University of Houston

Notes

COSC 3340
Intro. to Automata and Computability

Ernst Leiss

Khalid Hourani

Contents

1 Formal Languages

1.1 Introduction

Definition 1.1.1. An **Alphabet** is a finite, non-empty set of atomic symbols.

Definition 1.1.2. A **word** or **string** is any finite sequence of symbols from an alphabet.

Definition 1.1.3. The **length** of a string, s, denoted $|s|$, is the number of symbols in s.

Definition 1.1.4. Given strings $s = s_1 s_2 \dots s_n$ and $t = t_1 t_2 \dots t_m$, their **concatenation** is defined

$$s \cdot t = s_1 s_2 \cdots s_n t_1 t_2 \cdots t_m$$

We denote by ε the **empty string**, the unique string of 0 characters.

Definition 1.1.5. Let A be any alphabet. The **Kleene Closure** of A, denoted A^*, is the set of all strings of any length over A.

Theorem 1.1.1. Let A be any finite set. Then A^* is countably infinite.

Proof. That A^* is infinite is straightforward: since A is non-empty, take $a \in A$. Then $\{a, aa, aaa, \dots\} \subseteq A^*$.

To see that it is countable, we first write $|A| = n$. Now, consider the set of all strings of length 0. This is simply $\{\varepsilon\}$. Moreover, there are n strings of length 1, n^2 strings of length 2, n^3 strings of length 3, and so on. Thus, we map ε to 0, the strings of length 1 to $1, 2, \dots, n$, the strings of length 2 to $n+1, n+2, \dots, n+n^2$, the strings of length 3 to $n+n^2+1, n+n^2+2, \dots, n+n^2+n^3$, and so on. This is a bijection from A^* to $\mathbb{N}$, which completes the proof. □

Definition 1.1.6. Given an alphabet A, a **formal language** or simply **language** L is any subset of A^*.

Theorem 1.1.2. Given an alphabet A, the set of languages over A is uncountable.

Proof. Suppose, by way of contradiction, that the set of languages were countable, i.e., that we can enumerate the set as $\{L_1, L_2, L_3, \dots\}$. Consider the set of all strings $\{s_1, s_2, s_3, \dots\}$. Let L be the language defined as follows:

$$s_i \in L \text{ if and only if } s_i \notin L_i$$

To see that L is not in the above list, consider s_i. If s_i is in L, then s_i is not in L_i, by construction, and $L \neq L_i$. Similarly, if s_i is not in L, then s_i must be in L_i, by construction, and $L \neq L_i$. In other words, for all i, $L \neq L_i$. Then L is not in the above list, which is a contradiction. Hence, the set of languages is uncountable. □

All set operations, such as union, intersection, complement, set-difference, etc. can be applied to languages, since languages are simply subsets of a Kleene Closure of an alphabet.

Definition 1.1.7. Given two languages L_1 and L_2, the concatenation $L_1 \cdot L_2$ is given by

$$L_1 \cdot L_2 = \{s \cdot t | s \in L_1 \text{ and } t \in L_2\}$$

Clearly, we have

$$L \cdot \emptyset = \emptyset = \emptyset \cdot L$$
$$L \cdot \{\varepsilon\} = L = \{\varepsilon\} \cdot L$$

Note that $L_1 \cdot L_2$ is not the same as $L_1 \times L_2$. Let $L_1 = L_2 = \{\varepsilon, 0, 00\}$. Then

$$L_1 \times L_2 = \{(\varepsilon, \varepsilon), (\varepsilon, 0), (\varepsilon, 00), (0, \varepsilon), (0, 0), (0, 00), (00, \varepsilon), (00, 0), (00, 00)\}$$

whereas

$$L_1 \cdot L_2 = \{\varepsilon, 0, 00, 000, 0000\}$$

Definition 1.1.8. Given a language L, the **Kleene Closure** of L, L^*, is

$$L^* = \bigcup_{i=0}^{\infty} L^i$$

where

$$L^i = \begin{cases} \{\varepsilon\} & \text{if } i = 0 \\ L \cdot L^{i-1} & \text{otherwise} \end{cases}$$

Note that, while 0^0 is normally left undefined, we define $\emptyset^0 = \{\varepsilon\}$.

Theorem 1.1.3. L^* is finite if and only if $L = \emptyset$ or $L = \{\varepsilon\}$.

Proof. If $L = \emptyset$, then $L^i = \emptyset^i = \emptyset$ for $i > 0$. Then

$$\begin{aligned} \emptyset^* &= \bigcup_{i=0}^{\infty} \emptyset^i \\ &= \emptyset^0 \cup \bigcup_{i=1}^{\infty} \emptyset^i \\ &= \{\varepsilon\} \cup \bigcup_{i=1}^{\infty} \emptyset \\ &= \{\varepsilon\} \end{aligned}$$

Similarly, if $L = \{\varepsilon\}$, then $L^i = \{\varepsilon\}$ for all i, and

$$\begin{aligned} \{\varepsilon\}^* &= \bigcup_{i=0}^{\infty} \{\varepsilon\}^i \\ &= \bigcup_{i=1}^{\infty} \{\varepsilon\} \\ &= \{\varepsilon\} \end{aligned}$$

However, if L is neither $\emptyset$ nor $\{\varepsilon\}$, then there exists a string $s \in L$ with length at least 1. Then $s, ss, sss, \ldots$, are in L^*, hence L^* is infinite. □

1.2 Regular Languages

1.2.1 Finite Automata

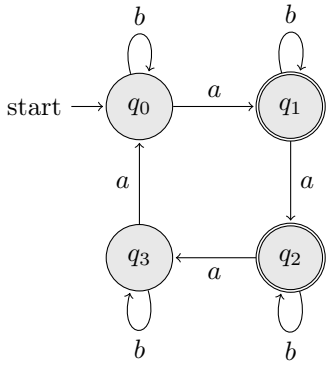
Definition 1.2.1. A **Deterministic Finite-State Automata** (DFA) or **Finite-State Machine** is a quintuple $(A, Q, \tau, q_0, \mathcal{F})$ where

A is the **alphabet**
Q is a finite, non-empty **set of states**
$\tau : Q \times A \to Q$ is the **transition function**
q_0 is the **initial state**
$\mathcal{F} \subseteq Q$ is the set of **final states**

We can extend τ as follows:
$\tau^* : Q \times A^* \to Q$

$$\tau^*(q, s) = \begin{cases} q & \text{if } s = \varepsilon \\ \tau^*(\tau(q, s_0), s') & \text{if } s = s_0 \cdot s' \end{cases}$$

We proceed informally and use τ to refer to τ^*.
Consider the following DFA:



The figure indicates that we begin at state q_0. The double-circles for states q_1 and q_2 indicate that they are accepting or final states. An arrow indicates the state to move to after receiving an input. For example, if we receive the input string *abba*, we begin at state q_0 and receive a, so we move to state q_1. We then receive b and stay in q_1. We repeat this for the next symbol, b, and then move to q_2 upon receiving the final a. Since q_2 is a final state, we say that this DFA **accepts** the string *abba*.

We can represent the above DFA using a table, as follows:

	a	b	
$\rightarrow q_0$	q_1	q_0	0
q_1	q_2	q_1	1
q_2	q_3	q_2	1
q_3	q_0	q_3	0

The first column indicates the states, while the first row indicates the symbols. The final column indicates whether a state is accepting: 0 refers to a non-final state, 1 to a final state. The remaining values indicate the transition function τ, e.g. $\tau(q_0, a) = q_1$, indicated by the entry corresponding to row q_0 and column a. Finally, the arrow pointing to q_0 indicates that it is the starting position.

Definition 1.2.2. Let $\underset{\sim}{D}$ be some DFA. Then $L(\underset{\sim}{D})$, the language accepted by the DFA, is

$$\{s \in A^* | \tau(q_0, s) \in \mathcal{F}\}$$

Definition 1.2.3. A language is **regular** if and only if there exists a DFA that accepts it.

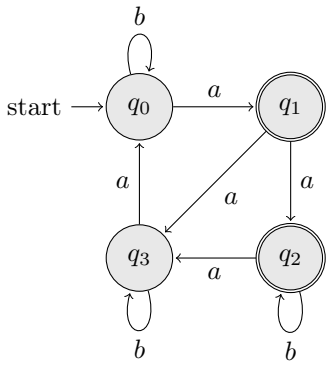
Definition 1.2.4. A **Non-Deterministic Finite-State Automata** (NFA) is a quintuple $(A, Q, \tau, q_0, \mathcal{F})$ where

A is the **alphabet**
Q is a finite, non-empty **set of states**
$\tau : Q \times A \rightarrow 2^Q$ is the **transition function**
q_0 is the **initial state**
$\mathcal{F} \subseteq Q$ is the set of **final states**

We can extend τ as follows:
$\tau^* : 2^Q \times A^* \rightarrow 2^Q$

$$\tau^*(P, s) = \begin{cases} P & \text{if } s = \varepsilon \\ \tau^*\left(\bigcup_{q \in P} \tau(q, s_0), s'\right) & \text{if } s = s_0 \cdot s' \end{cases}$$

We proceed informally and use τ to refer to τ^*.
Consider the following NFA:



The diagrams for an NFA and DFA follow the same notation. However, the notation for the table differs slightly:

	a	b	
$\rightarrow q_0$	q_1	q_0	0
q_1	q_2q_3	$\emptyset$	1
q_2	q_3	q_2	1
q_3	q_0	q_3	0

The values of the transition function are now sets. We informally refer to the set $\{q_0\}$ by q_0, and similarly the set $\{q_2, q_3\}$ by q_2q_3. In some cases, to avoid ambiguity, we will use commas, e.g. we may represent $\{q_2, q_3\}$ as q_2, q_3. We similarly say, given a string s, if there exists a path through an NFA that ends in a final state, we say that the NFA **accepts** s.

Similarly, we define the set of languages accepted by an NFA $\underset{\sim}{N}$, $L(\underset{\sim}{N})$, as

$$L(\underset{\sim}{N}) = \{s \in A^* | \tau(q_0, s) \cap \mathcal{F} \neq \emptyset\}$$

It should be clear that each DFA is an NFA, but the reverse is not true. However, we can convert an NFA to a DFA on the powerset $2^{\mathcal{Q}}$ by using the **subset construction**: begin with the initial state and traverse the NFA, adding unseen states to the left-most column until all paths have been exhausted. For example, with our NFA above, we begin with:

	a	b	
$\rightarrow q_0$	q_1	q_0	0

q_0 has already been seen, so we ignore it. q_1 is new, so we add it to the table:

	a	b
$\rightarrow q_0$	q_1	q_0
q_1		

We now visit the corresponding states of q_1, which are q_2q_3 and $\emptyset$, both of which have not yet been visited.

	a	b
$\rightarrow q_0$	q_1	q_0
q_1	q_2q_3	$\emptyset$
q_2q_3		
$\emptyset$		

When q_2 receives a, it transitions to state q_3. When q_3 receives a, it transitions to state q_0, so q_2q_3 transitions to q_0q_3. Similarly, q_2q_3 transitions to state q_2q_3 when it receives b.

	a	b
$\rightarrow q_0$	q_1	q_0
q_1	q_2q_3	$\emptyset$
q_2q_3	q_0q_3	q_2q_3
$\emptyset$		

The empty set transitions to the empty set, by definition.

	a	b
$\to q_0$	q_1	q_0
q_1	q_2q_3	$\emptyset$
q_2q_3	q_0q_3	q_2q_3
$\emptyset$	$\emptyset$	$\emptyset$

q_0q_3 has not yet been visited, so we add it to the left-most column:

	a	b
$\to q_0$	q_1	q_0
q_1	q_2q_3	$\emptyset$
q_2q_3	q_0q_3	q_2q_3
$\emptyset$	$\emptyset$	$\emptyset$
q_0q_3		

Then we visit its corresponding states:

	a	b
$\to q_0$	q_1	q_0
q_1	q_2q_3	$\emptyset$
q_2q_3	q_0q_3	q_2q_3
$\emptyset$	$\emptyset$	$\emptyset$
q_0q_3	q_0q_1	q_0q_3

Continuing, we end with the following DFA:

	a	b
$\to q_0$	q_1	q_0
q_1	q_2q_3	$\emptyset$
q_2q_3	q_0q_3	q_2q_3
$\emptyset$	$\emptyset$	$\emptyset$
q_0q_3	q_0q_1	q_0q_3
q_0q_1	$q_1q_2q_3$	q_0
$q_1q_2q_3$	$q_0q_2q_3$	q_2q_3
$q_0q_2q_3$	$q_0q_1q_3$	$q_0q_2q_3$
$q_0q_1q_3$	$q_0q_1q_2q_3$	q_0q_3
$q_0q_1q_2q_3$	$q_0q_1q_2q_3$	$q_0q_2q_3$

However, we need to include the accepting states. The accepting states of the NFA are q_1 and q_2, and thus any state including either state is accepting:

	a	b	
$\to q_0$	q_1	q_0	0
q_1	q_2q_3	$\emptyset$	1
q_2q_3	q_0q_3	q_2q_3	1
$\emptyset$	$\emptyset$	$\emptyset$	0
q_0q_3	q_0q_1	q_0q_3	0
q_0q_1	$q_1q_2q_3$	q_0	1
$q_1q_2q_3$	$q_0q_2q_3$	q_2q_3	1
$q_0q_2q_3$	$q_0q_1q_3$	$q_0q_2q_3$	1
$q_0q_1q_3$	$q_0q_1q_2q_3$	q_0q_3	1
$q_0q_1q_2q_3$	$q_0q_1q_2q_3$	$q_0q_2q_3$	1

Note that an NFA does not necessarily admit a DFA with as many states. Consider the following example:

	a	b	
$\to 0$	$\{1, 2, \ldots, n\}$	0	0
1	2	1	0
2	3	2	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$
i	$i+1$	i	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-1$	n	$n-1$	0
n	1	n	1

The NFA above admits the following DFA:

	a	b	
$\to 0$	$\{1, 2, \ldots, n\}$	0	0
$\{1, 2, \ldots, n\}$	$\{1, 2, \ldots, n\}$	$\{1, 2, \ldots, n\}$	1

The above DFA contains only 2 states, despite the NFA containing $n + 1$ states.

That every NFA admits a DFA which accepts the same language shows that the class of languages denoted by DFAs, $\mathcal{L}_{\text{DFA}}$, is the same as the class of languages denoted by NFAs, $\mathcal{L}_{\text{NFA}}$, i.e, that

$$\mathcal{L}_{\text{DFA}} = \mathcal{L}_{\text{NFA}}$$

For an NFA, there is no guarantee of a unique smallest NFA which accepts the same strings. However, for a DFA, such a notion exists.

Consider two states, p and q, and corresponding L_p and L_q, where L_p has initial state p and L_q has initial state q. We say that p and q are distinguishable if there exists a string s such that s is in L_p and not in L_q, or vice-versa. We use this notion to **reduce** a DFA.

Begin with a partition of Q into subsets $\mathcal{F}$ and $Q - \mathcal{F}$, i.e., the accepting and rejecting states. For a pair of states p, q if the result of transitioning p and q falls into different partitions, we partition the subset and continue.

For example, given the following DFA:

	a	b	
$\to 0$	1	2	0
1	2	3	1
2	3	4	0
3	0	5	1
4	5	6	0
5	6	7	1
6	7	0	0
7	4	1	1

We have two partitions:

Rejecting	Accepting
0, 2, 4, 6	1, 3, 5, 7

Now, 0 gets sent to the accepting partition by a and to the rejecting partition by b. Similarly, 2, 4, and 6 get sent to the accepting partition by a and to the rejecting partition by b. Thus, they belong to the same partition.

In the same vein, 1 gets sent to the rejecting partition by a and to the accepting partition by b. Similarly, 3, 5, and 7 get sent to the rejecting partition by a and to the accepting partition by b. Thus, our next partition is

Rejecting	Accepting
0, 2, 4, 6	1, 3, 5, 7
0, 2, 4, 6	1, 3, 5, 7

That our row is the same as the preceding one indicates that we have finished, and now have a minimal DFA. Call the first subset p and the second q. When an element in p receives a, it is sent to q. When it receives b, it is sent to p. Similar logic for q gives our new DFA:

	a	b	
$\rightarrow p$	q	p	0
q	p	q	1

Recall that p began as a subset of the rejecting elements and q the accepting elements, which informs the last column of the above table.

Not all DFAs can be reduced. An obvious example is the above reduced DFA. For a less trivial example, consider the following DFA:

	a	b	
$\rightarrow 0$	1	2	0
1	2	3	1
2	3	4	0
3	0	5	1
4	5	6	0
5	6	7	1
6	7	0	0
7	4	2	1

Begin, as in the previous problem, with two partitions:

Rejecting	Accepting
0, 2, 4, 6	1, 3, 5, 7

As in the previous problem, 0, 2, 4, and 6 get sent to the same partition under a and b, respectively. Under a, 1, 3, 5, and 7 go to the rejecting partition. However, under b, 7 goes to the rejecting partition while 1, 3, and 5 go to the accepting partition, which means we must create a new partition for 7.

Rejecting	Accepting	
0, 2, 4, 6	1, 3, 5, 7	
0, 2, 4, 6	1, 3, 5	7

We continue the process, noting that there is no need to consider singletones, i.e., the partition $\{7\}$ is already in its finale state. Under a, 0, 2, and 4 get sent to the $\{1, 3, 5\}$ partition. Under b, they get sent to the $\{0, 2, 4, 6\}$ partition. However, 6 gets sent to the $\{7\}$ partition, and so it must be partitioned separately. Similarly, 1 and 3 get sent to the $\{0, 2, 4, 6\}$ partition under a, and to the $\{1, 3, 5\}$ partition under b. 5, on the other hand, gets sent to the $\{7\}$ partition, and must be partitioned separately. In total, we have:

Rejecting	Accepting	
0, 2, 4, 6	1, 3, 5, 7	
0, 2, 4, 6	1, 3, 5	7
0, 2, 4 | 6	1, 3 | 5	7

We continue:

Rejecting	Accepting	
0, 2, 4, 6	1, 3, 5, 7	
0, 2, 4, 6	1, 3, 5	7
0, 2, 4 | 6	1, 3 | 5	7
0, 2 | 4 | 6	1 | 3 | 5	7
0 | 2 | 4 | 6	1 | 3 | 5	7

Notice that the reduced DFA has 8 states, like the original! This means that the original DFA is already reduced, and cannot be reduced further.

1.2.2 Regular Expressions

Definition 1.2.5. Given an alphabet A, we define a **regular expression**

(a)
- $a \in A$ is a regular expression denoting the language $\{a\}$
- ε is a regular expression denoting $\{\varepsilon\}$
- $\emptyset$ is a regular expression denoting $\emptyset$

(b) If α and β are regular expressions denoting the languages $L(\alpha)$ and $L(\beta)$, respectively, then

- $\alpha \cup \beta$ denotes $L(\alpha) \cup L(\beta)$
- $\alpha \cdot \beta$ denotes $L(\alpha) \cdot L(\beta)$
- α^* denotes $L(\alpha)^*$

By convention, we define precedence of the operations $\cup$, $\cdot$, and * in that order. Thus,

$$b \cdot a^* \cup c = (b \cdot (a^*)) \cup c)$$

A regular expression α over an alphabet A denotes the set of languages which accept α. Thus, we would like to construct an NFA $\underset{\sim}{N}$ such that $L(\underset{\sim}{N}) = L(\alpha)$.

Suppose we wish to construct an NFA for only the letter a, i.e., this NFA rejects all strings but a.

An NFA for only ε would appear as:

And finally, an NFA for only $\emptyset$ is:

Now, suppose we have an NFA for α and β. We wish to determine NFAs for $\alpha \cup \beta$, $\alpha \cdot \beta$, and α^*.

We define

$$\underset{\sim}{N_\alpha} = (A, Q_\alpha, \tau_\alpha, q_0)$$
$$\underset{\sim}{N_\beta} = (A, Q_\beta, \tau_\beta, q_0)$$

such that

$$L(\underset{\sim}{N_\alpha}) = L(\alpha)$$
$$L(\underset{\sim}{N_\beta}) = L(\beta)$$
$$Q_\alpha \cap Q_\beta = \{q_0\}$$

and clarify that these automata are non-returning, i.e., that $q_0 \notin \tau(q_0, s)$ for any s of length 1 or greater.

1.2.3 Solutions of Certain Language Equations

Given a regular expression, we can form an NFA which admits the same language by solving **Language Equations**. We show the following lemma before proceeding to examples:

Lemma 1. If $X = L \cdot X \cup M$ then $X = L^* \cdot M$ is a solution, and is unique if $\varepsilon \notin L$.

Proof. Clearly, $L^* \cdot M$ is a solution, since

$$L^* \cdot M = L \cdot (L^* \cdot M) \cup M$$

. To prove uniqueness, suppose s_1 and s_2 are distinct solutions. There must exist a shortest-length string in s_1, say s. □

Consider the following NFA:

	a	b	
$\to 1$	2	1, 3	0
2	$\emptyset$	3	0
3	2, 3	1	1

This admits the following set of equations

$$X_1 = aX_2 \cup bX_1 \cup bX_3 \tag{1}$$
$$X_2 = bX_3 \tag{2}$$
$$X_3 = aX_2 \cup aX_3 \cup bX_1 \cup \varepsilon \tag{3}$$

We substitute (2) into (1) and (3):

$$X_1 = abX_3 \cup bX_1 \cup bX_3$$
$$X_3 = abX_3 \cup aX_3 \cup bX_1 \cup \varepsilon$$

which we rewrite as

$$X_1 = (ab \cup b)X_3 \cup bX_1$$
$$X_3 = (ab \cup a)X_3 \cup bX_1 \cup \varepsilon$$

We now apply our lemma to the equation for X_3

$$X_1 = (ab \cup b)X_3 \cup bX_1$$
$$X_3 = (ab \cup a)^*(bX_1 \cup \varepsilon)$$

We substitute X_3 into the equation for X_1

$$\begin{aligned} X_1 &= (ab \cup b)(ab \cup a)^*(bX_1 \cup \varepsilon) \cup bX_1 \\ &= ((ab \cup b)(ab \cup a)^* \cup b)\, X_1 \cup (ab \cup b)(ab \cup a)^* \cup bX_1 \\ &= ((ab \cup b)(ab \cup a)^* \cup b)\, X_1 \cup (ab \cup b)(ab \cup a)^* \\ &= ((ab \cup b)(ab \cup a)^* \cup b)^*\, (ab \cup b)(ab \cup a)^* \end{aligned}$$

Consider the example:

	a	b	
$\to 1$	2	3	0
2	2	3	0
3	2, 3	1	1

This admits the following system of equations:

$$X_1 = aX_2 \cup bX_3$$
$$X_2 = aX_2 \cup bX_3$$
$$X_3 = aX_2 \cup bX_3 \cup \varepsilon$$

From our lemma, we have $X_2 = a^*bX_3$:

$$X_1 = aa^*bX_3 \cup bX_3$$
$$X_3 = aa^*bX_3 \cup bX_3 \cup \varepsilon$$

which can be simplified:

$$X_1 = (aa^*b \cup b)X_3$$
$$X_3 = (aa^*b \cup b)X_3 \cup \varepsilon$$

Applying our lemma to X_3, we have

$$X_3 = (aa^*b \cup b)^*$$

Subsituting into X_1 gives

$$X_1 = (aa^*b \cup b)(aa^*b \cup b)^*$$

One final example:

	a	b	
$\to 1$	$\emptyset$	1, 2	1
2	1	$\emptyset$	1

$$X_1 = bX_1 \cup bX_2 \cup \varepsilon$$
$$X_2 = aX_1 \cup \varepsilon$$

Substituting our equation for X_2 into X_1 gives

$$\begin{aligned} X_1 &= bX_1 \cup b(aX_1 \cup \varepsilon) \cup \varepsilon \\ &= (b \cup ba)X_1 \cup b \cup \varepsilon \\ &= (b \cup ba)^*(b \cup \varepsilon) \end{aligned}$$

Given any regular expression, the number of states in its corresponding NFA is the same as the number of symbols in the regular expression.

1.2.4 Regular Grammars

1.3 Accepted by Turing Machines

1.4 Exercise Set 1

Exercise 1: Construct DFAs for the following NFAs using the subset construction:

(a)

	a	
$\to 1$	2	0
2	3	0
3	4	0
4	5	0
5	6	0
6	7	0
7	1, 2	1

(b)

	a	b	c	
$\to 1$	2	2	2	1
2	3	1	1, 2	1
3	4	3	$\emptyset$	1
4	5	4	4	1
5	1	5	5	1

(c)

	a	b	c	
$\to 1$	2	2	2	1
2	3	1	2, 3	1
3	4	3	$\emptyset$	1
4	5	4	4	1
5	1	5	5	1

Solution.

(a)

	a	
$\to 1$	2	0
2	3	0
3	4	0
4	5	0
5	6	0
6	7	0
7	1, 2	1
1, 2	2, 3	0
2, 3	3, 4	0
3, 4	4, 5	0
4, 5	5, 6	0
5, 6	6, 7	0
6, 7	1, 2, 7	1
1, 2, 7	1, 2, 3	1
1, 2, 3	2, 3, 4	0
2, 3, 4	3, 4, 5	0
3, 4, 5	4, 5, 6	0
4, 5, 6	5, 6, 7	0
5, 6, 7	1, 2, 6, 7	1
1, 2, 6, 7	1, 2, 3, 7	1
1, 2, 3, 7	1, 2, 3, 4	1
1, 2, 3, 4	2, 3, 4, 5	0
2, 3, 4, 5	3, 4, 5, 6	0
3, 4, 5, 6	4, 5, 6, 7	0
4, 5, 6, 7	1, 2, 5, 6, 7	1
1, 2, 5, 6, 7	1, 2, 3, 6, 7	1
1, 2, 3, 6, 7	1, 2, 3, 4, 7	1
1, 2, 3, 4, 7	1, 2, 3, 4, 5	1
1, 2, 3, 4, 5	2, 3, 4, 5, 6	0
2, 3, 4, 5, 6	3, 4, 5, 6, 7	0
3, 4, 5, 6, 7	1, 2, 4, 5, 6, 7	1
1, 2, 4, 5, 6, 7	1, 2, 3, 5, 6, 7	1
1, 2, 3, 5, 6, 7	1, 2, 3, 4, 6, 7	1
1, 2, 3, 4, 6, 7	1, 2, 3, 4, 5, 7	1
1, 2, 3, 4, 5, 7	1, 2, 3, 4, 5, 6	1
1, 2, 3, 4, 5, 6	2, 3, 4, 5, 6, 7	0
2, 3, 4, 5, 6, 7	1, 2, 3, 4, 5, 6 7	1
1, 2, 3, 4, 5, 6, 7	1, 2, 3, 4, 5, 6, 7	1

(b)

	a	b	c	
$\to 1$	2	2	2	1
2	3	1	1, 2	1
3	4	3	$\emptyset$	1
1, 2	2, 3	1, 2	1, 2	1
4	5	4	4	1
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	0
2, 3	3, 4	1, 3	1, 2	1
5	1	5	5	1
3, 4	4, 5	3, 4	4	1
1, 3	2, 4	2, 3	2	1
4, 5	1, 5	4, 5	4, 5	1
2, 4	3, 5	1, 4	1, 2, 4	1
1, 5	1, 2	2, 5	2, 5	1
3, 5	1, 4	3, 5	5	1
1, 4	2, 5	2, 4	2, 4	1
1, 2, 4	2, 3, 5	1, 2, 4	1, 2, 4	1
2, 5	1, 3	1, 5	1, 2, 5	1
2, 3, 5	1, 3, 4	1, 3, 5	1, 2, 5	1
1, 2, 5	1, 2, 3	1, 2, 5	1, 2, 5	1
1, 3, 4	2, 4, 5	2, 3, 4	2, 4	1
1, 3, 5	1, 2, 4	2, 3, 5	2, 5	1
1, 2, 3	2, 3, 4	1, 2, 3	1, 2	1
2, 4, 5	1, 3, 5	1, 4, 5	1, 2, 4, 5	1
2, 3, 4	3, 4, 5	1, 3, 4	1, 2, 4	1
1, 4, 5	1, 2, 5	2, 4, 5	2, 4, 5	1
1, 2, 4, 5	1, 2, 3, 5	1, 2, 4, 5	1, 2, 4, 5	1
3, 4, 5	1, 4, 5	3, 4, 5	4, 5	1
1, 2, 3, 5	1, 2, 3, 4	1, 2, 3, 5	1, 2, 5	1
1, 2, 3, 4	2, 3, 4, 5	1, 2, 3, 4	1, 2, 4	1
2, 3, 4, 5	1, 3, 4, 5	1, 3, 4, 5	1, 2, 4, 5	1
1, 3, 4, 5	1, 2, 4, 5	2, 3, 4, 5	2, 4, 5	1

(c)

	a	b	c	
$\to 1$	2	2	2	1
2	3	1	2, 3	1
3	4	3	$\emptyset$	1
2, 3	3, 4	1, 3	2, 3	1
4	5	4	4	1
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	0
3, 4	4, 5	3, 4	4	1
1, 3	2, 4	2, 3	2	1
2, 3	3, 4	1, 3	2, 3	1
5	1	5	5	1
4, 5	1, 5	4, 5	4, 5	1
2, 4	3, 5	1, 4	2, 3	1
1, 5	1, 2	2, 5	2, 5	1
3, 5	1, 4	3, 5	5	1
1, 4	2, 5	2, 4	2, 4	1
1, 2	2, 3	1, 2	2, 3	1
2, 5	1, 3	1, 5	2, 3, 5	1
2, 3, 5	1, 3, 4	1, 3, 5	2, 3, 5	1
1, 3, 4	2, 4, 5	2, 3, 4	2, 4	1
1, 4, 5	1, 2, 4	2, 4, 5	2, 4, 5	1
2, 4, 5	1, 3, 5	1, 4, 5	2, 3, 4, 5	1
2, 3, 4	3, 4, 5	1, 3, 4	2, 3, 4	1
1, 2, 4	2, 3, 5	1, 2, 4	2, 3, 4	1
1, 3, 5	1, 2, 4	2, 3, 5	2, 5	1
2, 3, 4, 5	1, 3, 4, 5	1, 3, 4, 5	2, 3, 4, 5	1
3, 4, 5	1, 4, 5	3, 4, 5	4, 5	1
1, 3, 4, 5	1, 2, 4, 5	2, 3, 4, 5	2, 4, 5	1
1, 2, 4, 5	1, 2, 3, 5	1, 2, 4, 5	2, 3, 4, 5	1
1, 2, 3, 5	1, 2, 3, 4	1, 2, 3, 5	2, 3, 5	1
1, 2, 3, 4	2, 3, 4, 5	1, 2, 3, 4	2, 3, 4	1

□

Exercise 2: Reduce the following DFAs:

(a)

		a	b	
$\to$	1	2	3	0
	2	3	2	1
	3	4	5	0
	4	1	8	1
	5	6	7	0
	6	7	6	1
	7	8	1	0
	8	5	4	1

(b)

		a	b	
$\to$	1	2	3	0
	2	3	2	1
	3	4	5	0
	4	1	8	1
	5	6	7	0
	6	7	6	1
	7	8	1	0
	8	5	5	1

(c) Your result of 1(b).

(d) Your result of 1(c).

Solution.

(a)

Rejecting	Accepting
1, 3, 5, 7	2, 4, 6, 8
1, 3, 5, 7	2, 4, 6, 8

Setting $p = \{1, 3, 5, 7\}$ and $q = \{2, 4, 6, 8\}$:

	a	b	
$\to p$	q	p	0
q	p	q	1

(b)

Rejecting				Accepting			
1, 3, 5, 7				2, 4, 6, 8			
1, 3, 5, 7				2, 4, 6			8
1, 3, 5			7	2, 6		4	8
1	3	5	7	2	6	4	8

The DFA is already reduced.

□

Exercise 3: Construct NFAs for the following regular expressions using the construction given in class; then find the corresponding DFAs; then reduce them:

(a) $(a^2 \cup a^3 \cup a^5)^*$ over $\{a\}$

(b) $(a^2)^*(a^3)^*(a^5)^*$ over $\{a\}$

(c) $(abc \cup ab)^* aa^*(ab)^*$ over $\{a, b, c\}$

(d) $0^*(00 \cup 11)^*(01 \cup 10)^*1^*$ over $\{0, 1\}$

Exercise 4: Construct regular expressions for the languages accepted by the following automata:

(a)

		a	b	c	
$\rightarrow$	1	2	2	2	1
	2	3	1	2, 3	1
	3	4	3	$\emptyset$	1
	4	1	4	4	1

(b)

		a	b	
$\rightarrow$	A	B	C	0
	B	A	C	0
	C	B	A	1