solve it, we determine, for each $q \in Q$, whether there is a walk to a final state labeled v such that $v \in L_2$. If this is so, any x such that $\delta(q_0, x) = q$ will be in L_1/L_2. We modify the automaton accordingly to make q a final state.

Example 4.4 (continuation)

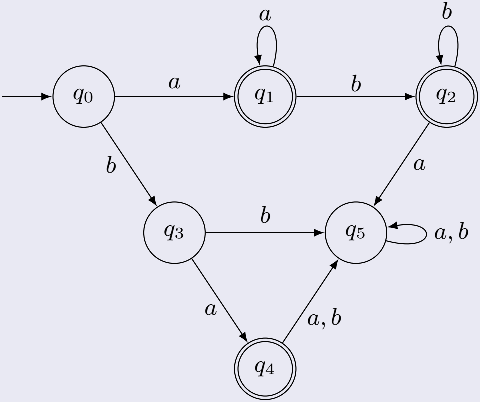


Since an automaton for L_1/L_2 must accept any prefix of strings in L_1, we shall try to modify M_1 so that it accepts x if there is any y satisfying (1).

$$L_1/L_2 = \{x\colon xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

The difficulty comes in finding whether there is some y such that $xy \in L_1$ and $y \in L_2$. To solve it, we determine, for each $q \in Q$, whether there is a walk to a final state labeled v such that $v \in L_2$. If this is so, any x such that $\delta(q_0, x) = q$ will be in L_1/L_2. We modify the automaton accordingly to make q a final state.

Example 4.4 (continuation)

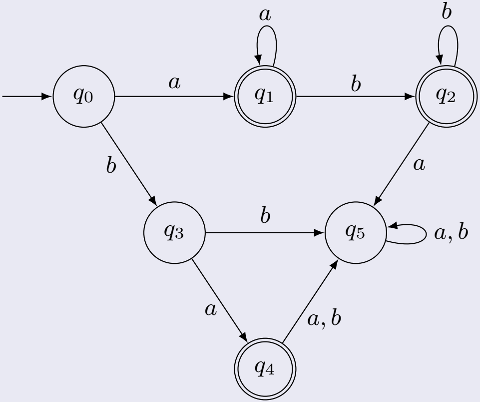


Since an automaton for L_1/L_2 must accept any prefix of strings in L_1, we shall try to modify M_1 so that it accepts x if there is any y satisfying (1).

$$L_1/L_2 = \{x \colon xy \in L_1 \text{ for some } y \in L_2\}. \quad (1)$$

The difficulty comes in finding whether there is some y such that $xy \in L_1$ and $y \in L_2$. To solve it, we determine, for each $q \in Q$, whether there is a walk to a final state labeled v such that $v \in L_2$. If this is so, any x such that $\delta(q_0, x) = q$ will be in L_1/L_2. We modify the automaton accordingly to make q a final state.

Example 4.4 (continuation)

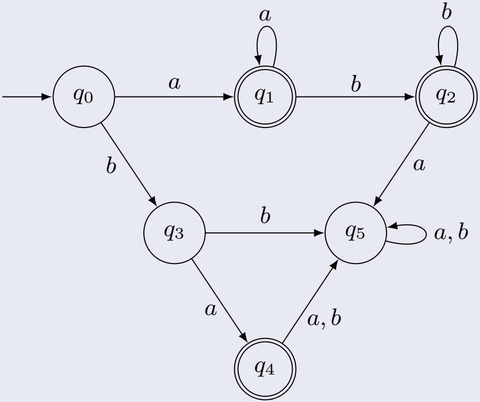


Since an automaton for L_1/L_2 must accept any prefix of strings in L_1, we shall try to modify M_1 so that it accepts x if there is any y satisfying (1).

$$L_1/L_2 = \{x \colon xy \in L_1 \text{ for some } y \in L_2\}. \quad (1)$$

The difficulty comes in finding whether there is some y such that $xy \in L_1$ and $y \in L_2$. To solve it, we determine, for each $q \in Q$, whether there is a walk to a final state labeled v such that $v \in L_2$. If this is so, any x such that $\delta(q_0, x) = q$ will be in L_1/L_2. We modify the automaton accordingly to make q a final state.

Example 4.4 (continuation)

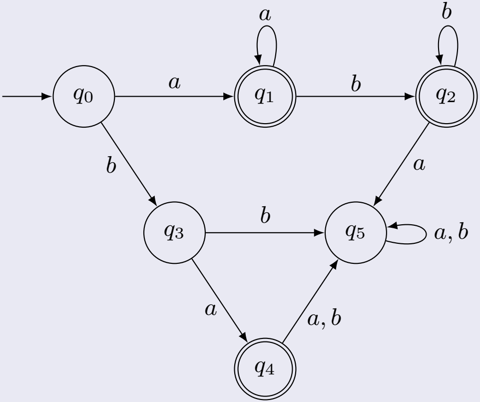


Since an automaton for L_1/L_2 must accept any prefix of strings in L_1, we shall try to modify M_1 so that it accepts x if there is any y satisfying (1).

$$L_1/L_2 = \{x\colon xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

The difficulty comes in finding whether there is some y such that $xy \in L_1$ and $y \in L_2$. To solve it, we determine, for each $q \in Q$, whether there is a walk to a final state labeled v such that $v \in L_2$. If this is so, any x such that $\delta(q_0, x) = q$ will be in L_1/L_2. We modify the automaton accordingly to make q a final state.

Example 4.4 (continuation)

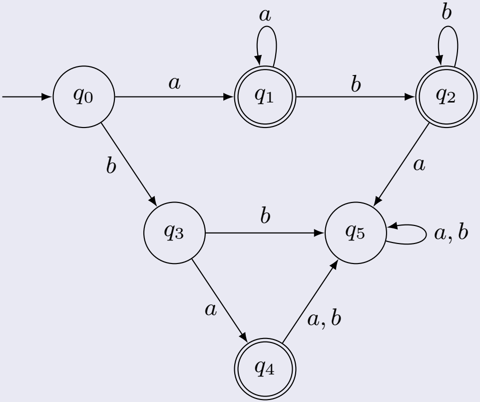


Since an automaton for L_1/L_2 must accept any prefix of strings in L_1, we shall try to modify M_1 so that it accepts x if there is any y satisfying (1).

$$L_1/L_2 = \{x \colon xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

The difficulty comes in finding whether there is some y such that $xy \in L_1$ and $y \in L_2$. To solve it, we determine, for each $q \in Q$, whether there is a walk to a final state labeled v such that $v \in L_2$. If this is so, any x such that $\delta(q_0, x) = q$ will be in L_1/L_2. We modify the automaton accordingly to make q a final state.

Example 4.4 (continuation)

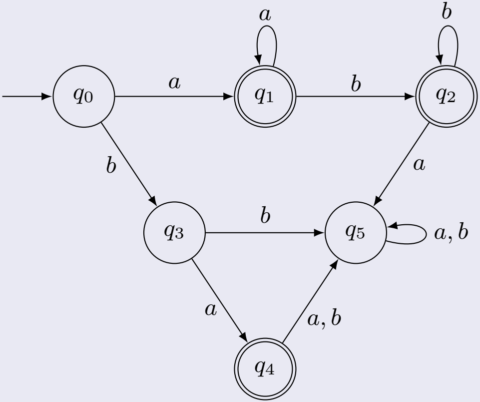


Since an automaton for L_1/L_2 must accept any prefix of strings in L_1, we shall try to modify M_1 so that it accepts x if there is any y satisfying (1).

$$L_1/L_2 = \{x\colon xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

The difficulty comes in finding whether there is some y such that $xy \in L_1$ and $y \in L_2$. To solve it, we determine, for each $q \in Q$, whether there is a walk to a final state labeled v such that $v \in L_2$. If this is so, any x such that $\delta(q_0, x) = q$ will be in L_1/L_2. We modify the automaton accordingly to make q a final state.

Example 4.4 (continuation)



Since an automaton for L_1/L_2 must accept any prefix of strings in L_1, we shall try to modify M_1 so that it accepts x if there is any y satisfying (1).

$$L_1/L_2 = \{x\colon xy \in L_1 \text{ for some } y \in L_2\}. \tag{1}$$

The difficulty comes in finding whether there is some y such that $xy \in L_1$ and $y \in L_2$. To solve it, we determine, for each $q \in Q$, whether there is a walk to a final state labeled v such that $v \in L_2$. If this is so, any x such that $\delta(q_0, x) = q$ will be in L_1/L_2. We modify the automaton accordingly to make q a final state.

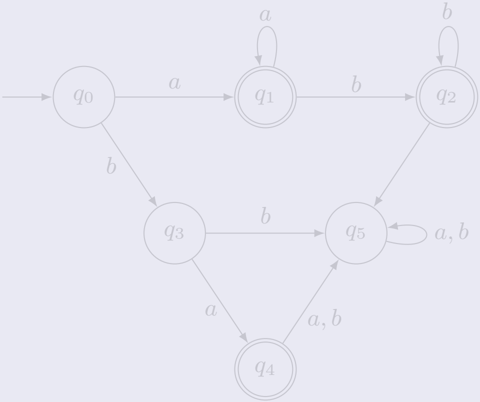
Example 4.4 (continuation)

To apply this to our present case, we check each state q_0, q_1, q_2, q_3, q_4, q_5 to see whether there is a walk labeled bb^* to any of the q_1, q_2, or q_4. We see that only q_1 and q_2 qualify; q_0, q_3, q_4 do not. The resulting automaton for L_1/L_2 is shown in the following Figure.

Check it to see that the construction works. The idea is generalized in the next theorem.

Example 4.4 (continuation)

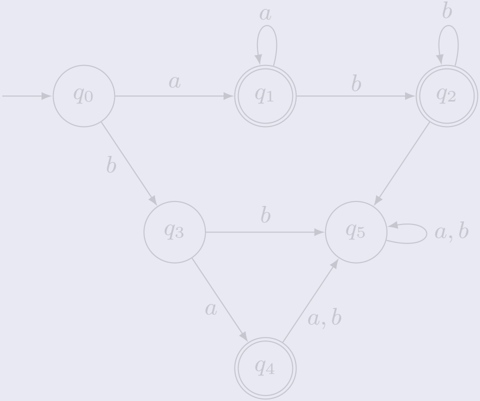
To apply this to our present case, we check each state q_0, q_1, q_2, q_3, q_4, q_5 to see whether there is a walk labeled bb^* to any of the q_1, q_2, or q_4. We see that only q_1 and q_2 qualify; q_0, q_3, q_4 do not. The resulting automaton for L_1/L_2 is shown in the following Figure.



Check it to see that the construction works. The idea is generalized in the next theorem.

Example 4.4 (continuation)

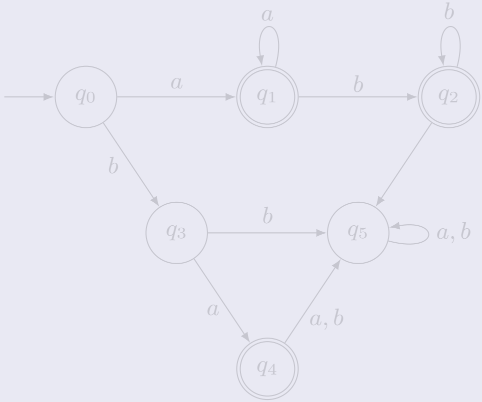
To apply this to our present case, we check each state q_0, q_1, q_2, q_3, q_4, q_5 to see whether there is a walk labeled bb^* to any of the q_1, q_2, or q_4. We see that only q_1 and q_2 qualify; q_0, q_3, q_4 do not. The resulting automaton for L_1/L_2 is shown in the following Figure.



Check it to see that the construction works. The idea is generalized in the next theorem.

Example 4.4 (continuation)

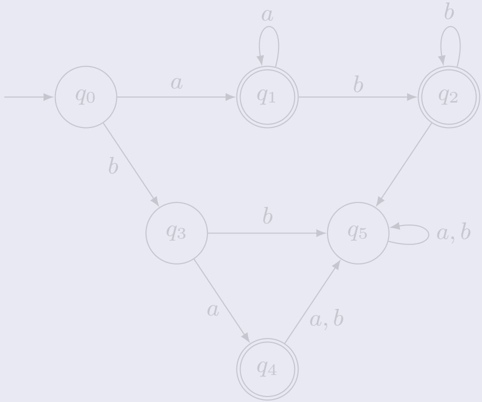
To apply this to our present case, we check each state q_0, q_1, q_2, q_3, q_4, q_5 to see whether there is a walk labeled bb^* to any of the q_1, q_2, or q_4. We see that only q_1 and q_2 qualify; q_0, q_3, q_4 do not. The resulting automaton for L_1/L_2 is shown in the following Figure.



Check it to see that the construction works. The idea is generalized in the next theorem.

Example 4.4 (continuation)

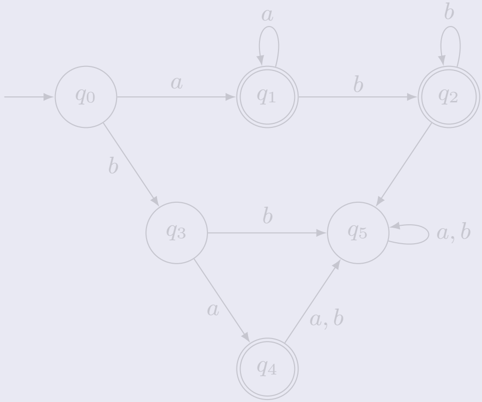
To apply this to our present case, we check each state q_0, q_1, q_2, q_3, q_4, q_5 to see whether there is a walk labeled bb^* to any of the q_1, q_2, or q_4. We see that only q_1 and q_2 qualify; q_0, q_3, q_4 do not. The resulting automaton for L_1/L_2 is shown in the following Figure.



Check it to see that the construction works. The idea is generalized in the next theorem.

Example 4.4 (continuation)

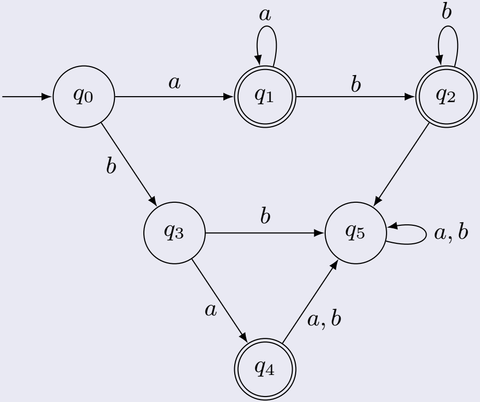
To apply this to our present case, we check each state q_0, q_1, q_2, q_3, q_4, q_5 to see whether there is a walk labeled bb^* to any of the q_1, q_2, or q_4. We see that only q_1 and q_2 qualify; q_0, q_3, q_4 do not. The resulting automaton for L_1/L_2 is shown in the following Figure.



Check it to see that the construction works. The idea is generalized in the next theorem.

Example 4.4 (continuation)

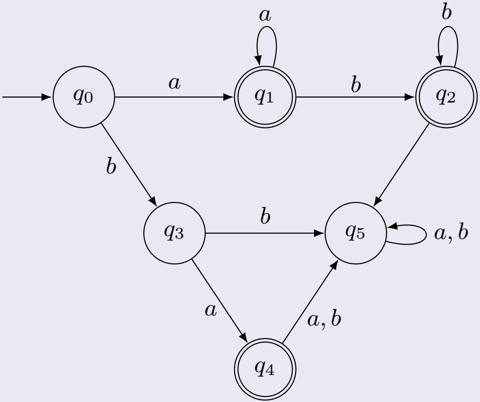
To apply this to our present case, we check each state q_0, q_1, q_2, q_3, q_4, q_5 to see whether there is a walk labeled bb^* to any of the q_1, q_2, or q_4. We see that only q_1 and q_2 qualify; q_0, q_3, q_4 do not. The resulting automaton for L_1/L_2 is shown in the following Figure.



Check it to see that the construction works. The idea is generalized in the next theorem.

Example 4.4 (continuation)

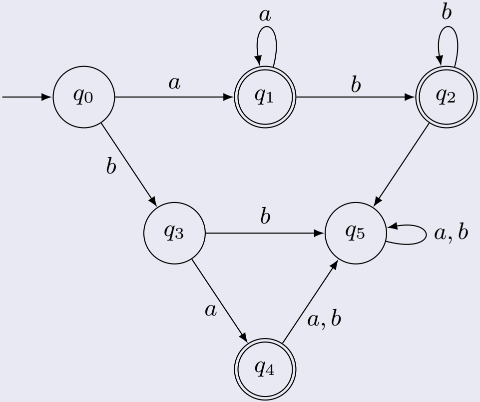
To apply this to our present case, we check each state q_0, q_1, q_2, q_3, q_4, q_5 to see whether there is a walk labeled bb^* to any of the q_1, q_2, or q_4. We see that only q_1 and q_2 qualify; q_0, q_3, q_4 do not. The resulting automaton for L_1/L_2 is shown in the following Figure.



Check it to see that the construction works. The idea is generalized in the next theorem.

Example 4.4 (continuation)

To apply this to our present case, we check each state q_0, q_1, q_2, q_3, q_4, q_5 to see whether there is a walk labeled bb^* to any of the q_1, q_2, or q_4. We see that only q_1 and q_2 qualify; q_0, q_3, q_4 do not. The resulting automaton for L_1/L_2 is shown in the following Figure.



Check it to see that the construction works. The idea is generalized in the next theorem.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

Theorem 4.4

If L_1 and L_2 are regular languages, then L_1/L_2 is also regular. We say that the family of regular languages is closed under right quotient with a regular language.

Proof. Let $L_1 = L(M)$, where $M = (Q, \Sigma, \delta, q_0, F)$ is a dfa. We construct another dfa $\widehat{M} = (Q, \Sigma, \delta, q_0, \widehat{F})$ as follows. For each $q_i \in Q$, determine if there exists a word $y \in L_2$ such that

$$\delta^*(q_i, y) = q_f \in F.$$

This can be done by looking at dfa's $M_i = (Q, \Sigma, \delta, q_i, F)$. The automaton M_i is M with the initial state q_0 replaced by q_i. We now determine whether there exists a word $y \in L(M_i)$ that is also in L_2. For this, we can use the construction for the intersection of two regular languages given in Theorem 4.1, finding the transition graph for $L_2 \cap L(M_i)$. If there is any path between its initial vertex and any final vertex, then $L_2 \cap L(M_i)$ is not empty. In that case, add q_i to $\widehat{F}$. Repeating this for every $q_i \in Q$, we determine $\widehat{F}$ and thereby construct $\widehat{M}$.

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ■

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ■

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ∎

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ∎

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ■

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ■

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ■

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ■

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ∎

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ∎

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ■

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ■

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ∎

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ∎

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ∎

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ■

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ■

To prove that $L(\widehat{M}) = L_1/L_2$, let x be any element of L_1/L_2. Then there must be a word $y \in L_2$ such that $xy \in L_1$. This implies that

$$\delta^*(q_0, xy) \in F,$$

so that there must be some $q \in Q$ such that

$$\delta^*(q_0, x) = q$$

and

$$\delta^*(q, y) \in F.$$

Therefore, by construction, $q \in \widehat{F}$, and $\widehat{M}$ accepts x because $\delta^*(q_0, x)$ is in $\widehat{F}$.

Conversely, for any x accepted by $\widehat{M}$, we have

$$\delta^*(q_0, x) = q \in \widehat{F}.$$

But again by construction, this implies that there exists a word $y \in L_2$ such that $\delta^*(q, y) \in F$. Therefore, xy is in L_1, and x is in L_1/L_2. We therefore conclude that

$$L(\widehat{M}) = L_1/L_2,$$

and from this that L_1/L_2 is regular. ■

Example 4.5

Find L_1/L_2 for

$$L_1 = L(a^*baa^*),$$
$$L_2 = L(ab^*).$$

We first find a dfa that accepts L_1. This is easy, and a solution is given in the following Figure.

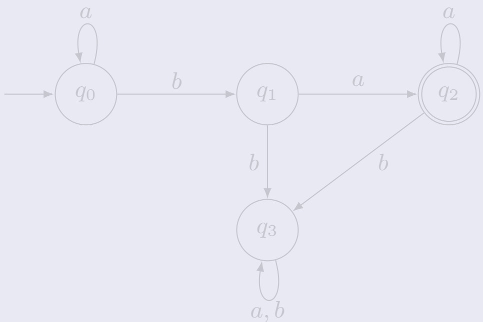
The example is simple enough so that we can skip the formalities of the construction.

Example 4.5

Find L_1/L_2 for

$$L_1 = L(a^*baa^*),$$
$$L_2 = L(ab^*).$$

We first find a dfa that accepts L_1. This is easy, and a solution is given in the following Figure.



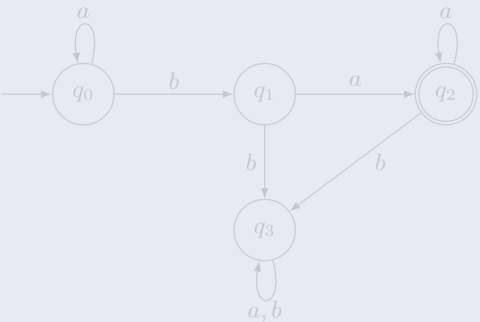
The example is simple enough so that we can skip the formalities of the construction.

Example 4.5

Find L_1/L_2 for

$$L_1 = L(a^*baa^*),$$
$$L_2 = L(ab^*).$$

We first find a dfa that accepts L_1. This is easy, and a solution is given in the following Figure.



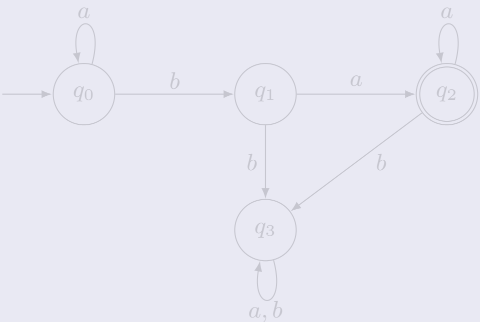
The example is simple enough so that we can skip the formalities of the construction.

Example 4.5

Find L_1/L_2 for

$$L_1 = L(a^*baa^*),$$
$$L_2 = L(ab^*).$$

We first find a dfa that accepts L_1. This is easy, and a solution is given in the following Figure.



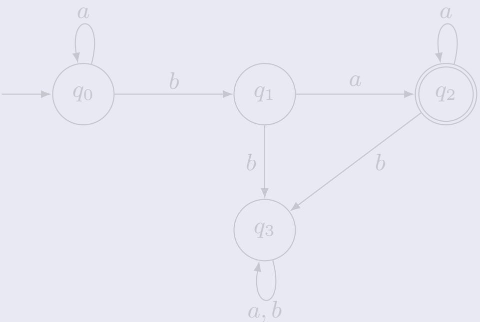
The example is simple enough so that we can skip the formalities of the construction.

Example 4.5

Find L_1/L_2 for

$$L_1 = L(a^*baa^*),$$
$$L_2 = L(ab^*).$$

We first find a dfa that accepts L_1. This is easy, and a solution is given in the following Figure.



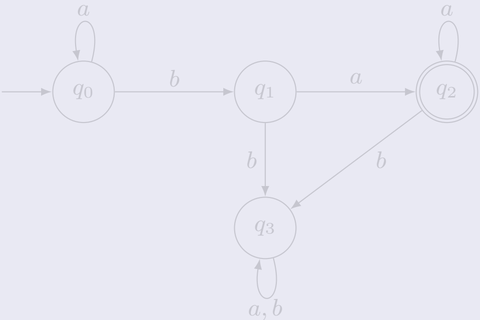
The example is simple enough so that we can skip the formalities of the construction.

Example 4.5

Find L_1/L_2 for

$$L_1 = L(a^*baa^*),$$
$$L_2 = L(ab^*).$$

We first find a dfa that accepts L_1. This is easy, and a solution is given in the following Figure.



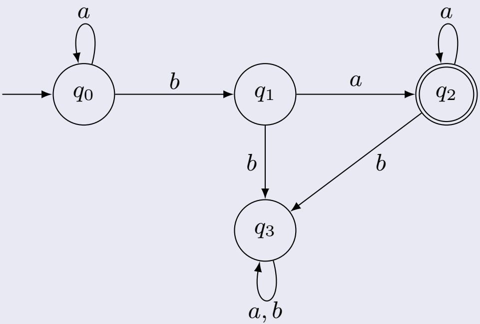
The example is simple enough so that we can skip the formalities of the construction.

Example 4.5

Find L_1/L_2 for

$$L_1 = L(a^*baa^*),$$
$$L_2 = L(ab^*).$$

We first find a dfa that accepts L_1. This is easy, and a solution is given in the following Figure.



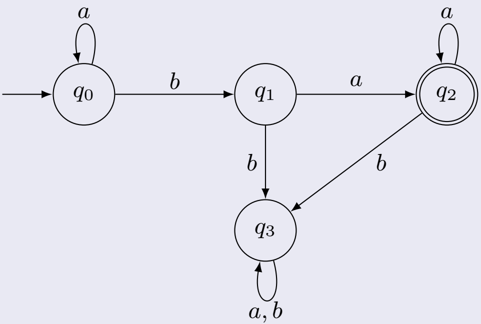
The example is simple enough so that we can skip the formalities of the construction.

Example 4.5

Find L_1/L_2 for

$$L_1 = L(a^*baa^*),$$
$$L_2 = L(ab^*).$$

We first find a dfa that accepts L_1. This is easy, and a solution is given in the following Figure.



The example is simple enough so that we can skip the formalities of the construction.

Example 4.5 (continuation)

From the graph in the previous Figure it is quite evident that

$$L(M_0) \cap L_2 = \varnothing,$$

$$L(M_1) \cap L_2 = \{a\} \neq \varnothing,$$

$$L(M_2) \cap L_2 = \{a\} \neq \varnothing,$$

$$L(M_3) \cap L_2 = \varnothing.$$

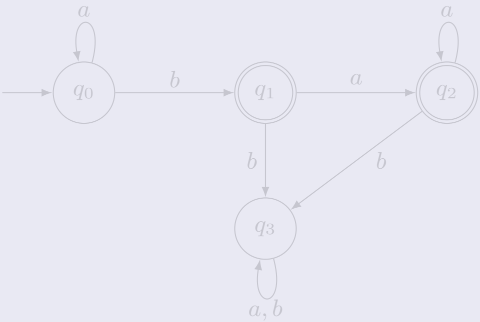
Therefore, the automaton accepting L_1/L_2 is determined. The result is shown in the Figure.

Example 4.5 (continuation)

From the graph in the previous Figure it is quite evident that

$$L(M_0) \cap L_2 = \varnothing,$$
$$L(M_1) \cap L_2 = \{a\} \neq \varnothing,$$
$$L(M_2) \cap L_2 = \{a\} \neq \varnothing,$$
$$L(M_3) \cap L_2 = \varnothing.$$

Therefore, the automaton accepting L_1/L_2 is determined. The result is shown in the Figure.

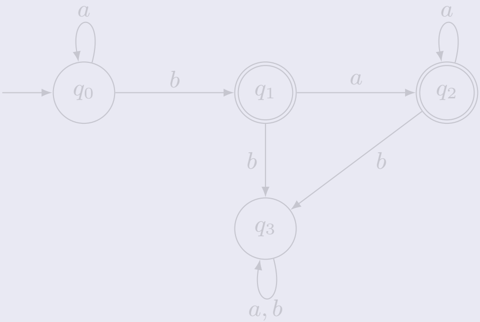


Example 4.5 (continuation)

From the graph in the previous Figure it is quite evident that

$$L(M_0) \cap L_2 = \varnothing,$$
$$L(M_1) \cap L_2 = \{a\} \neq \varnothing,$$
$$L(M_2) \cap L_2 = \{a\} \neq \varnothing,$$
$$L(M_3) \cap L_2 = \varnothing.$$

Therefore, the automaton accepting L_1/L_2 is determined. The result is shown in the Figure.

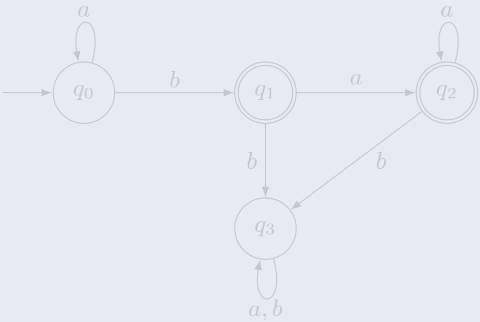


Example 4.5 (continuation)

From the graph in the previous Figure it is quite evident that

$$L(M_0) \cap L_2 = \varnothing,$$
$$L(M_1) \cap L_2 = \{a\} \neq \varnothing,$$
$$L(M_2) \cap L_2 = \{a\} \neq \varnothing,$$
$$L(M_3) \cap L_2 = \varnothing.$$

Therefore, the automaton accepting L_1/L_2 is determined. The result is shown in the Figure.

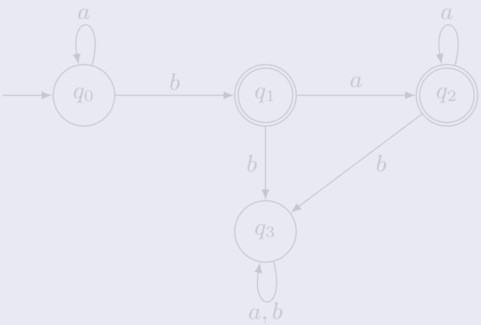


Example 4.5 (continuation)

From the graph in the previous Figure it is quite evident that

$$L(M_0) \cap L_2 = \varnothing,$$
$$L(M_1) \cap L_2 = \{a\} \neq \varnothing,$$
$$L(M_2) \cap L_2 = \{a\} \neq \varnothing,$$
$$L(M_3) \cap L_2 = \varnothing.$$

Therefore, the automaton accepting L_1/L_2 is determined. The result is shown in the Figure.

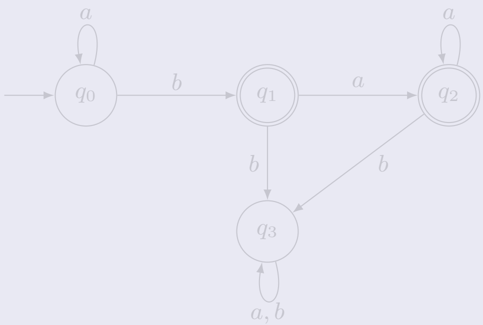


Example 4.5 (continuation)

From the graph in the previous Figure it is quite evident that

$$L(M_0) \cap L_2 = \varnothing,$$
$$L(M_1) \cap L_2 = \{a\} \neq \varnothing,$$
$$L(M_2) \cap L_2 = \{a\} \neq \varnothing,$$
$$L(M_3) \cap L_2 = \varnothing.$$

Therefore, the automaton accepting L_1/L_2 is determined. The result is shown in the Figure.

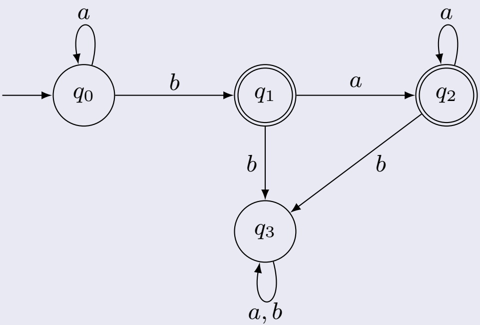


Example 4.5 (continuation)

From the graph in the previous Figure it is quite evident that

$$L(M_0) \cap L_2 = \varnothing,$$
$$L(M_1) \cap L_2 = \{a\} \neq \varnothing,$$
$$L(M_2) \cap L_2 = \{a\} \neq \varnothing,$$
$$L(M_3) \cap L_2 = \varnothing.$$

Therefore, the automaton accepting L_1/L_2 is determined. The result is shown in the Figure.



Example 4.5 (continuation)

It accepts the language denoted by the regular expression of $a^*b + a^*baa^*$, which can be simplified to a^*ba^*. Thus $L_1/L_2 = L(a^*ba^*)$.

Example 4.5 (continuation)

It accepts the language denoted by the regular expression of $a^*b + a^*baa^*$, which can be simplified to a^*ba^*. Thus $L_1/L_2 = L(a^*ba^*)$.

Example 4.5 (continuation)

It accepts the language denoted by the regular expression of $a^*b + a^*baa^*$, which can be simplified to a^*ba^*. Thus $L_1/L_2 = L(a^*ba^*)$.

Example 4.5 (continuation)

It accepts the language denoted by the regular expression of $a^*b + a^*baa^*$, which can be simplified to a^*ba^*. Thus $L_1/L_2 = L(a^*ba^*)$.

Thank You for attention!