# CSE 211 (Theory of Computation)

## Automata Introduction

Dr. Muhammad Masroor Ali

Professor
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology
Dhaka-1000, Bangladesh

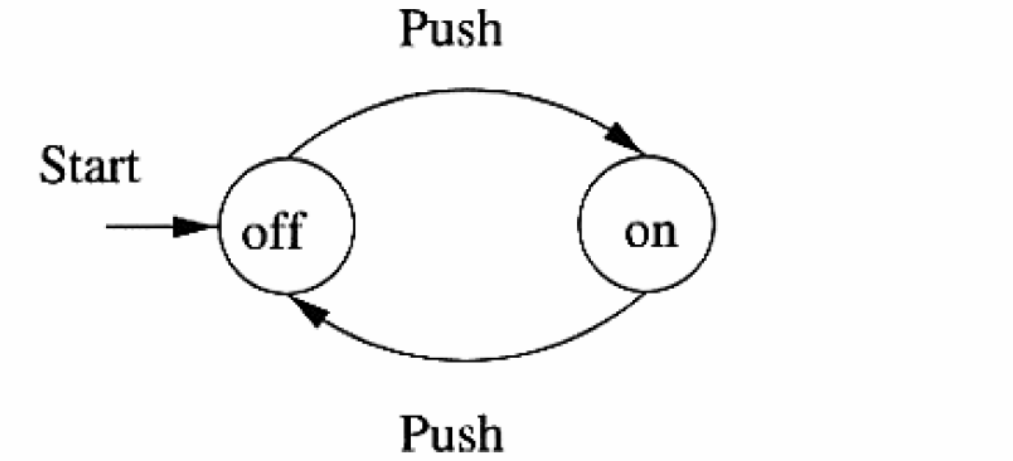
May 2012

*Version: 1.1, Last modified: May 10, 2012*

# Example

- Perhaps the simplest nontrivial finite automaton is an on/off switch.
- The device remembers whether it is in the “on” state or the “off” state.
- It allows the user to press a button whose effect is different, depending on the state of the switch.
- That is, if the switch is in the off state, then pressing the button changes it to the on state.
- And if the switch is in the on state, then pressing the same button turns it to the off state.

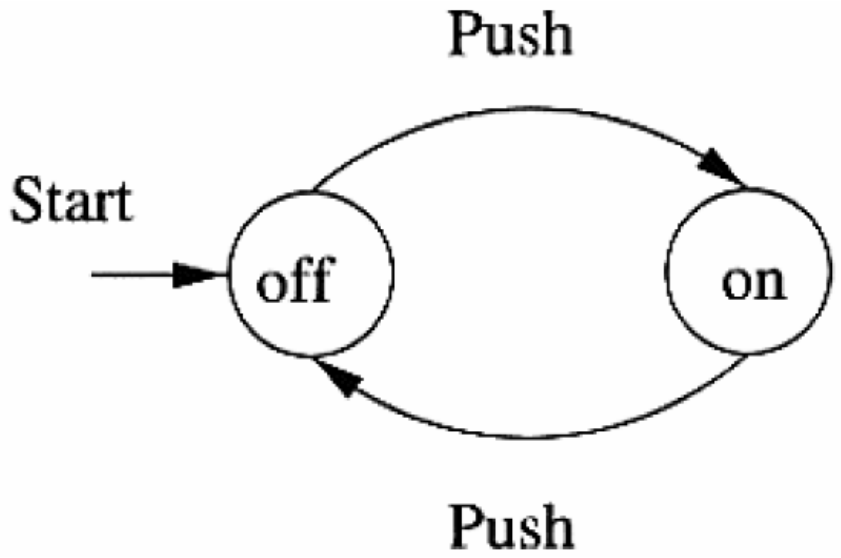
# Example — *continued*




A finite automaton modeling an on/off switch

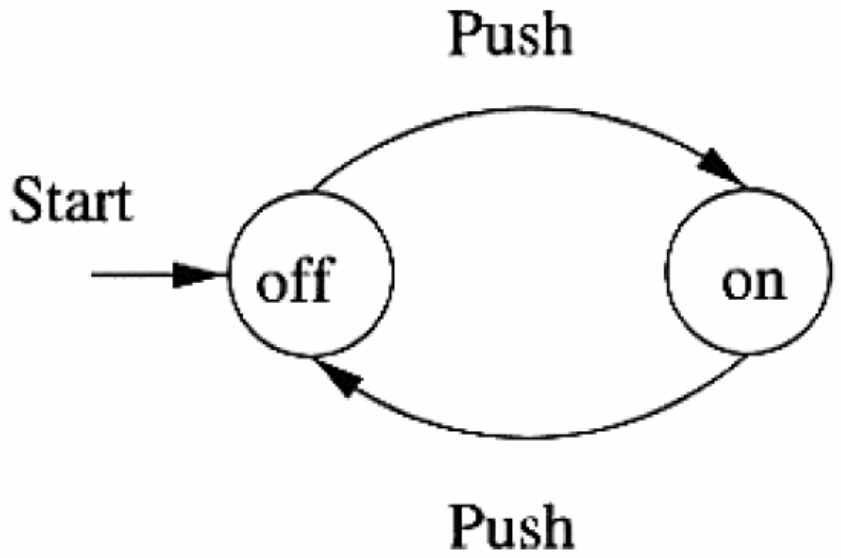
- The finite-automaton model for the switch is shown.
- As for all finite automata: the states are represented by circles.

# Example — *continued*



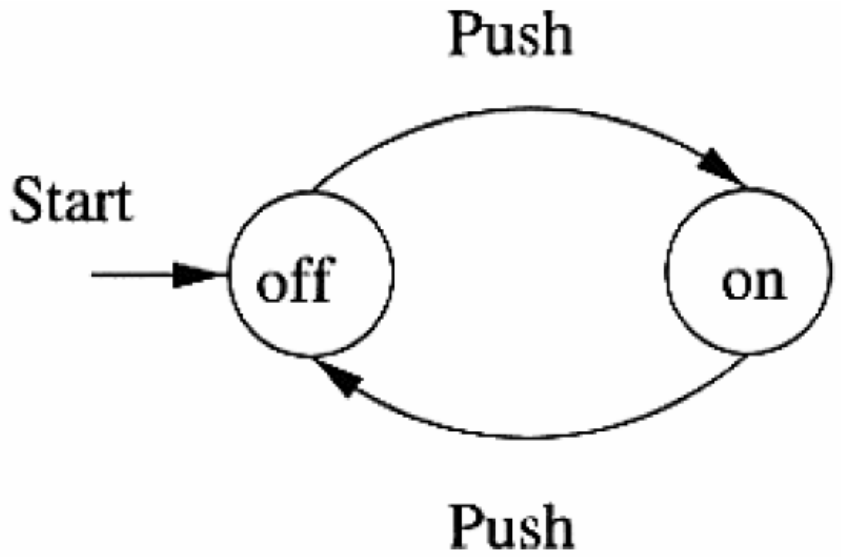

- In this example, we have named the states on and off.
- Arcs between states are labeled by “inputs,” which represent external influences on the system.
- Here, both arcs are labeled by the input Push, which represents a user pushing the button.
- The intent of the two arcs is that whichever state the system is in, when the Push input is received it goes to the other state.

# Example — *continued*



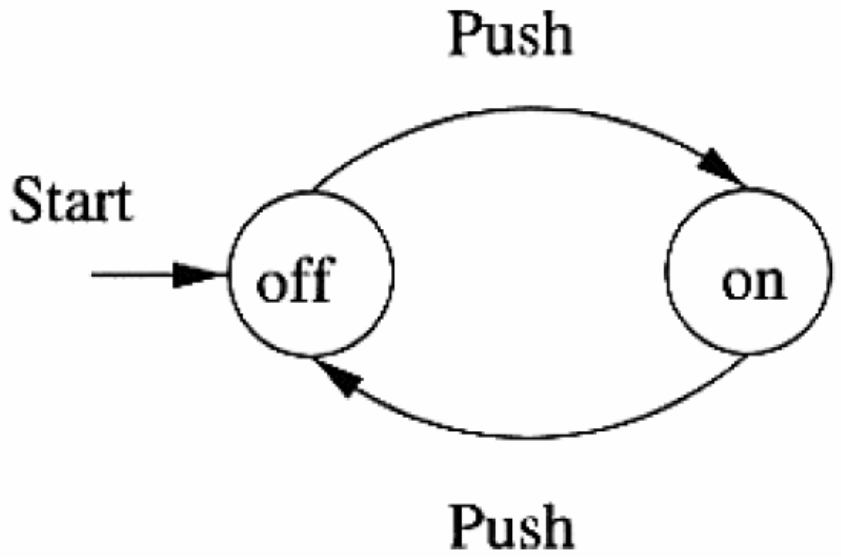

- One of the states is designated the “start state,” the state in which the system is placed initially.
- In our example, the start state is *off*.
- We conventionally indicate the start state by the word Start and an arrow leading to that state.

# Example — *continued*




- It is often necessary to indicate one or more states as “final” or “accepting” states.
- Entering one of these states after a sequence of inputs indicates that the input sequence is good in some way.

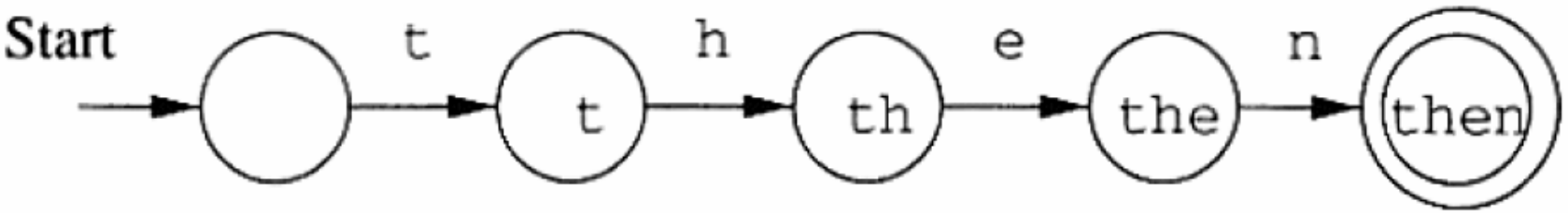
# Example — *continued*




- For instance, we could have regarded the state *on* as accepting.
- Because in that state: the device being controlled by the switch will operate.
- It is conventional to designate accepting states by a double circle, although we have not made any such designation in the figure.

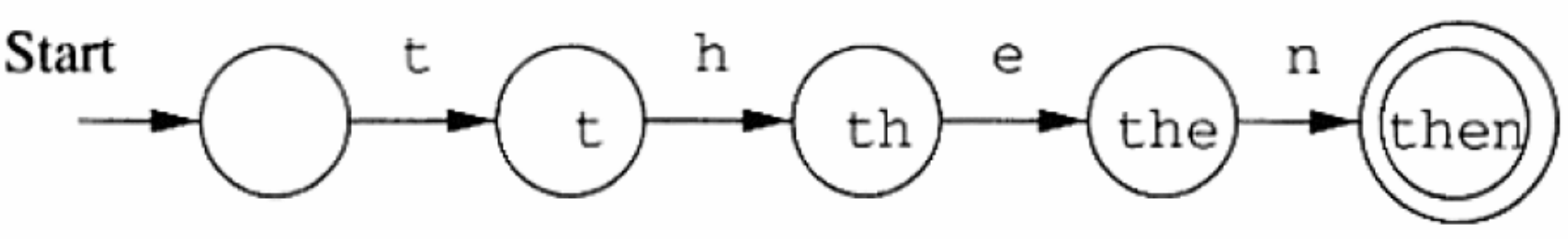
# Example

- Sometimes, what is remembered by a state can be much more complex than an on/off choice.
- Figure shows another finite automaton that could be part of a lexical analyzer.



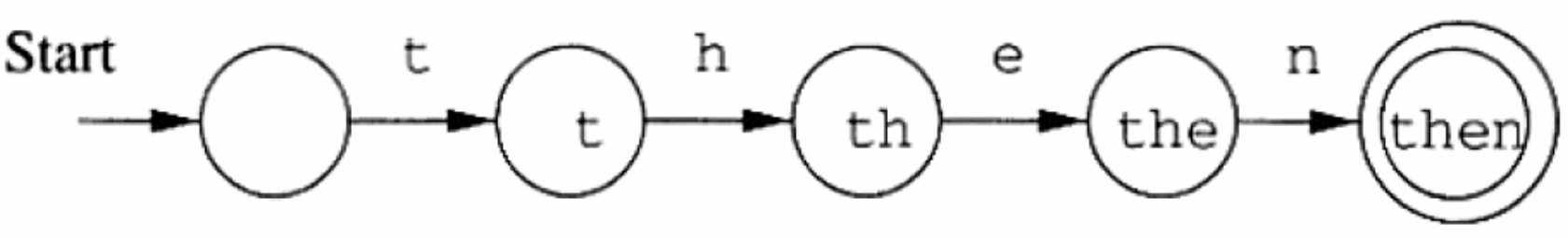

A finite automaton modeling recognition of **then**

# Example — *continued*



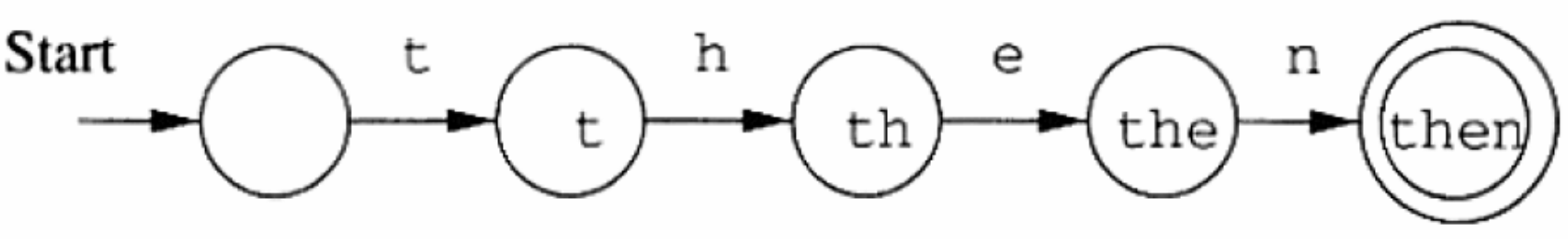

- The job of this automaton is to recognize the keyword `then`.
- It thus needs five states, each of which represents a different position in the word `then` that has been reached so far.
- These positions correspond to the prefixes of the word, ranging from the empty string (i.e., nothing of the word has been seen so far) to the complete word.

# Example — *continued*




- The five states are named by the prefix of then seen so far.
- Inputs correspond to letters.
- We may imagine that the lexical analyzer examines one character of the program that it is compiling at a time.
- The next character to be examined is the input to the automaton.

# Example — *continued*




- The start state corresponds to the empty string.
- Each state has a transition on the next letter of then to the state that corresponds to the next-larger prefix.
- The state named then is entered when the input has spelled the word `then`.
- Since it is the job of this automaton to recognize when `then` has been seen, we could consider that state the lone accepting state.

# End of Slides