ECS120 Fall 2006

Discussion Notes

October 4, 2006

1 General Information

General course information, my contact information, and discussion notes can be found on my TA website at `wwwcsif.cs.ucdavis.edu/~engle/ta/ecs120-f06/`.

1.1 Asking Questions

There are numerous ways to get your questions answered:

1. **Office Hours.** My office hours are Mondays from 2:00pm - 3:30pm and Wednesdays from 4:00pm - 5:30pm. You can find me in 3104 Kemper Hall during those times.
2. **Discussion.** Discussion sections are Wednesdays from 3:10pm - 4:00pm. The purpose of these sections is to cover extra material regarding the homework, and answer any questions you may have.
3. **Newsgroups.** There are two newsgroups for this class. They are listed on the course and TA websites. You may post any questions you have on the discussion newsgroup. I will be checking this newsgroup regularly and try to answer any questions posted. Other students may answer your questions as well.
4. **Email.** You may send me an email to either `sjengle@ucdavis.edu` or `engle@cs.ucdavis.edu`. I usually respond within 24 hours. However, this should be your last resort for getting your questions answered.

Some of the concepts and notation presented in this class are difficult to communicate through email or discussion boards. This is especially true for state diagrams. Therefore office hours and discussion sections are ideal for asking questions.

1.2 Homework

Homework is assigned every Thursday, and is due the following Thursday before class. Homework 1 is due Thursday, October 5th at the start of class. Homework will give you the practice necessary to pass the midterm and final exams.

1.3 Homework Handin

I will be grading everyone's homework for this class. It is my goal to grade and return homework within a week of being due. There are some things you can do to speed up this process:

- **WRITE LEGIBLY** or type your answers. If I can not read your answers, I can not give you any credit for it.

  It is possible to type up your homework. However, some questions ask for state diagrams. You may attach hand-drawn diagrams if you appropriately label and reference them.

  Also, some proofs require a large amount of mathematical notation. This may be difficult depending on your word processor. You may want to investigate Microsoft Word's Equation Editor, or LaTeX.

- **INCLUDE YOUR NAME** and the names of all people you worked with. Everyone is required to turn in his or her own homework writeup, but you may work in collaborate with other students.

  Note: It is **unacceptable** to copy someone else's homework writeup and simply acknowledge that person. You must write your answers in your own words.

- **CLEARLY MARK** which problem you are providing an answer for. If you type your answers but include hand-drawn state diagrams, be sure to label them clearly.

- **BE VERBOUS** and include the work you did on the problem. Even if you get the answer wrong, you may be able to get partial credit.

  This does not mean you should provide multiple answers and hope one is correct. Include a single answer, and your work detailing how you arrived at that answer.

I will post grades on the `my.ucdavis.edu` course website.

2 Definitions and Notation

2.1 Deterministic Finite Automaton

A **deterministic finite automaton** (DFA) is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where:

1. Q is a finite set of *states*,
2. Σ is a finite set of symbols called the *alphabet*,
3. $\delta : Q \times \Sigma \to Q$ is the *transition function*,
4. $q_0 \in Q$ is the *start state*, and
5. $F \subseteq Q$ is the set of *accept states*.

2.2 Transition Function

The transition function $\delta : Q \times \Sigma \to Q$ tells us where to go from a given state on a given input symbol. It is defined for every pair of states in Q and symbols in Σ. For example $\delta(q_0, 1) = q_1$ tells us that we move to state q_1 from q_0 on input symbol 1.

2.3 Computation

The definition of computation used in this class uses an extension of the transition function. We define this extension as $\widehat{\delta} : Q \times \Sigma^* \to Q$ such that:

$$\widehat{\delta}(q, \epsilon) = q$$

and for string $x \in \Sigma^*$ and character $a \in \Sigma$:

$$\widehat{\delta}(q, xa) = \delta\left(\widehat{\delta}(q, x), a\right)$$

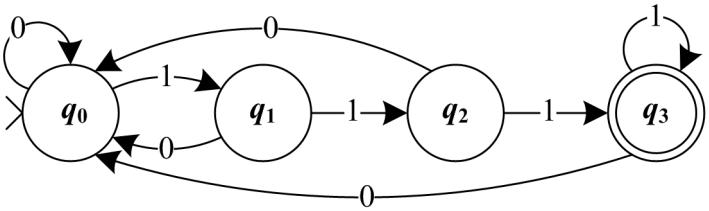
Therefore $\widehat{\delta}(q_0, w)$ gives us the computation of machine $M = (Q, \Sigma, \delta, q_0, F)$ for some word $w \in \Sigma^*$.

3 Examples

For all of these examples, assume $\Sigma = \{0, 1\}$ and let $\sigma \in \Sigma$ be a character.

3.1 Ends with 111

Define a DFA that recognizes the language $L_1 = \{w \in \Sigma^* \mid w \text{ ends with } 111\}$.



The formal definition of this machine is $M_1 = (Q_1, \Sigma, \delta_1, q_0, F_1)$ where:

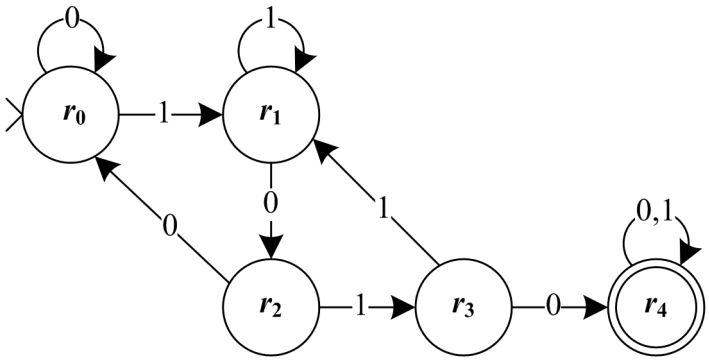
- $Q_1 = \{q_0, q_1, q_2, q_3\}$
- $F_1 = \{q_3\}$

and $\delta_1 : Q_1 \times \Sigma \to Q_1$ is given by the following table:

	0	1
q_0	q_0	q_1
q_1	q_0	q_2
q_2	q_0	q_3
q_3	q_0	q_3

3.2 Substring 1010

Define a DFA that recognizes the langauage $L_2 = \{w \in \Sigma^* \mid w \text{ contains substring } 1010\}$.



Go through examples 111010, 10110100.

The formal definition of this machine is $M_2 = (Q_2, \Sigma, \delta_2, r_0, F_2)$ where:

- $Q_2 = \{r_0, r_1, r_2, r_3, r_4\}$
- $F_2 = \{r_4\}$

and $\delta_2 : Q_2 \times \Sigma \to Q_2$ is given by the following table:

	0	1
r_0	r_0	r_1
r_1	r_2	r_1
r_2	r_0	r_3
r_3	r_4	r_1
r_4	r_4	r_4

3.3 Union of L_1 and L_2

Using M_1 and M_2, construct a DFA M_3 for $L_3 = L_1 \cup L_2$.

The formal definition of this machine is $M_3 = (Q_3, \Sigma, \delta_3, (q_0, r_0), F_3)$ where:

- $Q_3 = Q_1 \times Q_2$
- $F_3 = \{(q, r) \mid q \in F_1 \text{ or } r \in F_2\}$

Construct $\delta_3 : Q_1 \times Q_2 \times \Sigma \to Q_1 \times Q_2$ such that:

$$\delta_3(q, r, \sigma) = (\delta_1(q, \sigma), \delta_2(r, \sigma))$$

	0	1
(q_0, r_0)	(q_0, r_0)	(q_1, r_1)
(q_0, r_1)	(q_0, r_2)	(q_1, r_1)
(q_0, r_2)	(q_0, r_0)	(q_1, r_3)
(q_0, r_3)	(q_0, r_4)	(q_1, r_1)
(q_0, r_4)	(q_0, r_4)	(q_1, r_4)
(q_1, r_0)	(q_0, r_0)	(q_2, r_1)
	...	

4 Homework 1 Hints

We assume you have already gone through chapter 0 in the book. Please be sure to read this chapter so you understand the notation used in this class.

Problem 1

You are asked to assume that friendship is symmetric and anti-reflexive. These terms are defined on page 9 of your book.

A relation R is **reflexive** if for every x, we have xRx. For example, the following relations are reflexive:

$$\begin{aligned} x &= x \\ x &\geq x \\ x &\leq x \\ x &\subseteq x \end{aligned}$$

The following relations do not hold, and hence are not reflexive:

$$\begin{aligned} x &\neq x \\ x &> x \\ x &< x \\ x &\subset x \end{aligned}$$

A relation R is **symmetric** if for every x and y, xRy implies yRx. For example, $x = y$ implies that $y = x$ and is therefore symmetric.

Essentially, this is just a more formal way of saying that you may not be friends with yourself, and friendship is mutual.

Also, you are asked to use the **pigeonhole principle**. The basic idea behind the pigeonhole principle is that if you have more pigeons than holes, at least two pigeons will have to be placed in the same hole. This principle was used in class on Tuesday 10/03 to prove that a DFA is minimal.

Problem 2

Proof by induction is discussed on page 22. We also did a proof by induction in class on 10/03 to prove for all $x, y \in \Sigma^*$:

$$\widehat{\delta}(q, xy) \quad = \quad \widehat{\delta}\left(\widehat{\delta}(q, x), y\right)$$

Base Case:

Let $|\, y \,| = 0$ or $y = \epsilon$. Notice that $xy = x \cdot \epsilon = x$. Therefore we want to show that $\widehat{\delta}(q, xy) = \widehat{\delta}(q, x)$.

We do this by using our claim and the definition of $\widehat{\delta}$:

$$\begin{aligned} \widehat{\delta}(q, xy) &= \delta\left(\widehat{\delta}(q, x), y\right) \\ &= \delta\left(\widehat{\delta}(q, x), \epsilon\right) \\ &= \widehat{\delta}(q, x) \end{aligned}$$

Inductive Step:

We assume our claim holds for all y such that $0 < |\, y \,| \leq k - 1$ where $k > 1$.

Now we must prove our claim holds for k. Since we know $|\, y \,| > 0$ we can rewrite y as:

$$y = y'a$$

where $y' \in \Sigma^*$ and $a \in \Sigma$. Substituting this in, we get:

$$\widehat{\delta}(q, xy) = \widehat{\delta}(q, xy'a)$$

Since a is a character, we can use our definition of $\widehat{\delta}$ to get:

$$\widehat{\delta}(q, xy'a) = \delta\left(\widehat{\delta}(q, xy'), a\right)$$

Notice that $|\, y' \,| = k - 1$, allowing us to use our assumption. Therefore we get:

$$\delta\left(\widehat{\delta}(q, xy'), a\right) = \delta\left(\widehat{\delta}\left(\widehat{\delta}(q, x), y'\right), a\right)$$

Now, to simplify things, let $z = \widehat{\delta}(q, x)$ giving us:

$$\delta\left(\widehat{\delta}\left(\widehat{\delta}(q,x), y'\right), a\right) = \delta\left(\widehat{\delta}\left(z, y'\right), a\right)$$

We can use our definition of $\widehat{\delta}$ to go backwards and get:

$$\delta\left(\widehat{\delta}\left(z, y'\right), a\right) = \widehat{\delta}(z, y'a)$$

Since $y = y'a$ we get:

$$\widehat{\delta}(z, y'a) = \widehat{\delta}(z, y)$$

And finally since $z = \widehat{\delta}(q, x)$ we have:

$$\widehat{\delta}(z, y) = \widehat{\delta}\left(\widehat{\delta}(q, x), y\right)$$

This gives us our desired result:

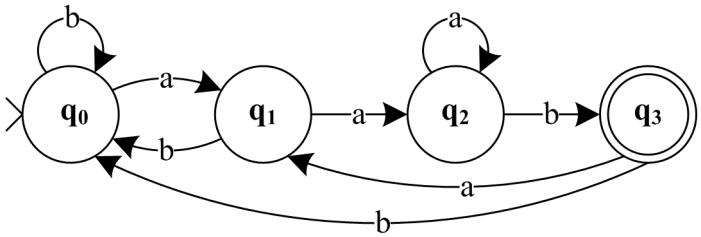
$$\widehat{\delta}(q, xy) = \widehat{\delta}\left(\widehat{\delta}(q, x), y\right)$$

which proves our claim by induction.

Problem 3

The language $L = \{01^*0\}$ should accept strings 011110 or 00, but should not accept the strings 01010 or 101110. We did a similar proof in class 10/03 for a different language and DFA.

In class, we wanted to provide a minimal DFA for the language $L = \{w \in \Sigma^* \mid w \text{ ends with } aab\}$ where $\Sigma = \{a, b\}$. We are given the DFA $M = \{Q, \Sigma, \delta, q_0, F\}$ represented by the following state diagram:



Claim: M is a minimal DFA for $L(M)$.

Suppose: For the sake of contradiction, suppose that there exists a smaller DFA $M' = \{Q', \Sigma', \delta', q_0', F'\}$ for L such that $\mid Q' \mid \leq 3$.

Consider the following strings:

$$\begin{aligned} x_1 &= \epsilon \\ x_2 &= a \\ x_3 &= aa \\ x_4 &= aab \end{aligned}$$

These strings are chosen such that the computation of these strings takes us into each of the four states in M. Observe:

$$\begin{aligned} \widehat{\delta}\,(q_0, \epsilon) &= q_0 \\ \widehat{\delta}\,(q_0, a) &= q_1 \\ \widehat{\delta}\,(q_0, aa) &= q_2 \\ \widehat{\delta}\,(q_0, aab) &= q_3 \end{aligned}$$

By the pigeonhole principle, two of these computations $\widehat{\delta}$ on strings x_1 to x_4 must yield the same state in M'. Therefore, we must show for each pair of computations $\left(\widehat{\delta}(q_0, x_i), \widehat{\delta}(q_0, x_j)\right)$ that:

$$\widehat{\delta}(q_0, x_i) \neq \widehat{\delta}(q_0, x_j)$$

There are $\binom{4}{2}$ cases we must show contradict our assumption. One of these cases is illustrated below:

Case 1: Show contradiction for x_1 and x_2. We start with the following statement:

$$\widehat{\delta}(q_0, \epsilon) = \widehat{\delta}(q_0, a)$$

Notice that by the definition of $\hat{\delta}$ we can pad both sides with the same string without affecting the equality. We pad each string with ab to get:

$$\widehat{\delta}(q_0, ab) = \widehat{\delta}(q_0, aab)$$

However, our language should accept the string aab but not the string ab. Therefore $\widehat{\delta}(q_0, ab) \notin F$ but $\widehat{\delta}(q_0, aab) \in F$. Thus these two computations can not result in the same state, giving us a contradiction.

By proving each of the cases results in a contradiction, we prove that our DFA is indeed minimal.

Problem 4

You may provide either a state diagram of the DFA, or the formal definition of a DFA that accepts the languages described.

4(a)

Several similar DFAs were given in class. Also, see Figure 1.22 in the book on page 44.

4(b)

Taking the complement of a DFA was discussed in class on 9/28. You simply take the complement of F for your new DFA.

4(c)

We can take the union of two languages for this one. Let $L_1 = \{w \in \Sigma^* \mid w \text{ has an even number of 0s}\}$ and $L_2 = \{w \in \Sigma^* \mid w \text{ has an even number of 1s}\}$. Define DFAs for both L_1 and L_2 and then use the algorithm described in class on 10/08 to combine them.

4(d)

You can try to provide the complement of $\{11, 111\}$ directly, or first find the DFA for $\{11, 111\}$ and take the complement of the DFA.

4(e)

This is a challenging problem. The language you want to accept is any binary encoding w such that w is 0 modulo 5. I suggest drawing a table and looking at the different cases (when w is 1 modulo 5, 2 modulo 5, and so on).

A minor detail: you should not accept the empty string. Notice that the language is:

$$\{\epsilon, 0, 00, 000, \dots\} \circ \{0, 101, 1010, \dots\}$$

This is basically saying we accept any number divisible by five, including 0, with any number of 0s front-padding our number. Hence we accept 101, 0101, 00101, and so on.