Discrete assignment (Question 8(b))

Mehedi Hasan

February 18, 2021

Problem 0.1. Construct deterministic finite set automata that recognize each of these languages:

(a) The set of bit strings that begin with two 0's.

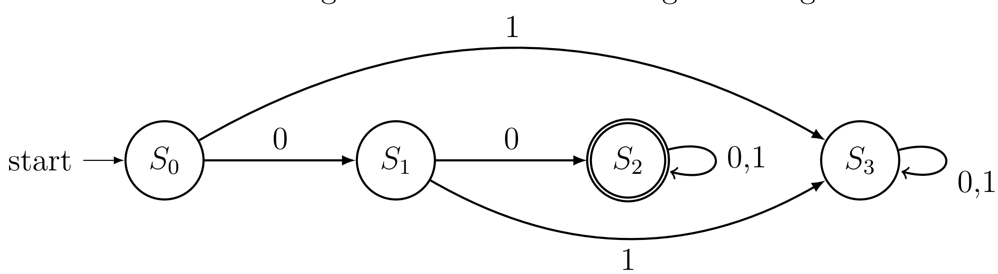
(b) The set of bit strings that contains two consecutive 0's.

(c) The set of bit strings that do not contain two consecutive 0's.

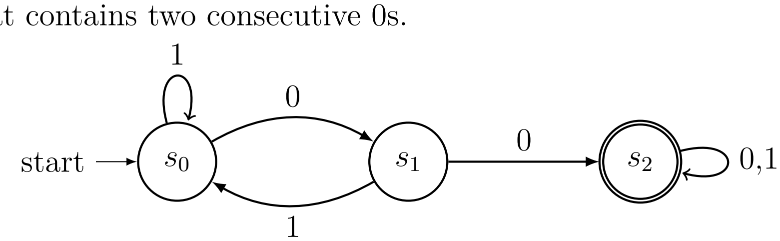
(d) The set of bit strings that end with two consecutive 0's.

Solution.

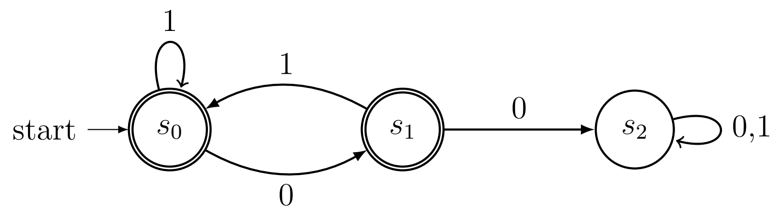
(a) We need to construct a deterministic finite-state automation that recognizes the set of bit strings that begins with two 0s. Besides the start state s_0, we include a nonfinal state s_1; we move to s_1 from s_0 if the first bit is a 0. Next, we add a final state s_2, which we move to from s_1 if the second bit is a 0. When we have reached s_1 we know that the first two input bits are both 0s, so we stay in the state s_2 no matter that the succeeding bits (if any) are. We move to a nonfinal state s_3 from s_0 if the first bit is a 1 and from s_1 if the second bit is a 1. The figure below represents the finite-state automation that recognizes the set of bit strings that begin with two 0s.



(b) We need to construct a deterministic finite-state automation that recognizes the set of bit strings that contain two consecutive 0s. Besides the start state s_0, we include a nonfinal state s_1, which tells us that the last input bit seen is a 0, but either the bit before it was a 1, or this bit was the initial bit of string. We include a final state s_2 that we move from s_1 when the next input bit after a 0 is also a 0. If a 1 follows a 0 in the string, we return to s_0 and begin looking for consecutive 0s all over again. The figure below represents the finite-state automation that recognizes the set of bit strings that contains two consecutive 0s.



(c) We need to construct a deterministic finite-state automation that recognizes the set of bit strings that do not contain two consecutive 0s. Besides the start state s_0, which should be a final state, we include a final state s_1, which we move to from s_0 when 0 is the first input bit. When an input bit is a 1, we return to, or stay in, state s_0. We add a state s_2, which we move to from s_1 when the input bit is a 0. Reaching s_2 tells us that we have seen two consecutive 0s as input bits. We stay in state s_2 once we have reached it; this state is not final. The figure below represents the finite-state automation that recognizes the set of bit strings that do not contain two consecutive 0s.



(d) We need to construct a deterministic finite-state automation that recognizes the set of bit strings that ends with two 0s. Besides the start state s_0, we include a nonfinal state s_1, which we move to if the first bit is 0. We include a final state s_2, which we move to from s_1 if the next input bit after a 0 is also a 0. If an input of 0 follows a previous 0, we stay in state s_2 because the last two input bits are still 0s. Once we are in state s_2, an input bit of 1 sends us back to s_0, and we begin looking for consecutive 0s all over again. We also return to s_0 if the next input is a 1 when are in state s_1. The figure below represents the finite-state automation that recognizes the set of bit strings that ends with two 0s.