

CS385-Assignment 2

Due Tuesday, September 18

September 14, 2007

1 Exercises from the Text

Make sure that you can answer all parts of Exercises 1.1 and 1.2 of the text. (The answers are supplied; you need not hand these in.)

Exercises 1.3, 1.6(a,d,i,j,m),1.4(a,g). Observe that this is the logical order, since 1.6 just asks you to construct DFAs, while 1.4 asks you to apply a particular method for constructing DFAs. Your answers to all these problems will be state-transition diagrams.

2 Problem on Divisibility Tests (based on Problem 1.37 from the text)

How do you tell if an integer, given its decimal representation, is even? Just look at the rightmost digit, of course! Divisible by 5? Again, it's enough to examine the rightmost digit?

Divisible by 3? Here's an algorithm that you might know: Just add up the digits—the original number is divisible by 3 if and only if the sum of the digits is divisible by 3. Note that this algorithm can be implemented by a finite-state automaton, because as we add the digits we only have to keep track of the remainder on division by 3, not the complete sum. (So there would be three states, labeled 0,1 and 2).

How do we tell whether an integer written in binary is divisible by 3? The sum-of-the-digits trick does not work (101 and 11 have the same digit sum, but the first is five – not divisible by three—and the second is three). Nonetheless, it is still possible to do this by keeping track of the remainder upon division by 3 as we accumulate digits. Note that each time we adjoin a new digit to the right of a number we either multiply by 2:

$$101 \rightarrow 1010(\text{five} \rightarrow \text{ten}),$$

or multiply by 2 and add 1:

$$101 \rightarrow 10101(\text{five} \rightarrow \text{eleven}),$$

so you just have to figure out what these two operations do to the remainder mod 3.

(a) Use this idea to draw the state-transition diagram of a three-state DFA that recognizes the set of binary representations of integers divisible by 3. (Following the book's notation in Problem .37, we'll call this B_3 . So this construction proves that B_3 is regular.)

Here is a different approach to testing binary integers for divisibility by 3: Read the number from right to left, grouping the bits in pairs. Then add up the value of each pair, retaining only the remainder upon division by 3. For example, seventy-five in binary is

1001011.

Grouping in pairs gives

1, 00, 10, 11.

So we add (reading from the right) $3+2+0+1$. Since we only save remainders, we get the successive answers 0,2,2,0, and thus the number is divisible by 3.

(b) Show how to implement this algorithm with a DFA. The resulting automaton reads a binary integer beginning with the rightmost bit, and determines if the number is divisible by 3. It therefore recognizes the language B_3^R (reversal of B_3 .) (In this connection, see Problem 1.31, whose result we will discuss in class: If we know B_3 is regular, we can conclude B_3^R is regular and vice-versa, without having to construct both automata, although here I had you construct them both.)

(c) Now do Problem 1.37. The entire idea of the proof is in the prelude to part (a) above (you can ignore part (b) for these purposes). The challenge is to describe the DFA for B_n in precise mathematical language.