The Theory of Computation

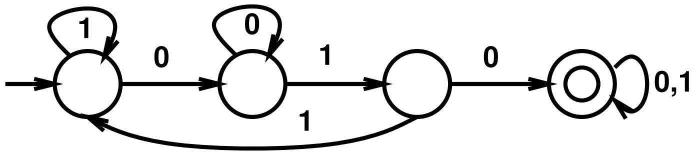
Bernard M.E. Moret

Addison-Wesley-Longman, 1998

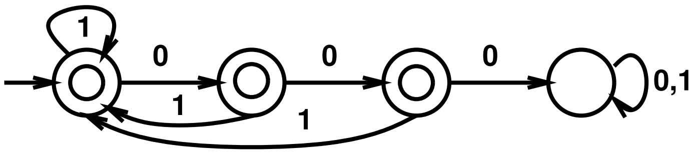
Solution Manual

Chapter 3, Problem 3.4

1. All that the automaton needs to do is to "watch" for the string 010 with three states, moving to an accepting trap as soon as the substring has been observed; so the machine has 4 states in all, one of them accepting, as shown below.

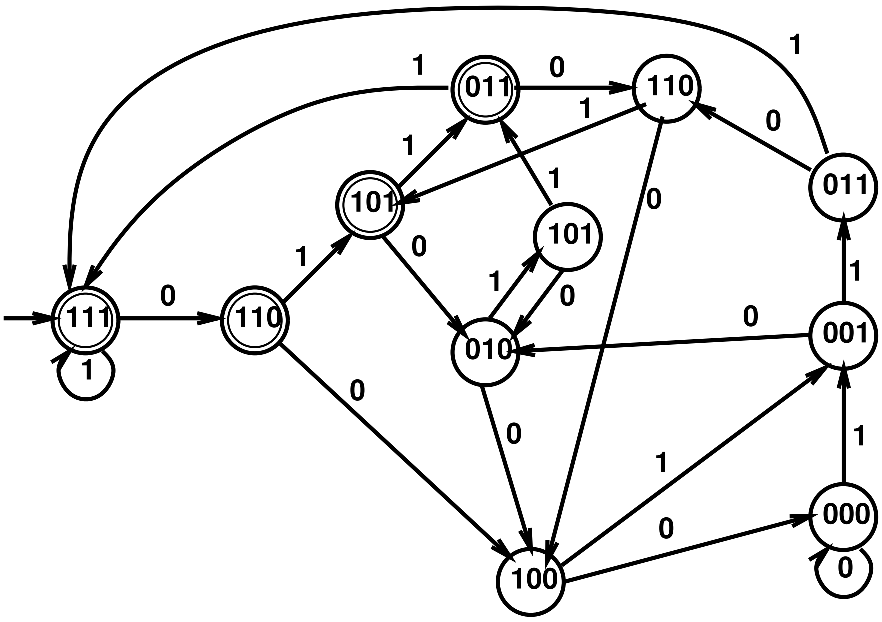


2. The same idea as in the previous problem works, but this time, once the critical substring has been observed, the FA moves to a rejecting trap. Again, we have 4 states, but now 3 of the four are accepting.

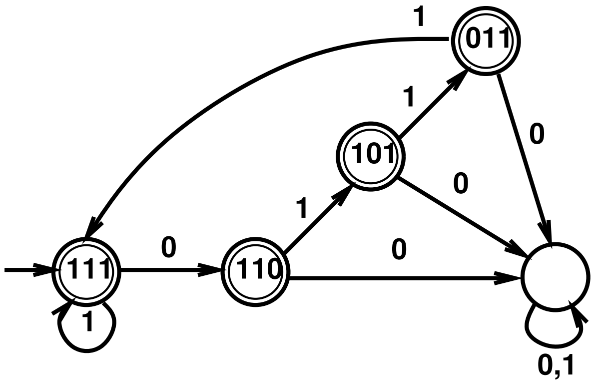


3. Before solving this problem, we need to finish defining it. The statement does not say what to do with strings of length 3 or less; since that leaves us free to make our own choices, we pick what will simplify the final FA as much as possible. Since the final FA keeps a window of 3 characters in its state memory (8 states, defining what has been seen so far, so that a move to the fourth character in any substring simply shifts the 3-character window), we let acceptance/rejection of short strings be defined by that mechanism. Note that windows with two 1s and one 0 exist in both an accepting and a rejecting version, while the window with three 1s exists only as an accepting window, and windows with two or three 0s exist only in rejecting versions. Thus we have 11 states in all, 4 of them accepting. We arbitrarily choose to

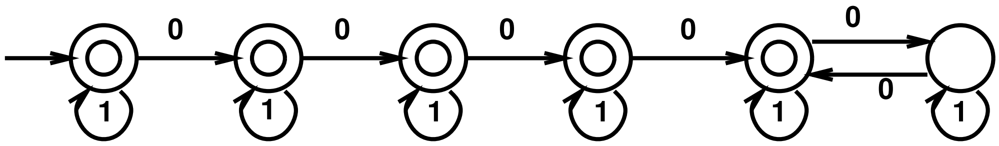
start the machine in a state corresponding to an illusory window of 3 1s, i.e., an accepting state; thus all strings can be regarded as prefixed by three 1s, which means that our short strings (of length less than 4) are handled as follows: we accept all those that have at most one 0 and reject all those that have two or three 0s. (Note that forcing all strings of length less than 4 to be treated in the same way—all rejected or all accepted—considerably increases the length of the FA, since now the first three characters must be treated separately from everything else, giving us an additional 15 states . . .)



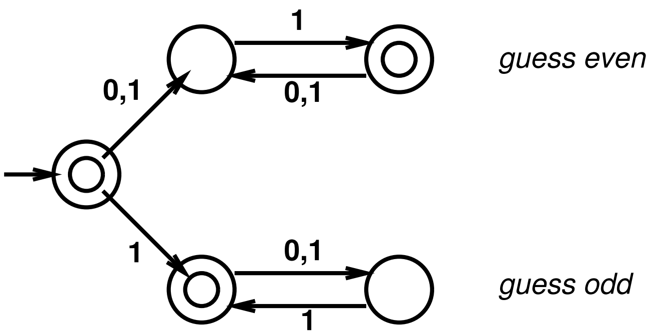
However, this machine does not really do what we are asked: it is forgiving, in that it is willing to go past a substring of length 4 with at most two 1s and still accept later . . . IN fact, we want to reject as soon as we move to a non-accepting state, so we can merge all our non-accepting states into a single trap, which cuts down the machine to just 5 states!

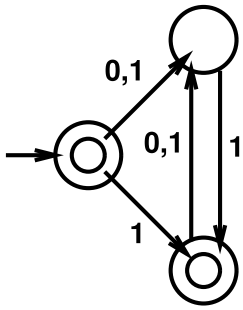


4. In spite of the apparent choice (odd or even), this machine is quite simple. It watches for 0s and counts up to 3 of them, accepting anything; after that, it stops counting, but checks parity of the number of 0s. This gives us a machine with 4 counting states (zero through three 0s) and 2 parity checking states, or 6 states in all, one of them rejecting.



5. It would be much simpler here to give a nondeterministic FA, since an NFA can guess whether the string has odd or even length and thus have two paths from the start state, one for odd-indexed checks and one for even-indexed checks. The resulting machine has five states and is very simple indeed; it can be further simplified down to just three states as seen on the right. (Note that we used the convention that an absence of applicable transitions implies immediate rejection.)





The deterministic version, however, must accomplish both tasks at once; Perhaps a better way to think about the problem is to realize that, as long as we see only 1s, we can accept both odd- and even-length strings; the first 0, however, forces the issue.