CS310

Finite Automata

Sections:

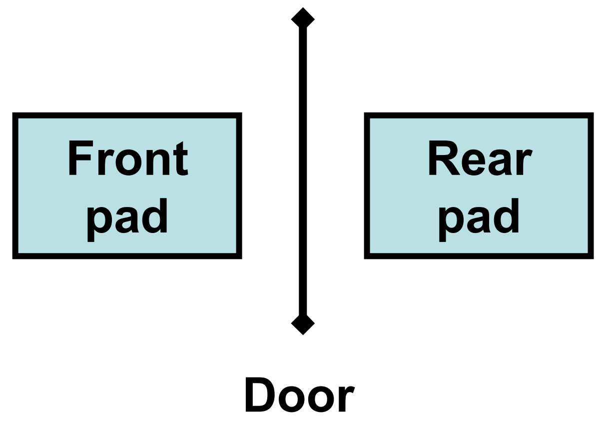
September 1, 2006

Quick Review

- Alphabet: $\Sigma = \{a,b\}$
  - Σ^*: Closure:
- String: any finite sequence of symbols from a given alphabet. $|w|$ = length
  - Concatenation/Prefix/Suffix/Reverse
- Language L over Σ is a subset of Σ^*
  - $L = \{ x \mid \text{rule about } x\}$
  - Concatenation/Union/Kleene Star
  - Recursive Definition

Finite Automata

- Computer Science is really the science of computation, not of computers.
- How can we reason about computation?
- Simple model of computation
  - Finite Automata
  - extremely limited amount of memory
  - represent *states* of computation



Door State: Open, Closed

Inputs: Front, Rear, Both, Neither

State Transition Table

Input / State	Neither	Front	Rear	Both
Open				
Closed				

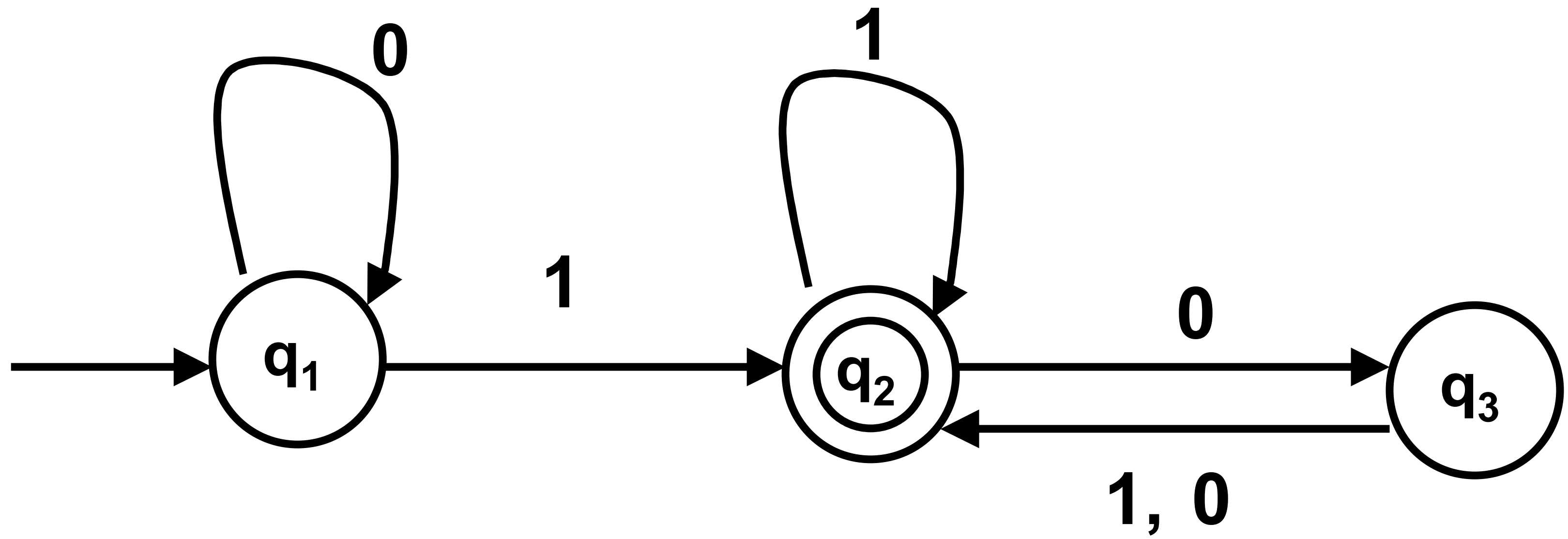
State Diagram

More uses…

- Recognize patterns in data
- Build an automata that can classify a string as part of a language or not

Language:

L = { x є {0,1}* | x contains at least one 1 and the last 1 is followed by even number of 0s}



Inputs: Accept or Reject?

1101	**1110**	**0**
1100	**1**	**11**

Set of all strings (A) accepted by a machine (M) is the *Language of the Machine*
M *recognizes* A or M *accepts* A

Formal Definition

- Deterministic Finite Automata:

  5-tuple $(Q, \Sigma, \delta, q_0, F)$

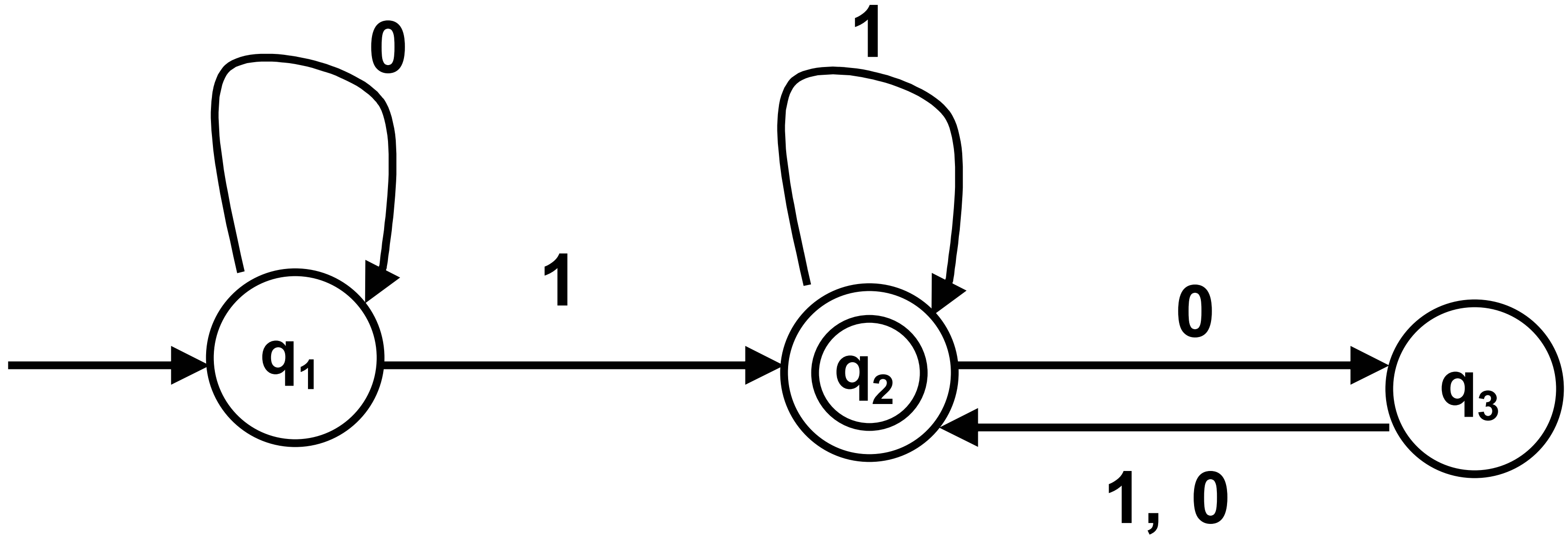
  Q: finite set of states

  Σ: alphabet (finite set)

  δ : transition function ($\delta: Q \times \Sigma \rightarrow Q$)

  q_0: start state

  F : accepting states (subset of Q)



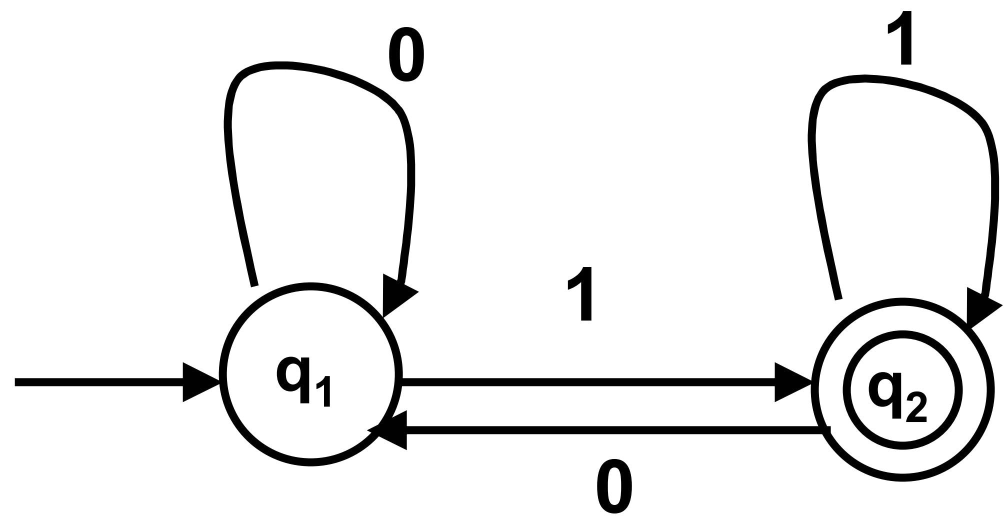
Q: finite set of states

Σ: alphabet

δ : transition function

q_0: start state

F : accepting states



Q:

∑:

δ :

q_0:

F :

What strings get accepted?

L(M) = {   }

Designing a DFA

- Identify small pieces
  - alphabet, each state needs a transition for each symbol
  - finite memory, what crucial data does the machine look for?
  - can things get hopeless? do we need a trap?
  - where should the empty string be?
  - what is the transition into the accept state?
  - can you transition out of the accept state?
- Practice!

$L(M) = \{ w \mid w = \varepsilon \text{ or } w \text{ ends in } 1\}$

$\Sigma = \{ 0,1\}$

Q:

δ :

q_0:

F :

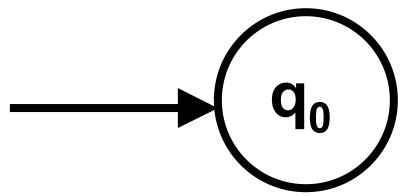
- $\Sigma = \{0,1\}$, $L(M)=\{w \mid \text{odd \# of 1s}\}$

Build a DFA to do math!

L(M) = Accept sums that are multiples of 3

$\sum = \{ 0,1,2, \text{<Reset>}\}$

Keep a running total of input, modulo 3



- $\sum = \{0,1\}$, $L(M)=\{w \mid \text{begins with 1, ends with 0}\}$

- $\Sigma = \{0,1\}$, $L(M)=\{w \mid \text{contains } 110\}$

- $\sum = \{0,1\}$, $L(M)=\{w \mid \text{does not contain } 110\}$

- $\Sigma = \{0,1\}$, $L(M)=\{w \mid (01)^* \}$

- $\Sigma = \{0,1\}$, $L(M)=\{w \mid w$ even #0s, odd #1s $\}$

- $\Sigma = \{0,1\}$, $L(M)=\{w \mid w$ any string except 11 and 111 $\}$

Formal Definition of Computing

- Given a machine $M = (Q, \sum, \delta, q_0, F)$ and a string $w = w_1 w_2 \ldots w_n$ over $\sum$, then M ***accepts*** w if there exists a sequence of states $r_0, r_1 \ldots r_n$ in Q such that:
  - $r_0 = q_0$: r_0 is the start state
  - $\delta(r_i, w_{i+1}) = r_{i+1}$, $i = 0, \ldots, n-1$: legal transitions
  - $r_n \in F$: stop in an accept state
- M ***recognizes*** A if $A = \{w \mid M \text{ accepts } w\}$
- Language A is ***regular*** if there exists a Finite Automata that recognizes A.