Finite State Machine

A finite state machine (FSM), also known as a finite state automaton (FSA), is a mathematical model of computation used to design both computer programs and sequential logic circuits. It is a behavior model composed of a finite number of states, transitions between those states, and actions.

FSMs are widely used in various fields, including computer science, mathematics, logic, linguistics, and engineering.

Key components of a finite state machine

1. States: A finite set of states that the machine can be in at any given time. The machine can only be in one state at a time.
2. Transitions: A set of rules that define how the machine moves from one state to another based on the input it receives.
3. Input: A set of symbols or events that trigger state transitions.
4. Output: A set of actions or symbols that the machine produces based on the current state and input.
5. Initial state: The state in which the machine starts operating.
6. Final states (optional): A set of states that indicate the completion of processing or acceptance of input.

How the FSM works

A finite state machine works by transitioning between a finite number of predefined states based on the input it receives. The machine starts in an initial state and then moves to other states as it processes the input. The behavior of the machine is determined by the current state and the input symbol.

Here's a step-by-step explanation of how a finite state machine works:

1. Define the states: Identify the possible states that the machine can be in. Each state represents a specific condition or mode of operation.
2. Define the input symbols: Determine the set of input symbols that the machine will accept. These symbols trigger the transitions between states.
3. Define the transitions: Create a transition function that specifies how the machine moves from one state to another based on the current state and the input symbol. The transition function is typically represented as a table or a diagram.
4. Set the initial state: Specify the state in which the machine starts operating.
5. Process the input: Feed the input symbols to the machine one at a time. For each input symbol:  
   a. Determine the current state of the machine.  
   b. Look up the transition function to find the next state based on the current state and the input symbol.  
   c. Move the machine to the next state.  
   d. If the machine produces output, generate the output based on the current state and/or the input symbol (depending on whether it's a Mealy or Moore machine).
6. Repeat step 5 until all input symbols have been processed or until the machine reaches a final state (if defined).
7. Determine the output: The output of the machine depends on the type of FSM:
   * For a Mealy machine, the output is determined by the current state and the input symbol.
   * For a Moore machine, the output is determined solely by the current state.

Here's a simple example to illustrate the process:

Suppose we have a finite state machine that accepts binary strings ending with "00". The machine has three states: S0 (initial state), S1, and S2 (final state). The input symbols are 0 and 1.

The transition function is defined as follows:

* If the current state is S0 and the input is 0, move to state S1.
* If the current state is S0 and the input is 1, stay in state S0.
* If the current state is S1 and the input is 0, move to state S2.
* If the current state is S1 and the input is 1, move to state S0.
* If the current state is S2 and the input is 0 or 1, stay in state S2.

Now, let's process the input string "1010100":

1. Start in state S0.
2. Input: 1, stay in state S0.
3. Input: 0, move to state S1.
4. Input: 1, move to state S0.
5. Input: 0, move to state S1.
6. Input: 1, move to state S0.
7. Input: 0, move to state S1.
8. Input: 0, move to state S2 (final state).

Convention symbols used to explain FST

When explaining or illustrating finite state machines (FSMs), several conventional symbols are commonly used to represent the different components and behaviors of the machine. These symbols help in creating clear and concise diagrams or tables that describe the FSM. Here are some of the most common symbols used:

1. States:
   * Circles or ovals: Used to represent the states of the FSM.
   * Double circles or ovals: Used to represent the final or accepting states.
2. Transitions:
   * Arrows: Used to indicate the transitions between states.
   * Labels on arrows: Used to specify the input symbol and/or output symbol associated with each transition.
3. Initial state:
   * Arrow pointing to a state: Used to indicate the initial state of the FSM.   
     The initial state is indicated by an arrow with a block dot
4. Input symbols:
   * Lowercase letters or numbers: Used to represent the input symbols that the FSM accepts.
5. Output symbols:
   * Lowercase letters, numbers, or actions: Used to represent the output symbols or actions that the FSM produces.
6. Transition table:
   * Table with rows and columns: Used to represent the transition function of the FSM.
   * Rows: Represent the current state.
   * Columns: Represent the input symbols.
   * Cells: Contain the next state and/or output symbol for each combination of current state and input symbol.

