The Theory of Computation

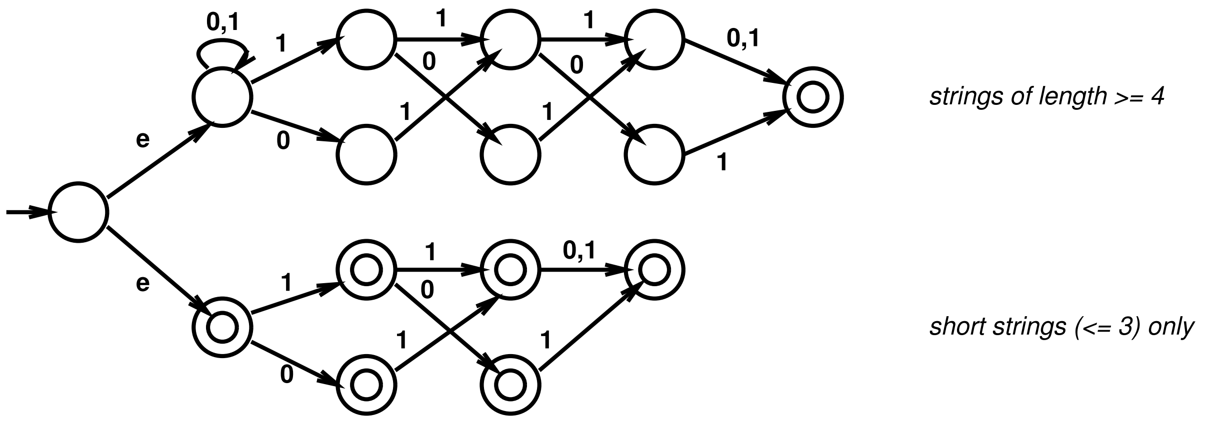
Bernard M.E. Moret

Addison-Wesley-Longman, 1998

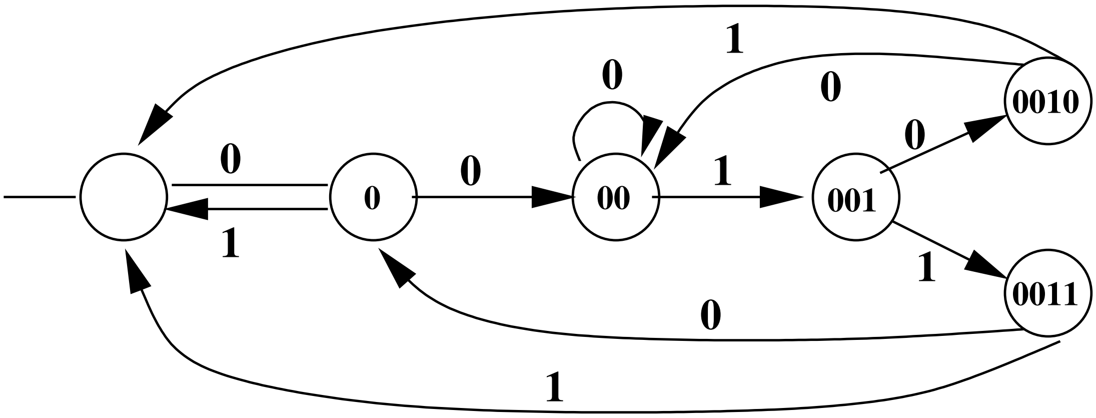
Solution Manual

Chapter 3, Problem 3.5

1. This definition is unambiguous; since we have to keep track of the last four characters, it is probably easiest to design an NFA. We can have the NFA guess when 4 characters remain and check that they do not contain a pair of adjacent 0s; short strings (3 or fewer characters) need independent checking, so our NFA can start by guessing whether the string is short or long.



Actually, a very different outlook can net us an elegant and concise DFA: maintain a window of 4 characters, in which we want to avoid seeing two adjacent 0s. The resulting machine has just 6 states and is purely deterministic, as illustrated below.



2. This definition is somewhat ambiguous, because a pair of 0s could cross the "boundary", i.e., be in the 5th and 4th characters from the end, in which case we do not know whether to count it or not. We choose to count it, i.e., to consider it not covered by the clause about the last 4 characters, so that, for instance, the string 000100 will be rejected.
   We can design an NFA that checks for adjacent pairs of 0s (essentially, any three 0s in a row) and can guess at any time that it is entering the last four characters. Since the basic NFA has effectively 3 states (0, 1, and 2 0s in a row), we can have it move on any character from the first two states to a final string of 3 states that simply swallow four characters in a row; by not having a nondeterministic move on input 0 from the third state (already seen two 0s in a row), we ensure that strings with three 0s in a row where the third 0 is the 4th character from the end will be rejected.