OUR COVERAGE OF CHAPTER 1 OF SIPSER

JOEL FRIEDMAN

The following is an outline of what I plan to cover and test on in Sipser's textbook Chapter 1.

(1) The definition of a finite automaton (DFA, the D to emphasize "deterministic") and of a non-deterministic finite automata (NFA).
(2) The Myhill-Nerode Theorem (Exercise 1.52 in Sipser's textbook) to determine which languages can be recognized by a DFA and how many states are required; and
(3) the use of NFA's and a comparison of NFA's to DFA's.

In class we will cover some of the sample exam problems given in the Sample Exam Problem section below.

To save you some time, once you see the definition of a finite automaton (see Section 1.1 of Sipser) you should be able to do all the exercises below, provided you know the following information (all of which is found in various places in Chapter 1 of Sipser's textbook):

(1) a finite automaton is called a DFA, the "D" for deterministic;
(2) NFA's are just like DFA's except that (1) you can transition to any number of states upon reading a letter, and (2) for convenience you can transition on the "empty string";
(3) For any language, $L \subset \Sigma^*$ and word $w \in \Sigma^*$ we set the "future of L after seeing w" to be
$$\text{Future}(L, w) = \{u \in \Sigma^* \mid wu \in L\};$$
the number of futures of a language is precisely the minimum number of states in a minimal DFA recognizing L (if this number of futures is infinite then there is no DFA recognizing L).
(4) Any NFA whose state set is Q can be viewed as a DFA where the set of states is $2^Q = Power(Q)$, i.e., the set of all subsets of Q. In practice:
   (a) Consider an NFA whose state set is Q. To see if the NFA recognizes a word $w \in \Sigma^*$, we need only remember which states of Q can be reached each time we process a letter of w; this information is a Boolean string of size $|Q|$. This means that an NFA can be run in roughly $|Q|$ times the work it would take if it were a DFA.
   (b) Given two DFA's, the concatenation of the languages they recognize can be recognized by an NFA which consists of the two DFA's, where the initial state of the NFA is the initial state of the DFA for the first

Research supported in part by an NSERC grant.

language, and where the final states of the first DFA each have an "empty word" transition to the second DFA.

(c) Similarly, the "star" of a regular language is easy to describe as an NFA.

Again, the above material is all we will cover, and we will not cover (1) GNFA's and the formal procedure to convert a DFA to a regular expression (is there any practical reason to do this?); (2) the Pumping Lemma (an easy consequence of the Myhill-Nerode theorem, based on the clever observation that if u and uv have the same future, then this same future is shared also by $uv^2, uv^3, \ldots$).

In class we will solve some of the problems below.

Sample Exam Problems

(1) For the following languages, give a diagram of a DFA that recognizes any of languages in Sipser Exercises 1.4, 1.5, and 1.6.

(2) For the following alphabets, Σ, and languages, $L \subset \Sigma^*$, compute Future(L, w) for every $w \in \Sigma^*$ and use this to construct the DFA with the minimum number of states for the language, L.
   (a) $\Sigma = \{1\}$, L is the langauge of words with at least three 1's.
   (b) $\Sigma = \{0, 1\}$, L is the langauge of words with at least three 1's.
   (c) $\Sigma = \{1\}$, L is the langauge of words with exactly three 1's.
   (d) $\Sigma = \{0, 1\}$, L is the langauge of words with exactly three 1's.
   (e) $\Sigma = \{1\}$, L is the langauge of words with an odd number of 1's.
   (f) $\Sigma = \{0, 1\}$, L is the langauge of words with at least three 1's and at least two 1's.
   (g) $\Sigma = \{0, 1\}$, $L = \{0^*1^*\}$;
   (h) $\Sigma = \{0, 1\}$, $L = \{10^*1^*\}$;

(3) Show that the following languages, $L \subset \Sigma^*$, cannot be recognized by a DFA by that there are infinitely many futures for L (i.e., Future(L, w) takes on infinitely many values):
   (a) $\Sigma = \{1\}$, $L = \{1^n$ such that n is a perfect square$\}$ [Hint: consider $w_m = 1^{m^2+1}$, and use the fact that there is no perfect square between $m^2 + 1$ and $m^2 + 2m$.]
   (b) $\Sigma = \{1\}$, $L = \{1^n$ such that n is a prime number$\}$ [Hint: use the fact that the number of primes is infinite, and show that for $m \geq 2$ there is no prime number between $m! + 2$ and $m! + m$.]
   (c) $\Sigma = \{0, 1\}$, $L = \{0^n 1^n\}$ [Hint: show that Future$(L, 0^m)$ contains 1^m but does not contain any other string of 1's.]
   (d) $\Sigma = \{0, 1\}$, $L = \{0^n 1^{n+2}\}$ [Hint: show that Future$(L, 0^m)$ contains 1^{m+2} but does not contain any other string of 1's.]
   (e) $\Sigma = \{0, 1\}$, $L = \{0^n 1^{2n}\}$ [Hint: show that Future$(L, 0^m)$ contains 1^{2m} but does not contain any other string of 1's.]

(4) Explain the following the questions regarding NFA's:
   (a) Write an NFA for the languages in Sipser Exercise 1.7
   (b) Any NFA with k states can be written as a DFA with 2^k states. Would you want to use this observation to implement an NFA in practice?

(c) Why are NFA's convenient to show that the concatenation of two regular languages is regular?
(d) Why are NFA's convenient to show that the star of a regular language is regular?
(e) Under which circumstances is it useful to do the following:
  (i) convert a regular expression to an NFA;
  (ii) convert a regular expression to a DFA; and
  (iii) convert a DFA or NFA to a regular expression.

(5) Argue that the class of regular languages is closed under the following operations: (1) complementation, (2) intersection, (3) concatenation, (4) union, and (5) star. Explain why it is easy to do (1)–(3) by considering DFA's alone, while to show (4) and (5) it is easier to consider NFA's.

(6) Show that a language and its complement have the same number of futures by arguing that

$$\text{Future}(L^{\text{comp}}, w) = \big(\text{Future}(L, w)\big)^{\text{comp}}$$

where complementation in both cases means complementation with respect to Σ^* where Σ is the alphabet over which L is defined.

(7) Show that if for a language $L \subset \Sigma^*$ and $w_1, w_2 \in \Sigma^*$ we have

$$\text{Future}(L, w_1) = \text{Future}(L, w_1 w_2)$$

then for any $m = 2, 3, \ldots$ we have

$$\text{Future}(L, w_1) = \text{Future}\Big(L, w_1 (w_2)^m\Big).$$

(This is the basic idea behind the "Pumping Lemma.")

Solutions to some problems appearing two pages from now

Solutions to some problems appear starting on the next page

Solutions to Some Problems

Note that Problems (2) and (3) contain many parts which are all variants of the same question; hence we provide only one or two solutions from these problems.

(1) Some solutions are given in Siper's textbook.

(2) (a) $\Sigma = \{1\}$, L is the langauge of words with at least three 1's. **Solution:** Future$(L, 1^n)$ is 1^* for $n \geq 3$, and for $n < 3$ this is the set of words with at least $3 - n$ 1's; hence there are four possible futures, and they form the following DFA: the initial state is represented by the future $L = 1^3 1^*$; which transitions under 1 to $1^2 1^*$; which transitions under 1 to 11^*; which transitions under 1 to 1^*; which transitions to itself. This is a four state DFA.

(b) $\Sigma = \{0, 1\}$, L is the langauge of words with at least three 1's. **Solution:** Here Future(L, w) is just as in the previous problem, except that any occurrence of 0 in w is ignored. Hence we get the futures consisting of all u that has at least $3 - n$ 1's in them, for $n = 0, 1, 2, 3$, and we get the same four state DFA except that upon reading the letter 0 we simply transition to the same state.

(3) Show that the following languages, $L \subset \Sigma^*$, cannot be recognized by a DFA by that there are infinitely many futures for L (i.e., Future(L, w) takes on infinitely many values):

(a) $\Sigma = \{1\}$, $L = \{1^n$ such that n is a perfect square$\}$ [Hint: consider $w_m = 1^{m^2+1}$, and use the fact that there is no perfect square between $m^2 + 1$ and $m^2 + 2m$.] **Solution:** The future of a word $w_m = 1^{m^2+1}$ contains 1^{2m} but nothing smaller. Hence the futures of all these w_m are distinct.

Alernate solution: Since the number of perfect squares is infinite, the future of any word contains some word 1^k. If there were a finite number of futures, then there would be a k_0 such that each future would contain some 1^k with $k \leq k_0$. But the word $w_m = 1^{m^2+1}$ contains no 1^k with $k < 2m$, so it is impossible to have a finite number of words.

The other solutions are very similar to the first part, given the hints.

(4) Explain the following the questions regarding NFA's:

(a) Sipser Exercise 1.7: Some solutions given in the text.

(b) Any NFA with k states can be written as a DFA with 2^k states. Would you want to use this observation to implement an NFA in practice? **Solution:** No. To implement an NFA with Q states takes roughly only $|Q|$ time longer than if it was a DFA (we simply have to record what possible states we are in after reading each letter, and this record is essentially a binary string over $|Q|$). To write out all states and transitions in the associated DFA would take exponential time on $|Q|$.

(c) Why are NFA's convenient to show that the concatenation of two regular languages is regular? **Solution:** Given in the previous section.

(d) Why are NFA's convenient to show that the star of a regular language is regular? **Solution:** Given a DFA, to "star" the recognized language

we simple add "empty word" transitions from each final state to the initial state.

(e) Under which circumstances is it useful to do the following:
  (i) convert a regular expression to an NFA; **Solution:** certainly when dealing with regular expressions that allow for concatenations or stars;
  (ii) convert a regular expression to a DFA; **Solution:** when working with very limited regular expressions (e.g., that search for strings individual strings), where NFA's are not needed;
  (iii) convert a DFA or NFA to a regular expression **Solution:** I don't know of any situation where it is beneficial to convert a DFA or NFA into a regular expression. Theoretically it is interesting to know the equalence of regular languages with those described by regular expressions.

(5) Argue that the class of regular languages is closed under the following operations: **Solution:**
  (a) complementation: interchange the final and non-final states;
  (b) intersection and union: form the product of the two DFA's (i.e., the new DFA has state set $Q_1 \times Q_2$ where Q_1, Q_2 are the state sets of the original DFA's); the constructions are similar except for the set of final states; note that to implement the product of two DFA's you can leave the DFA's alone and simply compute the product state upon reading a letter, i.e., which state you would be in for each of the two DFA's separately;
  (c) concatenation and star: described in the previous section.

(6) Show that a language and its complement have the same number of futures by arguing that

$$\mathrm{Future}(L^{\mathrm{comp}}, w) = \big(\mathrm{Future}(L, w)\big)^{\mathrm{comp}}$$

where complementation in both cases means complementation with respect to Σ^* where Σ is the alphabet over which L is defined. **Solution:** This is straightforward: for any u we have that $wu \in L^{\mathrm{comp}}$ iff $wu \notin L$; but the set of u such that $wu \notin L$ is precisely the complement of the set of u such that $wu \in L$. The futures of L^{comp} are in one-to-one correspondence with their complements in Σ^*, and by the previous sentence these are precisely the futures of L.

(7) Show that if for a language $L \subset \Sigma^*$ and $w_1, w_2 \in \Sigma^*$ we have

$$\mathrm{Future}(L, w_1) = \mathrm{Future}(L, w_1 w_2)$$

then for any $m = 2, 3, \ldots$ we have

$$\mathrm{Future}(L, w_1) = \mathrm{Future}\Big(L, w_1 (w_2)^m\Big).$$

(This is the basic idea behind the "Pumping Lemma.") **Solution:** Let

$$\mathcal{F} = \mathrm{Future}(L, w_1) = \mathrm{Future}(L, w_1 w_2)$$

Then for any $u \in \Sigma^*$ we have

$$w_2 u \in \mathcal{F} \iff u \in \mathcal{F}. \tag{0.1}$$

Taking $u = w_2u'$ in (0.1) we have

$$w_2w_2u' \in \mathcal{F} \iff w_2u' \in \mathcal{F}$$

but taking $u = u'$ in (0.1) we have

$$w_2u' \in \mathcal{F} \iff u' \in \mathcal{F}$$

Hence for any u' we have

$$(w_2)^2u' \in \mathcal{F} \iff u' \in \mathcal{F}.$$

Similarly it then follows that

$$(w_2)^3u' \in \mathcal{F} \iff u' \in \mathcal{F},$$

and then the same if $(w_2)^3$ is replaced by $(w_2)^m$ for any $m = 4, 5, \ldots$ But

$$\text{Future}\Big(L, w_1(w_2)^m\Big) = \{u \mid w_1(w_2)^m u \in L\} = \{u \mid (w_2)^m u \in \mathcal{F}\},$$

which, by the above, is the same as

$$\{u \mid u \in \mathcal{F}\} = \text{Future}(L, w_1).$$

Solution 2: Consider the minimal DFA for L. Then w_1 and w_2 are in the same state; call this state q. It follows that if we are in state q, and we follow w_2 from state q, we wind up again in state q. Hence $w_1(w_2)^m$ also winds up in state q. But any two words in state q have the same future, so $w_1(w_2)^m$ has the same future, for $m = 0, 1, 2, \ldots$.

Department of Computer Science, University of British Columbia, Vancouver, BC V6T 1Z4, CANADA, and Department of Mathematics, University of British Columbia, Vancouver, BC V6T 1Z2, CANADA.
E-mail address: jf@cs.ubc.ca or jf@math.ubc.ca
URL: http://www.math.ubc.ca/~jf