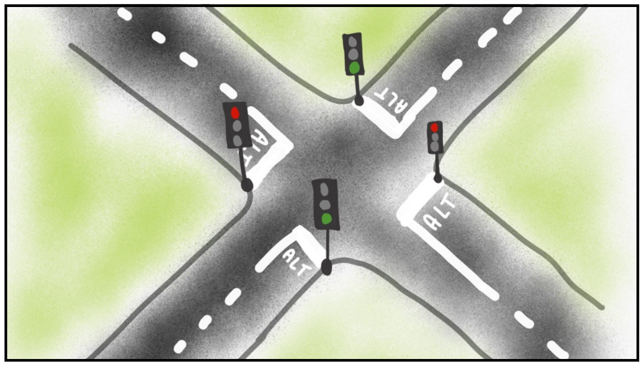
Finite-State Machines

Dr. Mark Anthony A. Ozaeta, MBA

1



2

Finite-State Machine

- A **finite-state machine** (**FSM**) or **finite-state automaton** (**FSA**, plural: *automata*), **finite automaton**, or simply a **state machine**, is a mathematical model of computation. It is an abstract machine that can be in exactly one of a finite number of *states* at any given time.

3

Finite-State Machine

- The FSM **can change from one state to another** in response to some inputs; the change from one state to another is called a *transition*.

4

Finite-State Machine

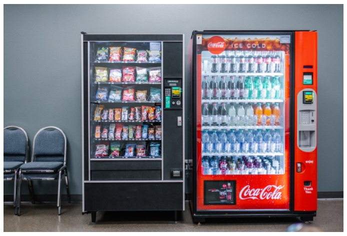
- FSM is defined by a list of its states, its initial state, and the inputs that trigger each transition. Finite-state machines are of two types – deterministic finite-state machines and non-deterministic finite-state machines. A deterministic finite-state machine can be constructed equivalent to any non-deterministic one.

5

Real-world Examples

6

Real-world Examples



7

Real-world Examples

8

Real-world Examples

9

Real-world Examples

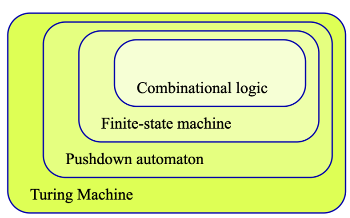
- Simple examples are vending machines, which dispense products when the proper combination of coins is deposited.
- Elevators, whose sequence of stops is determined by the floors requested by riders.
- Traffic lights, which change sequence when cars are waiting.
- Combination locks, which require the input of a sequence of numbers in the proper order.

10

How powerful is Finite State Machine?

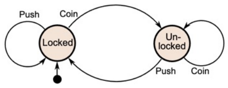
- The finite-state machine **has less computational power than some other models of computation** such as the Turing machine.
- The computational power distinction means there are computational tasks that a Turing machine can do but a FSM cannot.
- This is because a FSM's memory is limited by the number of states it has. FSMs are studied in the more general field of automata theory.

11



Classes of automata

12



State diagram for a turnstile

A turnstile

A turnstile, used to control access to subways and amusement park rides, is a gate with three rotating arms at waist height, one across the entryway.

Initially the arms are locked, blocking the entry, preventing patrons from passing through.

Depositing a coin or token in a slot on the turnstile unlocks the arms, allowing a single customer to push through.

After the customer passes through, the arms are locked again until another coin is inserted.

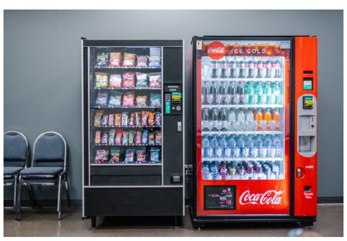
13

The turnstile state machine can be represented by a state-transition table, showing for each possible state, the transitions between them (based upon the inputs given to the machine) and the outputs resulting from each input:

Current State	Input	Next State	Output
Locked	coin	Unlocked	Unlocks the turnstile so that the customer can push through.
	push	Locked	None
Unlocked	coin	Unlocked	None
	push	Locked	When the customer has pushed through, locks the turnstile.

14

Real-world Examples



15

Vending Machine State Table

Assume the vending machine sells items for ₱20 and accepts ₱10 coins.

Current State	Input	Next State	Output
Idle	₱10 inserted	₱10 Collected	Display: "₱10 more needed"
₱10 Collected	₱10 inserted	Dispense Item	Dispense item
Dispense Item	Done	Idle	Reset machine, display: "Insert ₱20"

16

Real-world Examples

17

Elevator State Table

Assume elevator serves 3 floors (1, 2, 3) and handles simple up/down commands.

Current State	Input	Next State	Output
Floor 1	Up	Floor 2	Move to Floor 2
Floor 2	Up	Floor 3	Move to Floor 3
Floor 2	Down	Floor 1	Move to Floor 1
Floor 3	Down	Floor 2	Move to Floor 2
Floor 1-3	Stop	Same Floor	Open door, wait for passengers

18

Exercise (Get ½ Crosswise)

Scenario A: Coffee Vending Machine

The machine sells one cup of coffee for ₱15. It accepts only ₱5 coins.

Requirements:

- If a customer inserts less than ₱15, prompt them to insert more.
- After receiving ₱15, dispense coffee and return to idle.
- Do not accept more than ₱15.

19

Concepts and Terminologies

A *state* is a description of the status of a system that is waiting to execute a *transition*.

A transition is a set of actions to be executed when a condition is fulfilled or when an event is received.

For example, when using an audio system to listen to the radio (the system is in the "radio" state), receiving a "next" stimulus results in moving to the next station. When the system is in the "CD" state, the "next" stimulus results in moving to the next track. Identical stimuli trigger different actions depending on the current state.

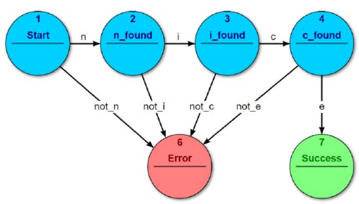
In some finite-state machine representations, it is also possible to associate actions with a state:

- an entry action: performed *when entering* the state, and
- an exit action: performed *when exiting* the state.

20

Classifications: Acceptors

- **Acceptors** (also called **detectors** or **recognizers**) produce binary output, indicating whether or not the received input is accepted.
- Each state of an acceptor is either *accepting* or *n on accepting*.



21

Classifications: Classifiers

- **Classifiers** are a generalization of acceptors that produce *n*-ary output where *n* is strictly greater than two.

The machine classifies inputs into **multiple categories** (e.g., Class 1, Class 2, ..., Class n)

- Instead of a binary decision, it makes a **multi-class decision**

Think of it as:

- **Acceptor**: "Does this string belong to the language?" → YES/NO
- **Classifier**: "To which of the n languages (or categories) does this string belong?" → Class 1, Class 2, ..., Class n

22

Classification: Transducers

- A **transducer** is a type of FSM that, **unlike acceptors or classifiers**, produces **output strings** (not just decisions or class labels) in response to input strings. It essentially **maps inputs to outputs**, making it a **translator or converter**.

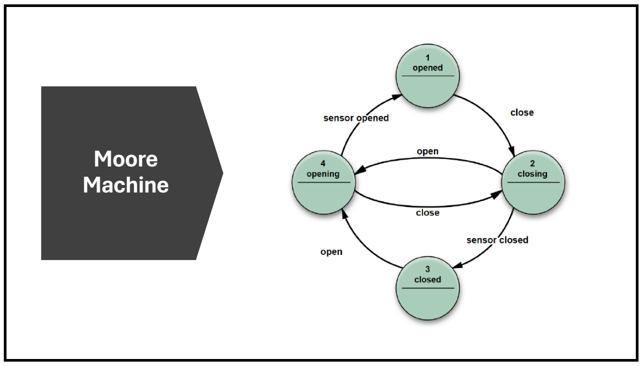
23

Classification: Transducers

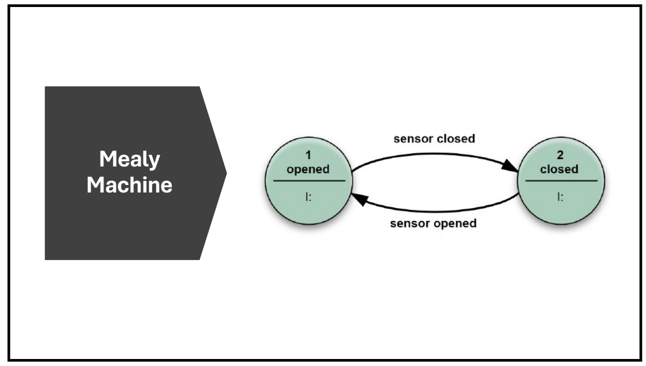
There are two main types:

- **Moore Machine**: Output depends **only on the current state**
- **Mealy Machine**: Output depends on **both current state and input symbol**

24



25



26

Use-case Example

FSM Variant	Input	Output	Output Form	Function
Acceptor	String	Accept / Reject	Boolean decision	Language recognition
Classifier	String	Class label (n > 2)	Symbol from finite set	Categorization
Transducer	String	Output string (sequence)	New string/symbol stream	Transformation / translation

27