Finite Automata with Output

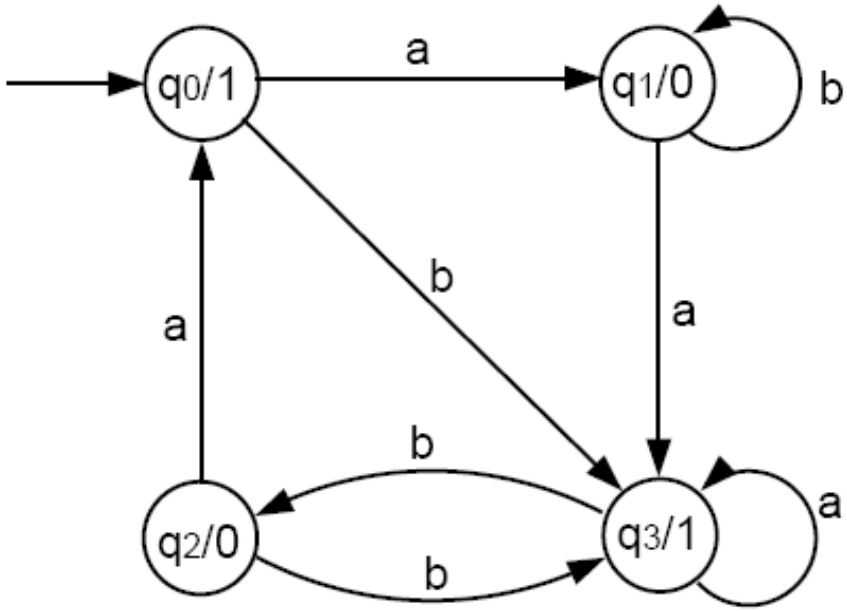
Shakir Ullah Shah

Agenda

- **Moore Machines**

- **Mealy Machines**

- There are two type of machines:
  - Recognizer (Accept/reject)
  - Generator (output)
    - There are two types of machines with output.
      - Moore machine and
      - Mealy machine


q0/1
a
q1/0
b
b
a
a
b
q2/0
b
q3/1
a

Moore Machine Definition

Moore machine is a collection of five things:

1. A finite set of states q_0, q_1, q_2, ..., where q_0 is designated as the start state.

2. An alphabet of letters for forming the input string $\sum$= {a, b, c, ...}.

3. An alphabet of possible output characters **Γ** = {0, 1, 2, ...}.

4. A transition table that shows for each state and each input letter what state is reached next.

5. An output table that shows what character from **Γ** is printed by each state as it is entered.

Notes

- To keep the output alphabet separate from the input alphabet, we give it a different name **Γ** (instead of ∑) and use number symbols {0, 1, ...} (instead of {a, b, ...}).
- We refer to input symbols as **letters**, whereas we refer to output symbols as **characters**.
- We adopt the policy that a Moore machine always begins by printing the character dictated by the mandatory start state. So, if the input string has 7 letters, then the output string will have 8 characters, because it includes 8 states in its path.

Defining a Language

- To change FA into a Moor Machine
  - Name each state
  - Name start state q0
  - Output 0 in all non-final states
  - Output 1 in all final states
- A string is accepted if after it has been completed read in the last letter printed 1.

Example: Moore machine defined by a table

- Input alphabet: $\sum = \{a, b\}$
- Output alphabet: $\Gamma = \{0, 1\}$
- Names of states: q_0, q_1, q_2, q_3 with q_0 being the start state.
- Transition and output table (combined):

Old States	New states after reading		Output by Old State
	a	*b*	
q_0	q_1	q_3	1
q_1	q_3	q_1	0
q_2	q_0	q_3	0
q_3	q_3	q_2	1

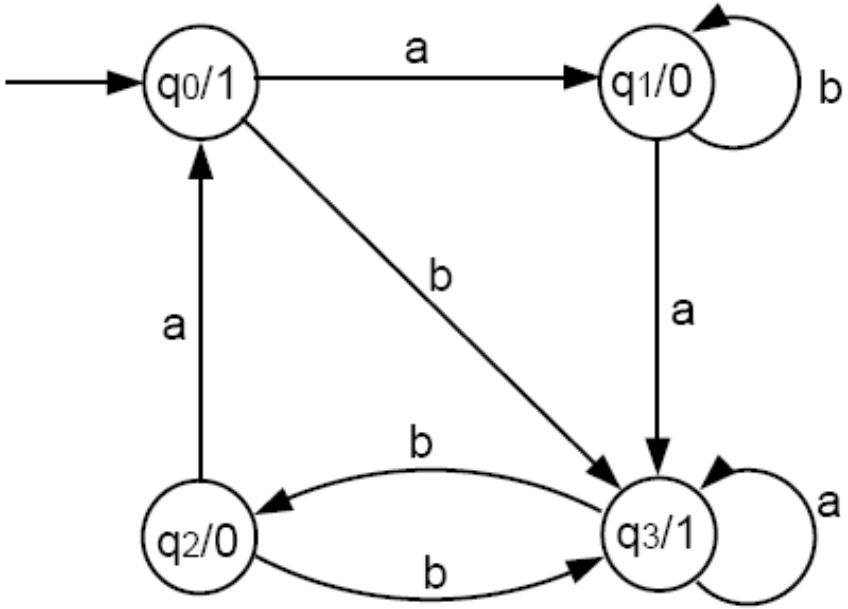
Pictorial Representation

- Moore machines have pictorial representations similar to FAs.

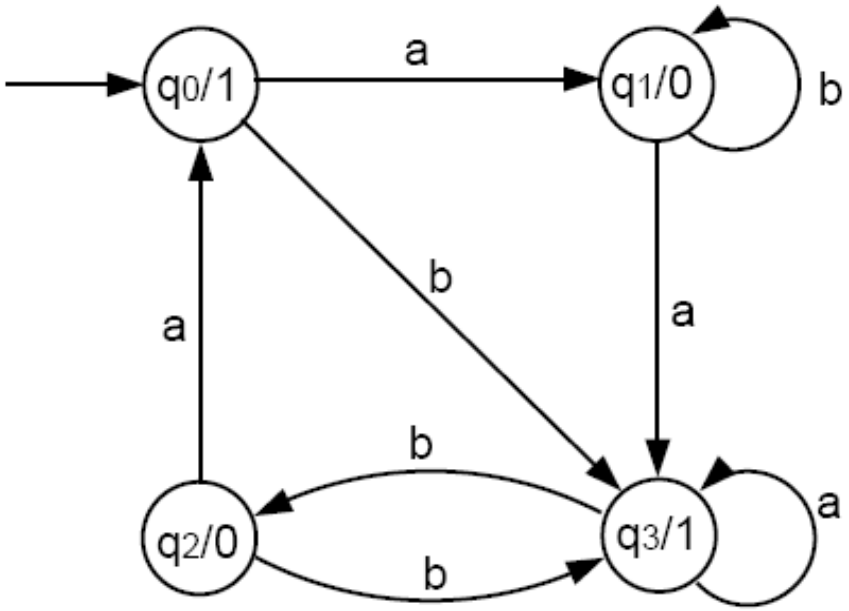
- The difference is that inside each state, in addition to the state name, we also specify the output character printed by that state, using the format *state – name/output*.

- Hence, the Moore machine in the above example has the following pictorial representation:

Example



- We indicate the start state by an outside arrow since there is no room for the usual - sign.
- Given the input string *abab*, the output sequence is 10010.
- Note that the length of the output string is one longer than the length of the input string.



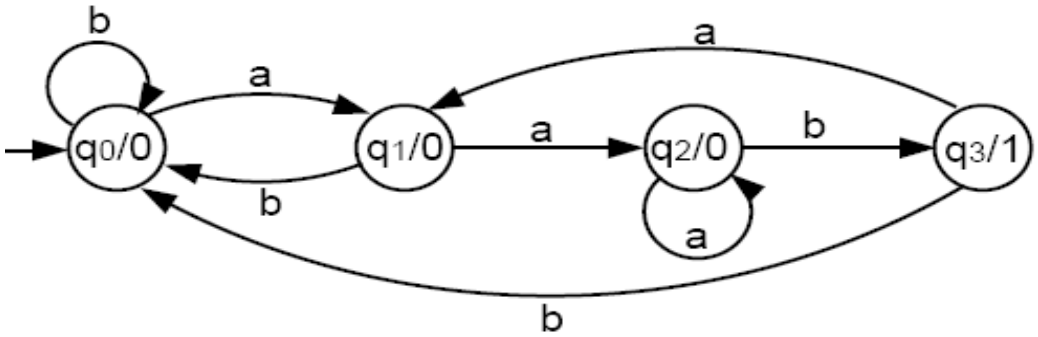
Input		**a**	**b**	**b**	**a**	**b**	**b**	**b**	**a**
State	q_0	q_1	q_1	q_1	q_3	q_2	q_3	q_2	Q_0
output	1	0	0	0	1	0	1	0	1

It may be noted that the length of output string is 1 more than that of input string as the initial state prints out the extra character 1, before the input string is read.

Example

- Suppose we are interested in knowing exactly how many times the substring *aab* occurs in a long input string. The following Moore machine will count this for us:

Example



- Every state of this machine prints out a 0, except for q3, which prints a 1.
- To get to q_3, we must have come from q_2 and have just read a b. To get to q_2, we must have read at least two a's in a row.

- After finding the subtring *aab* and tallying a 1 for it, the machine looks for the next *aab*. Hence, the number of 1's in the output string is exactly the number of substrings *aab* in the input string.
- It accepts all words that end with aab.

Melay machine Definition

A **Mealy machine** is a collection of four things:

1. A finite set of states q_0, q_1, q_2, ..., where q_0 is designated as the start state.

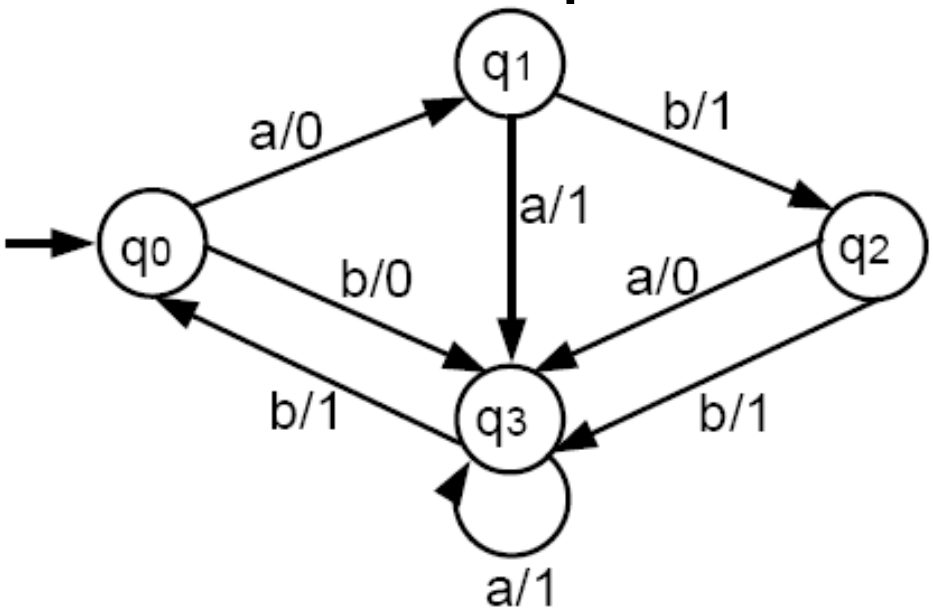
2. An alphabet of letters for forming the input string $\sum$ = {a, b, ...}.

3. An alphabet of possible output characters Γ = {0, 1, ...}.

4. A pictorial representation with states represented by small circles and directed edges indicating transition between states.

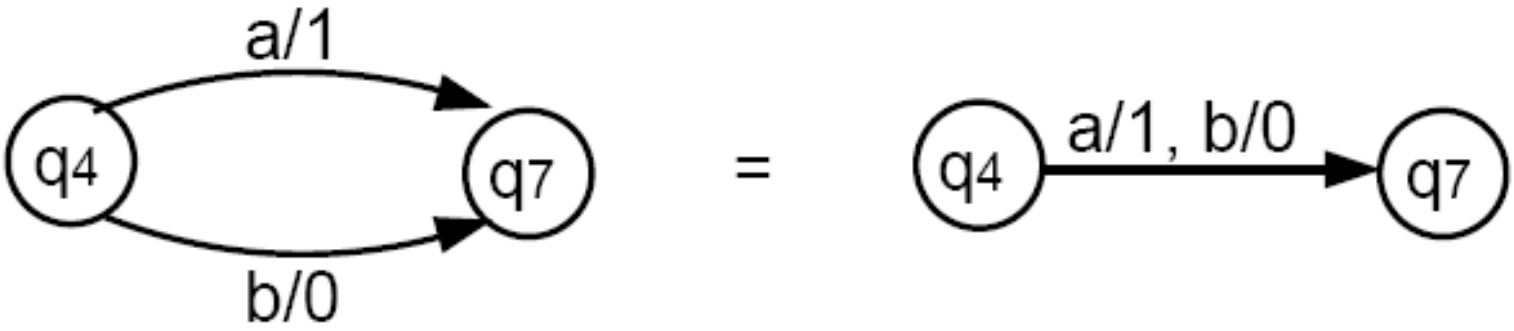
- Each edge is labeled with a compound symbol of the form ***i/o*** where *i* is an input letter and *o* is an output character.
- Every state must have exactly one outgoing edge for each possible input letter.
- The edge we travel is determined by the input letter *i*. While traveling on the edge, we must print the output character *o*.

Example



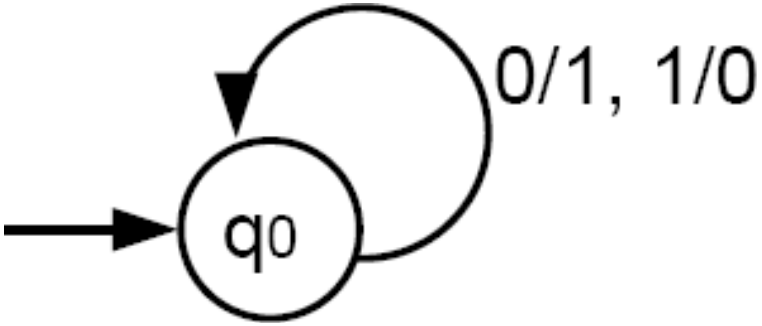
- Given the input string *aaabb*, the output is 01110.
- In a Mealy machine the output string has the same number of characters as the input string has letters.

- A Mealy machine does not define a language by accepting and rejecting input strings: It has no final states.
- However, there is a sense in which a Mealy machine can recognize a language, as we will see later.
- Note the following notation simplification:



Example

- The following Mealy machine prints out the 1’s complement of an input bit string.
- This means that it will produce a bit string that has a 1 whenever the input string has a 0, and a 0 whenever the input has a 1.



- If the input string is 001010, the output will be 110101

Example

- Let consider a Mealy machine, called **increment machine**, which reads a binary number and prints out the binary number that is **one larger**.
- Assume that the input bit string is a binary number fed in backward; that is, unit digit first, then 2's digit, 4's digit, etc.
- The output string will be the binary number that is one greater and that is **generated right to left**.
- The machine will have 3 states: start, owe-carry and no-carry. The owe-carry state represents the overflow when two bits of 1's are added, we print a 0 and we carry a 1.

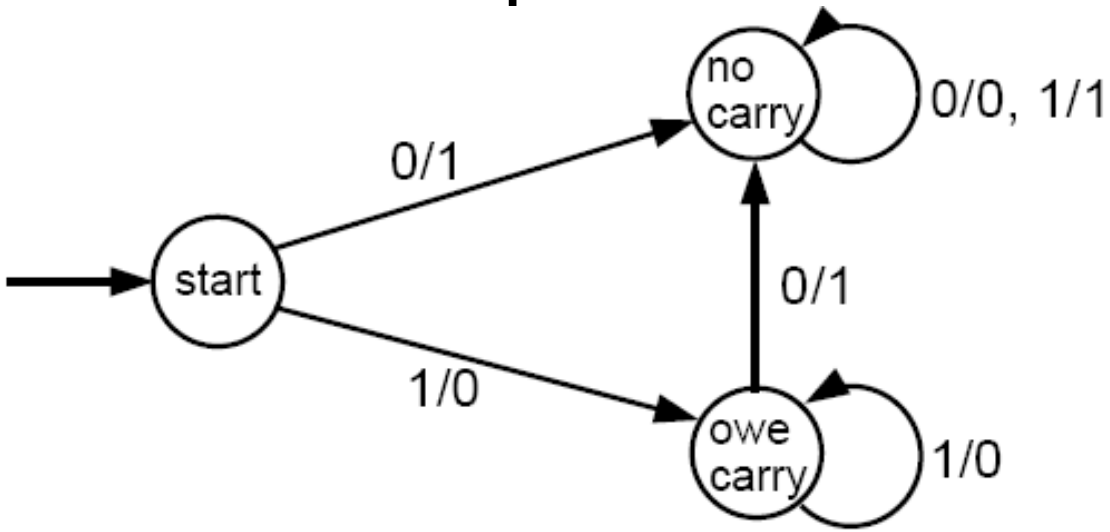
Example contd.

- From the start state, if we read a 0, we print a 1 (incrementing process), and we go to the no-carry state. If we read a 1, we print a 0 (incrementing) and we go to the owe-carry state.

- At any point in the process, in we are in the no-carry state, we print the next bit just as we read it and remains in no-carry.

- However, if we are in the owe-carry state and read a 0, we print a 1 and go to no-carry. If we are in owe-carry and read a 1, we print a 0 and we loop back to owe-carry.

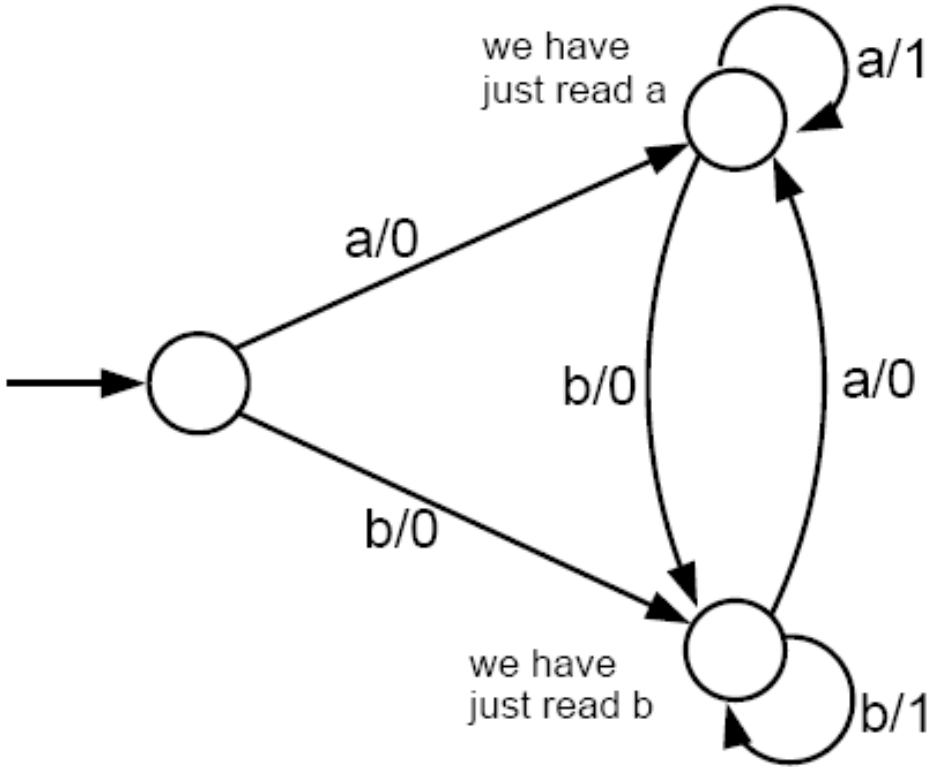
Example contd.



- Let the input string be 1011 (binary representation of 11).
- The string is fed into the machine as 1101 (backwards).
- The output will be 0011, which when reversed is 1100 and is the binary representation of 12.
- In Mealy machine, output length = input length. Hence, if input were 1111, then output would be 0000 (**overflow situation**).

Example

- Although a Mealy machine does not accept or reject an input string, it can recognize a language by making its output string answer some question about the input.
- Consider the language of all words that have a double letter (*aa* or *bb*) in them.
- We can build a Mealy machine that can take an input string of a's and b's, and print out an output string of 0's and 1's such that **if the n-th output character is a 1, it means that the n-th input letter is the second letter in a pair of double letters**.
- The complete picture of this machine is as follows:



- If the input string is *ababbaab*, the output will be 00001010.
- This machine recognizes the occurrences of *aa* or *bb*.
- Note that the triple letter word *aaa* produces the output 011 since the second and third letters are both the back end of a pair of double a's.