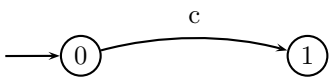
# Finite-State Automata

L245

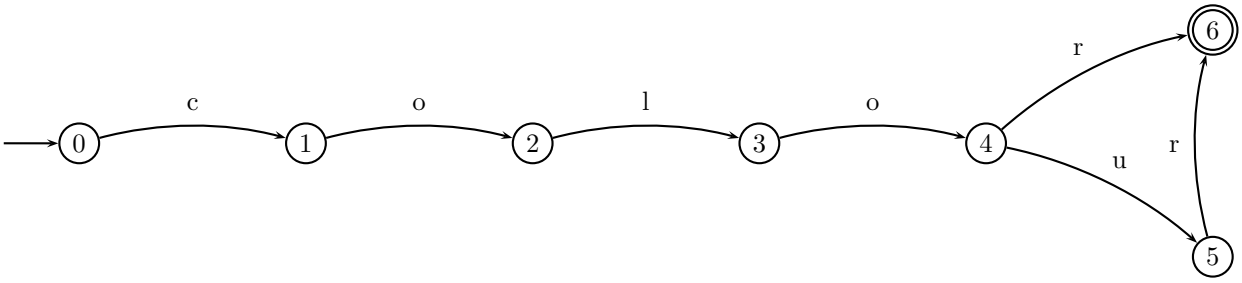
Spring 2017

A **finite-state automaton** (FSA), or finite-state machine (FSM), is a series of **states** and **transitions** between states.

- By moving through the FSA, you try to match an input string.
- The idea is that you can move from state to state, when conditions on the arcs are met.
  e.g. Move from state 0 to state 1 if I encounter the letter *c*. (Otherwise, do nothing.)



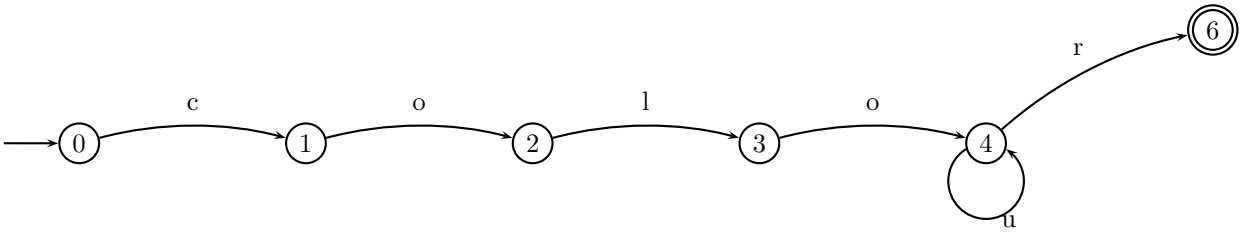

We represent states with **nodes** (circles) and transitions with **arcs** (arrows) in a picture. (Note that the numbers on the nodes are just for our convenience; they don't really "mean" anything.)



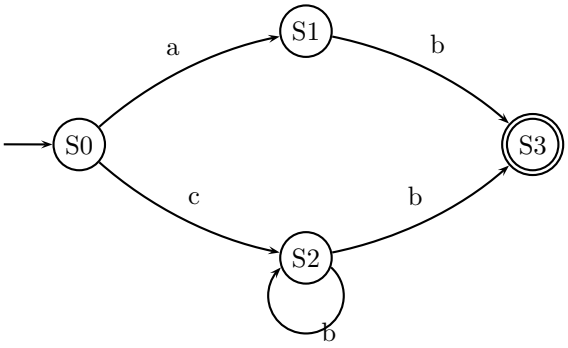

The FSA above matches (or generates) the string *color* or the string *colour*.

- The double lines around node 6 means that this is a **final state** = can end at this state.
- You have to be in a final state when you're done, or it fails—if you match *colo*, you are not done.

Now, the following FSA is similar, but subtly different. How is it different?




What does the following FSA do?




Note:

- You have to have a single start node, but you can have more than one end node.
- The nodes don't necessarily have to go in order from left to right.
- FSAs are what are used to match regular expressions. Logically, they are equivalent.

Let's draw FSAs which recognize the following regular expressions:

- /bla*h/
- /b[lr]a+h/ (equivalently: /b(l|r)a+h/)