6.045J/18.400J: Automata, Computability and Complexity Prof. Ron Rivest

Homework 1

Due: 19 February 2003

Problem 1: Consider a nondeterministic finite automaton with three states (I), (II), (III) and the following transition rules:

- Transition from state (I) to state (I) on input 'a'
- Transition from state (II) to state (III) on input 'a'
- Transition from state (I) to state (I) on input 'b'
- Transition from state (I) to state (II) on input 'b'
- Transition from state (II) to state (III) on input 'b'

In this automaton, state (I) is the start state and state (III) is the only accepting state.

Part A: Draw a state diagram for this NFA.

Part B: Describe the language recognized by this NFA.

Part C: Use the transformation that we discussed in class to create a DFA which is equivalent to this NFA. Draw a state diagram for the resulting DFA.

Problem 2: For each of the following, draw a state diagram for a machine that satisfies the stated property. Additionally, for one of the following, provide a proof that the machine you have created recognizes the indicated language. (From Sipser 1.4 and 1.27)

1. A Finite Automaton that recognizes the language of binary strings that do not contain the substring 0101.
2. A Finite Automaton that recognizes the language of binary strings that contain at least three 1's.
3. A Deterministic Finite Automaton that recognizes the language of binary strings that contain exactly two 1's or an even number of zeros.
4. A Nondeterministic Finite Automaton that recognizes the language of binary strings that contain exactly two 1's or an even number of zeros.
5. Let Σ be the following alphabet:

   $$\{(0,0)(0,1)(1,0)(1,1)\}$$

   A string from this alphabet can be interpreted as two binary numbers (a left number and a right number). For example, the string

   $$(0,1)(1,0)(1,1)(1,0)(0,0)$$

   Can be interpreted as the left number 01110 and the right number 10100. Create a finite automaton that recognizes the language of strings where the left number is greater than the right number.

Problem 3: Consider a new kind of finite automaton called an all-paths-NFA. An all-paths-NFA M is a five tuple $(Q, \Sigma, \delta, q_0, F)$ that recognizes $x \in \Sigma^*$ if every possible computation of M on x ends in a state from F. Note, in contrast, that an ordinary NFA accepts a string if *some* computation ends in an accept state. Prove that all-paths-NFAs recognize the class of regular languages. (Sipser Problem 1.31)

Problem 4: Let Σ be a finite alphabet and let $L \subseteq \Sigma^*$ be a language accepted by some deterministic finite automaton M. (Sipser Problem 1.27) Prove that

$$L' = \{a_1 a_2 a_3 \dots a_{n-1} a_n \mid a_n a_{n-1} \dots a_3 a_2 a_1 \in L\}$$

is accepted by some deterministic finite automaton M'. (Sipser Problem 1.24)