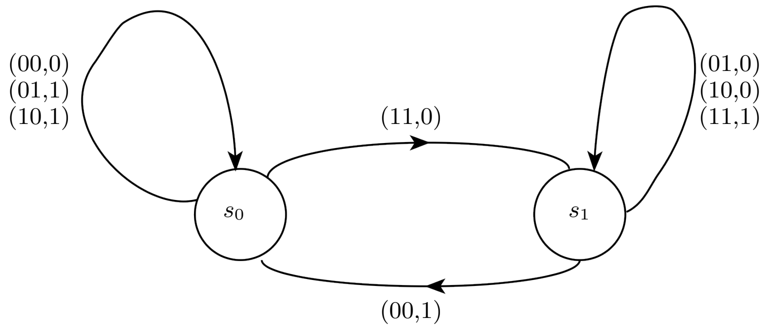
Summary on Lecture 3, April 1st, 2015

Languages and Finite State Machines

More examples. (3) Adding machine. Now we describe a finite state machine which adds integers written as binary sequences. For example, we take a sum $z = x + y$, where $x = 10011101$ and $y = 00110111$. Then

$$\begin{array}{rcrrrrrrrr} x & = & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ +y & = & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ \hline z & = & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \end{array}$$

Now, the input is the set of pairs $I = \{00, 01, 10, 11\}$, and the output is the set $O = \{0, 1\}$. Here is the diagram describing the functions ν and ω:

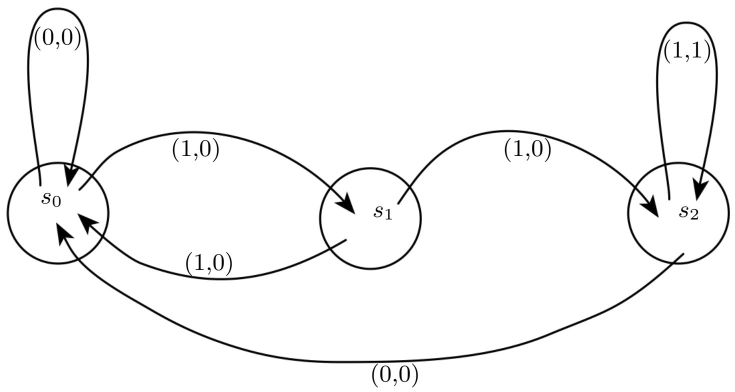


Here the state s_0 has to be a starting point, and the state s_1 serves to carry of 1.

(4) Sequence recognizer. The next example is a finite state machine which recognizes each occurence of the sequence 111 in an arbitrary binary sequence. Namely, an input sequence 111101111011101 should give the output sequence written below the input:

1111011111011101
0011000111000100

Here is the corresponding diagram:

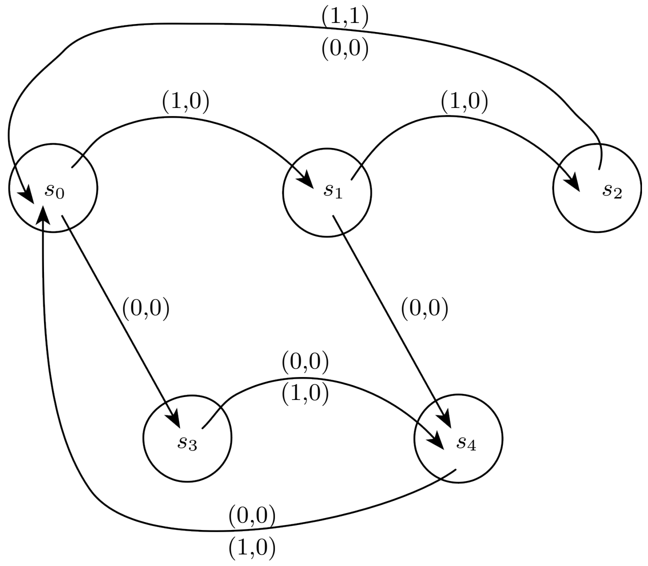


Indeed, we let the sequence 111101111011101 go through the machine. We start with the state s_0. The input 11 will take us to the state s_2 and give the output 00. Then the next 11's will give us output 11. After that 0 will take us to the initial state s_0, and so on. Here $M = (S, I, O, \nu, \omega)$, where $S = \{s_0, s_1, s_2\}$, $I = O = \{0, 1\}$, and the functions ν and ω are given above.

(5) Another sequence recognizer. Here we would like to recognize the occurence of the sequence 111, but with extra condition that the pattern 111 ends in a position that is a multiple of three. To illustrate, we use the same input sequence as above:

1111011111011101
0010000010000000

Here is the corresponding diagram:

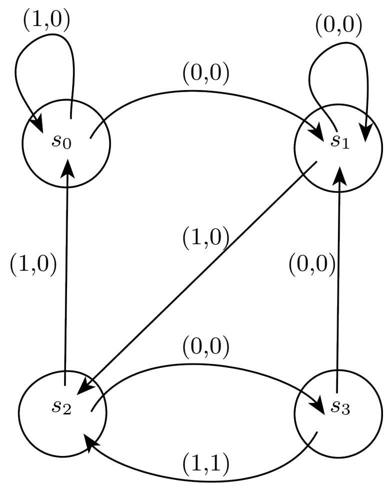


Again, we start at s_0. The states s_3 and s_4 are designed to divert any sequence of the form 0** back to the initial state s_0, and a sequence of the form 10* gets diverted back to s_0.

(6) One more sequence recognizer. Suppose again we have $I = O = \{0, 1\}$. Now we would like for our new machine to recognize the pattern 0101 in the input binary sequence. For instance, if the input sequence is 10111010101111110101, then we have the following output sequence:

10111010101111110101
00000000101000000001

Here is the diagram describing the machine:



In order to achieve this, the finite state machine must have at least four states, a passive state when the previous two entries are 11 (or when no entry has yet been made), an expectant state when the previous three entries are 000 or 100 or 110 (or when only one entry has so far been made and it is 0), an excited state when the previous two entries are 01, and a cannot-wait state when the previous three entries are 010.