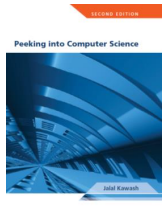
4 Finite State Machines

Peeking into Computer Science



- Mandatory: Chapter 3 – Section 3.6

Reading Assignment

Finite State Machines

Graphs as Solutions

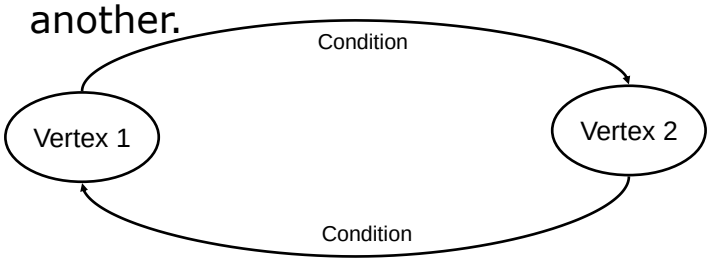
3

At the end of this section, the student will be able to:

1. Define Finite State Machines (FSMs)
2. Represent FSMs using state-tables and state-diagrams
3. Use FSMs to design High-Level programs
4. Understand FSM examples

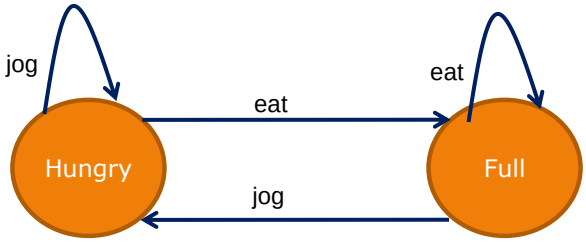
Objectives

- It's a special form of multi graph (vertices and edges) that have conditions that show how you go from one vertex to another.



JT's Extra: What Is A Finite State Machine?

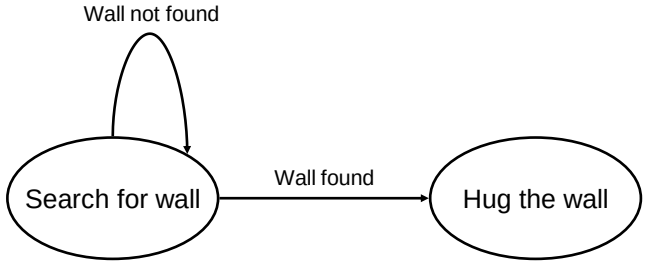
- Vertices are **States**
- Edges are labeled **Transitions**



Finite State Machines

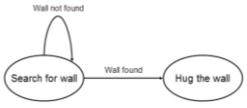
- State machine: illustrating a treasure hunt.

JT's Extra: Video



JT' Extra: First Example: Robot Redux (State Machine)

- A state machine is another level of abstraction that specifies what a computer is supposed to do.
  - Level I: Finite state machine



  - Level II: Human language instructions (pseudo code)

Search for wall
If RS = W, then done this phase
If FS = W, then L, done this phase
If FS = S, then F

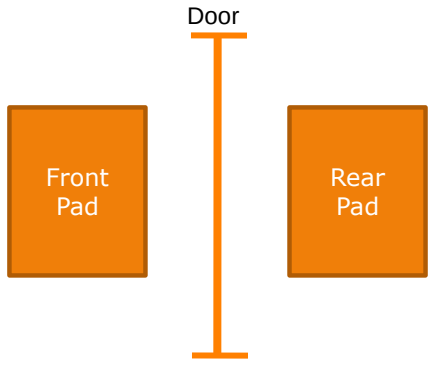
Hug the wall
Repeat the following steps:
If RS = W and FS = S, then F
If FS = W, then L
If RS = S and FS = S, then R and F

JT's Extra: Why Learn About State Machines

  - Level III: Programming language instructions

```
if (robot.wallToRight () == true)
{
   isDone = true;
   return;
}
```

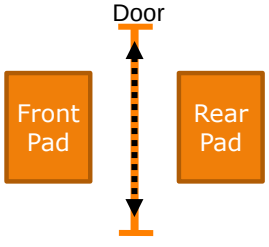
JT's Extra: Why Learn About State Machines



Based on Sipser, *Introduction to the Theory of Computation*, Thomson

Automatic Door Controller

- Identify the door states
- Identify events that trigger transitions
- Version 1 : sliding door



Automatic Door Controller

- Identify the door states
- Identify events that trigger transitions

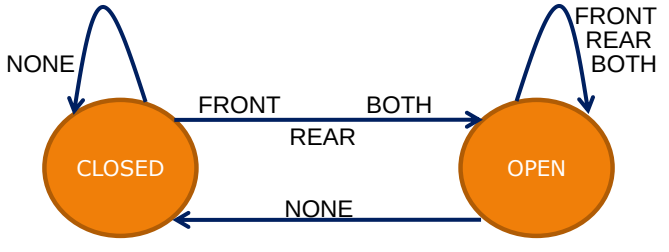
Events

Door States	NONE	FRONT	REAR	BOTH
CLOSED	CLOSED	OPEN	OPEN	OPEN
OPEN	CLOSED	OPEN	OPEN	OPEN

FSM – Sliding Door

Events

Door States	NONE	FRONT	REAR	BOTH
CLOSED	CLOSED	OPEN	OPEN	OPEN
OPEN	CLOSED	OPEN	OPEN	OPEN



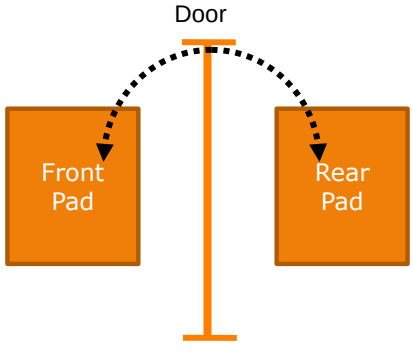
FSM Diagram – Sliding Door

	NONE	FRONT	REAR	BOTH
CLOSED	CLOSED	OPEN	OPEN	OPEN
OPEN	CLOSED	OPEN	OPEN	OPEN

Input	Current State	Next State
NONE	CLOSED	CLOSED
NONE	OPEN	CLOSED
FRONT	CLOSED	OPEN
FRONT	OPEN	OPEN
REAR	CLOSED	OPEN
REAR	OPEN	OPEN
BOTH	CLOSED	OPEN
BOTH	OPEN	OPEN

Alternative Table

- Version 2 : door opens both ways



Door Open Both Ways

- Identify the door states
- Identify events that trigger transitions

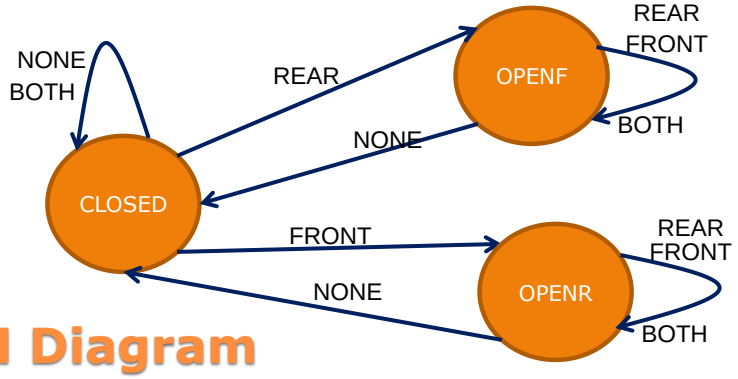
Events

Door States	NONE	FRONT	REAR	BOTH
CLOSED	CLOSED	OPENR	OPENF	CLOSED
OPENF	CLOSED	OPENF	OPENF	OPENF
OPENR	CLOSED	OPENR	OPENR	OPENR

FSM – Door Opens Both Ways

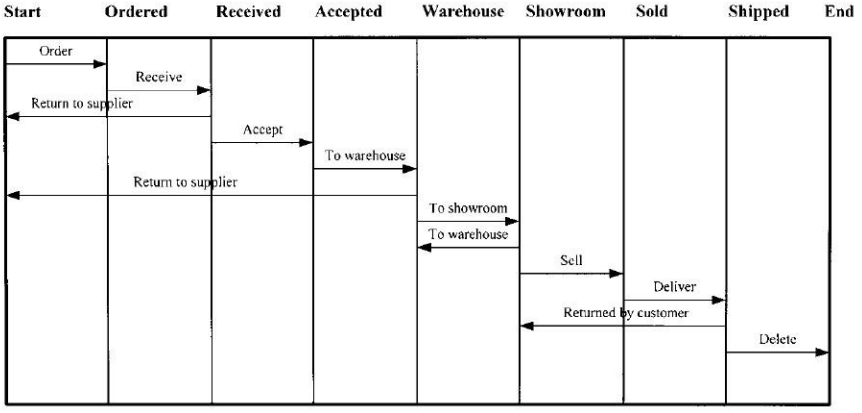
Events

Door States	NONE	FRONT	REAR	BOTH
CLOSED	CLOSED	OPENR	OPENF	CLOSED
OPENF	CLOSED	OPENF	OPENF	OPENF
OPENR	CLOSED	OPENR	OPENR	OPENR



FSM Diagram

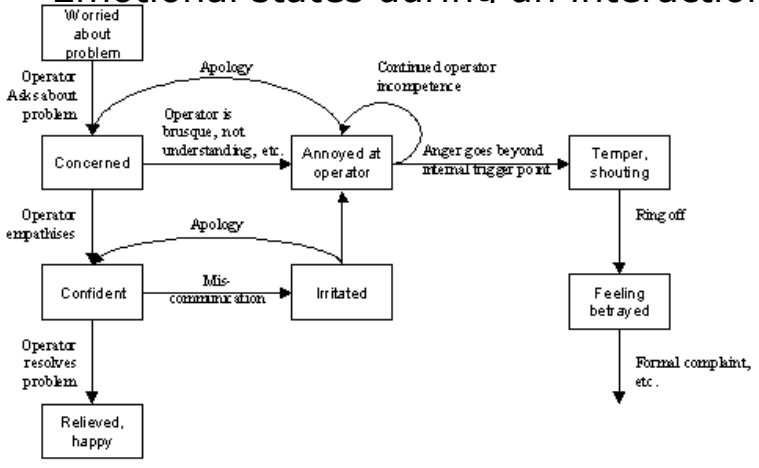
- Control flows or steps in a process[1]



JT's Extra, Real Life Example: State Machine

1 From www.hit.ac.il

- Emotional states during an interaction[1]



JT's Extra: State Machines Can Model More Than Just Electronics/Machinery

1From http://syque.com/articles/state-transition/state-transition.htm

- There are two colors in the game of chess: white and black. White begins the game and thereafter the game alternates between white and black until either player wins (checkmate) or the game is a draw (stalemate). A win or stalemate can occur during either player's turn.

JT's Extra 1, A Chess Game

- Design a controller for a garage door
- The door receives input from one remote control only
- It also responds to sensing obstacles

Exercise

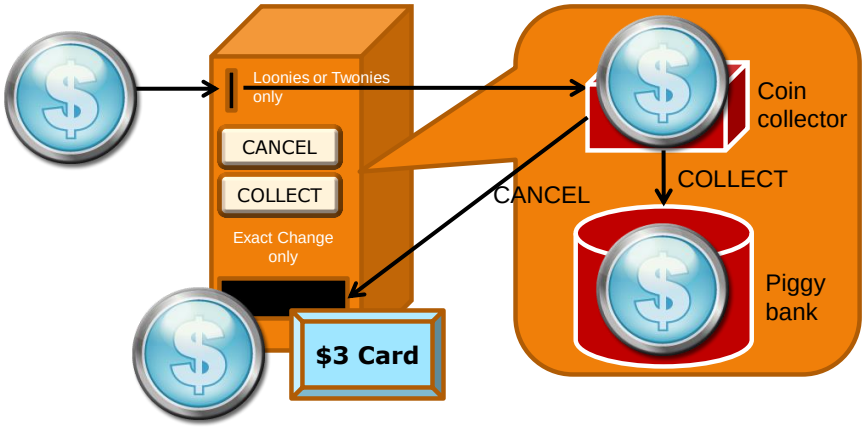
- Consider a faucet with two taps: hot and cold. For simplicity consider that there's four possibilities. If both taps are off then no water comes out. If the hot tap is on then only hot water comes out. If the cold tap is on then only cold water comes out. If the hot and cold tap are on then warm water comes out.

JT's Extra 2, Water Taps

- Dispenses $3 phone cards
- Accepts $1 or $2 only
- No change
- Keeps coins in *coin collector* until a card is dispensed; then, coins are dropped into the piggy bank

- User
  - Insert coins
  - Press COLLECT to collect
  - Press CANCEL to cancel

Simple Vending Machine



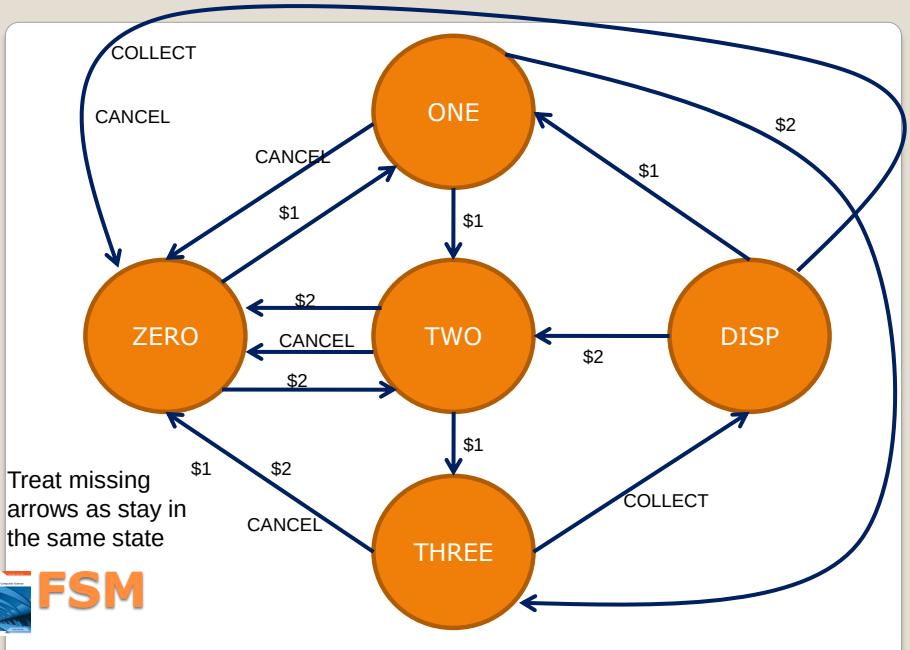
That is

- Need five states:
- ONE: Total in coin collector is $1
- TWO: Total in coin collector is $2
- THREE: Total in coin collector is $3
- DISP: dispenses a card, roll in coins to piggy (coin collector becomes empty)
- ZERO: return coins in coin collector; also serves as a start state
  - Initially coin collector is empty

Vending Machine – States

- Events/Transition labels:
- $1: user inserts a loonie
- $2: user inserts twonie
- CANCEL: user presses CANCEL
- COLLECT: user presses COLLECT

Vending Machine – Events



Treat missing arrows as stay in the same state

FSM

- State-transition diagram
- Finite state automata
- Finite state machines

JT's Extra: For More Information

- Personal electronics e.g. digital watch, MP3 player, regular (non-smart) phones.
- Home electronics e.g. PVR, DVD, TV, VCR, clock radio (some PVR's and VCR's may be too complex).
- Simple machines e.g., vending machines, ticket dispenser, car dashboard controls, ATM (some could be too complex).
- Simple interactive websites (must allow for user interaction rather than ones that just statically display information e.g., Facebook or a banking site as opposed to a news site that doesn't allow for user comments).
- Software that is modal e.g., drawing and painting programs

JT's Extra: Other Sources Of Practice

