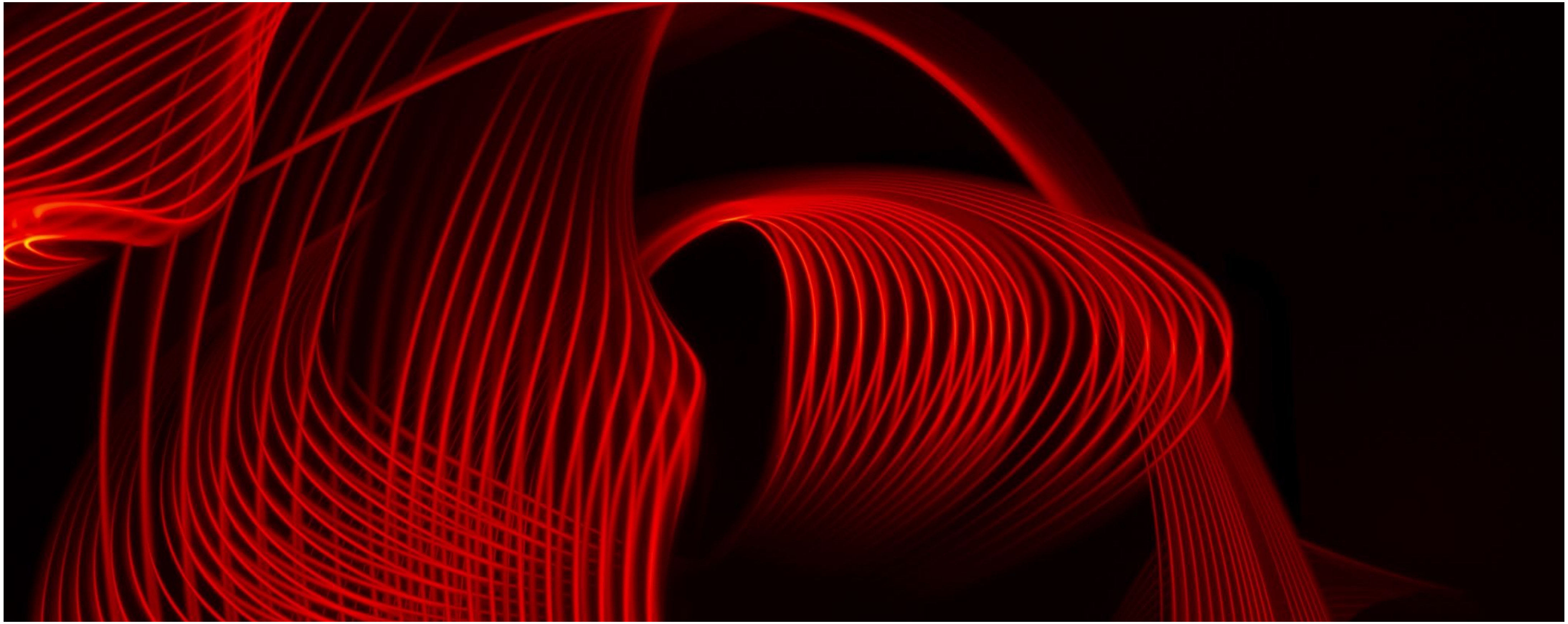
Finite Automata



First off, Reminder on TA Office Hours

- James Halvorsen (james.halvorsen@wsu.edu)
  - Fri 3:30--5pm
- Funso Oje (olufunso.oje@wsu.edu)
  - Tue 2:30--4pm
- Nathan Waltz (nathan.waltz@wsu.edu)
  - Mon 3--4:30pm
- Zoom links are set and are available via Canvas

What is a computer?

We use a **computational model** (an idealized computer) to explore this

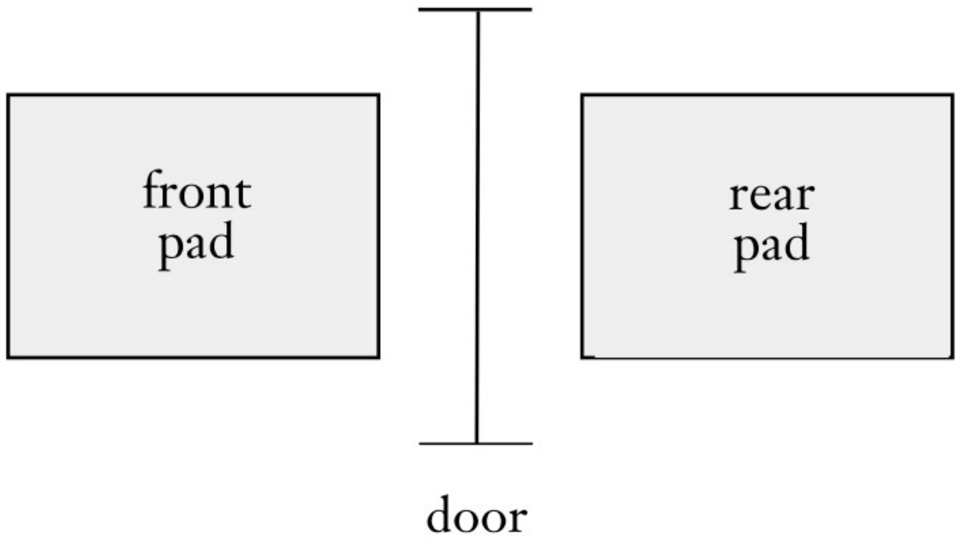
Like any model, computational models abstract away many things, but are still useful in understanding fundamental things

Finite state machine, or **finite automaton**, is the simplest computational model we will study

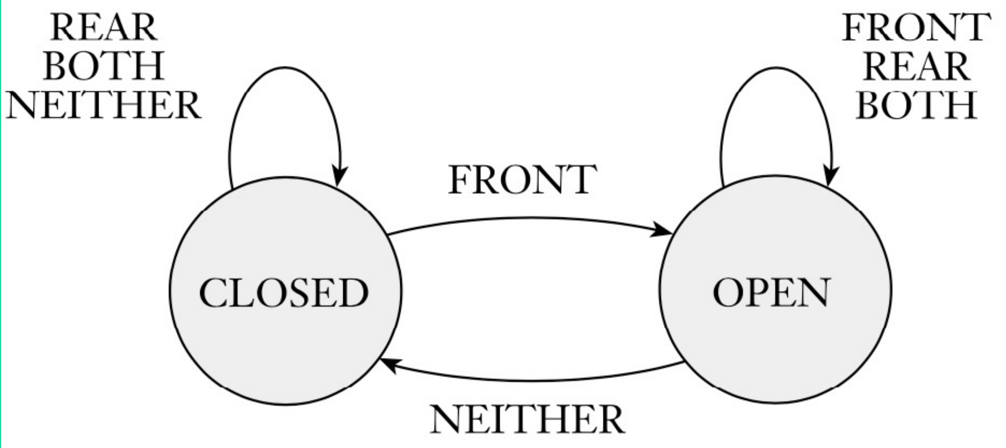
Finite Automata

- Good models for computers with an *extremely limited amount of memory*

- Despite what may come to mind first, such a computer can do many useful things – as can be in seen in the workings of many electromechanical devices
  - **Example**: controller of an automatic door



Controller of automatic door



State diagram

State transition table

		input signal			
		NEITHER	FRONT	REAR	BOTH
state	CLOSED	CLOSED	OPEN	CLOSED	CLOSED
	OPEN	CLOSED	OPEN	OPEN	OPEN

More examples of controller with limited memory

- *Elevator controller*
- *Dishwasher controller*
- *Electronic thermostat*
- *Digital watches*

The design of such devices requires keeping the methodology and terminology of finite automata in mind

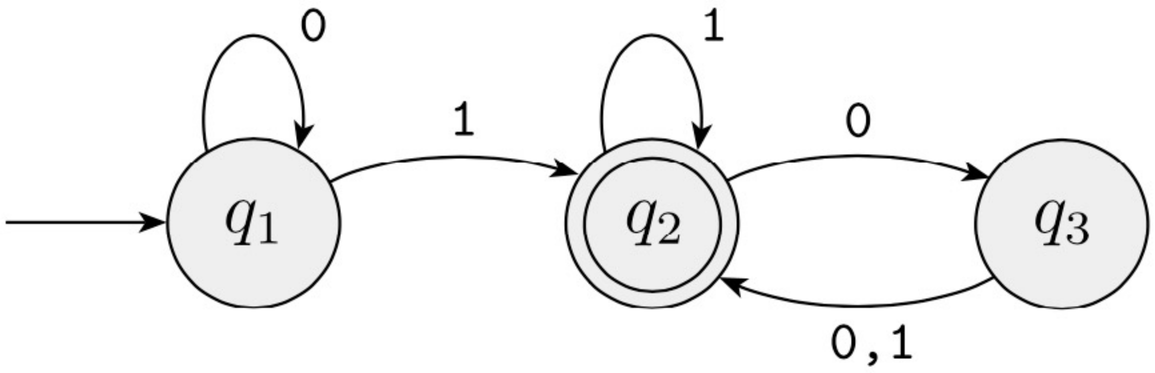
Other applications of finite automata

- **Finite Automata** and their cousin **Markov Chains** (probabilistic model) are useful tools in trying to ***recognize patterns in data***
  - *Speech processing*
  - *Optical character recognition*
  - *Price change prediction in markets* (Markov chains)
  - *Search engines* (PageRank uses Markov chains theory)

Closer look at Finite Automata – from mathematical perspective

- Precise definition of finite automata
- Terminology for describing and manipulating finite automaton
- Theoretical results that describe their power and limitations

An example – a FA called M_1



State diagram of M_1

- Has three states: q_1, q_2, q_3
- q_1 is the start state
- q_2 is the accept state
- Arrows show transitions

<u>Example</u>: what would M_1 do on input 1101?

What would M_1 accept more generally?

Formal definition of Finite Automaton

DEFINITION 1.5

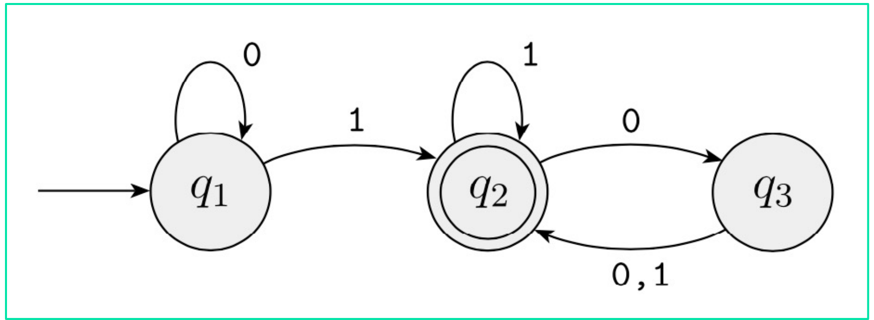
A ***finite automaton*** is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

1. Q is a finite set called the ***states***,
2. Σ is a finite set called the ***alphabet***,
3. $\delta : Q \times \Sigma \longrightarrow Q$ is the ***transition function***,[1]
4. $q_0 \in Q$ is the ***start state***, and
5. $F \subseteq Q$ is the ***set of accept states***.[2]

footnote

1: note the use of the Cartesian product
2: accept states are also called **final states**

Revisiting M_1



We can describe M_1 formally by writing $M_1 = (Q, \Sigma, \delta, q_1, F)$, where

1. $Q = \{q_1, q_2, q_3\}$,
2. $\Sigma = \{0,1\}$,
3. δ is described as

	0	1
q_1	q_1	q_2
q_2	q_3	q_2
q_3	q_2	q_2,

4. q_1 is the start state, and
5. $F = \{q_2\}$.

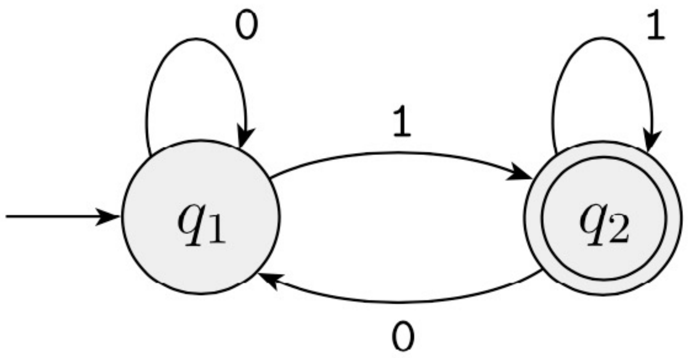
The language of a machine

- If A is the set of all strings that machine M accepts, we say that A is the language of machine M and write $L(M) = A$
- We say that M recognizes A or that M accepts A
- We prefer to use term recognizes for languages. A machine may accept several strings, but it always recognizes one language.
  - If the machine accepts no strings, it still recognizes one language – namely, the empty language $\emptyset$.
- In our example with M_1, let

$$A = \{w \mid w \text{ contains at least one 1 and an even number of 0s follow the last 1}\}.$$

Then $L(M_1) = A$, or equivalently, M_1 recognizes A.

Examples of finite automata – M_2



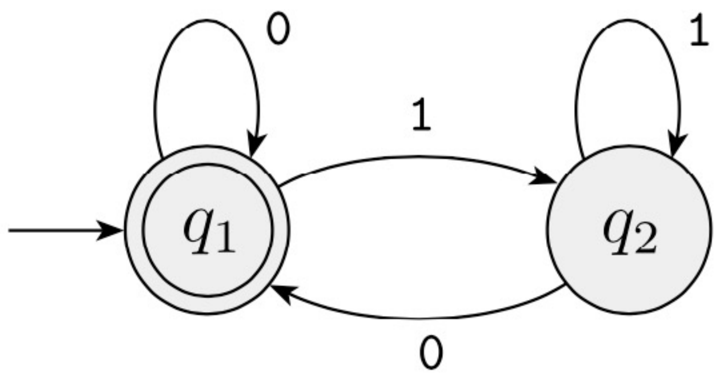
State diagram of M_2

In the formal description, M_2 is $(\{q_1, q_2\}, \{0,1\}, \delta, q_1, \{q_2\})$. The transition function δ is

	0	1
q_1	q_1	q_2
q_2	q_1	q_2.

Example: what would M_2 do on input string 1101?
How about on 110?

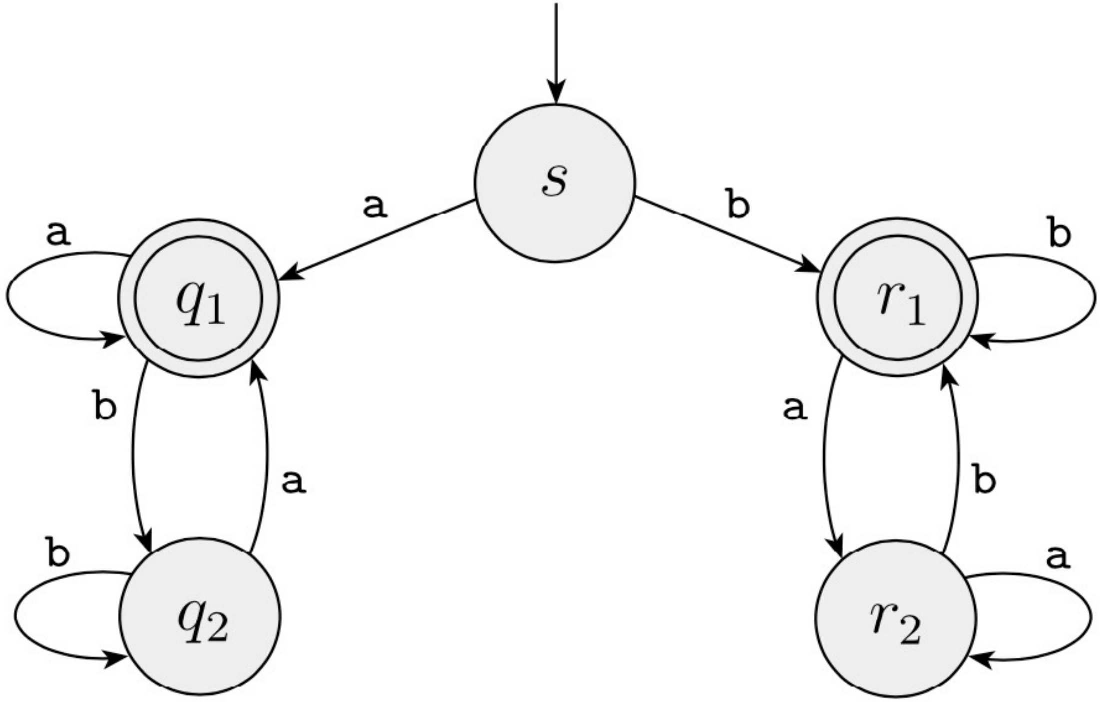
Examples of finite automata – M_3



- M_3 is similar to M_2 except for the location of the accept state
- Note that because the start state is also accept state, M_3 accepts the empty string ε

$$L(M_3) = \{w \mid w \text{ is the empty string } \varepsilon \text{ or ends in a } 0\}.$$

Examples of finite automata – M_4



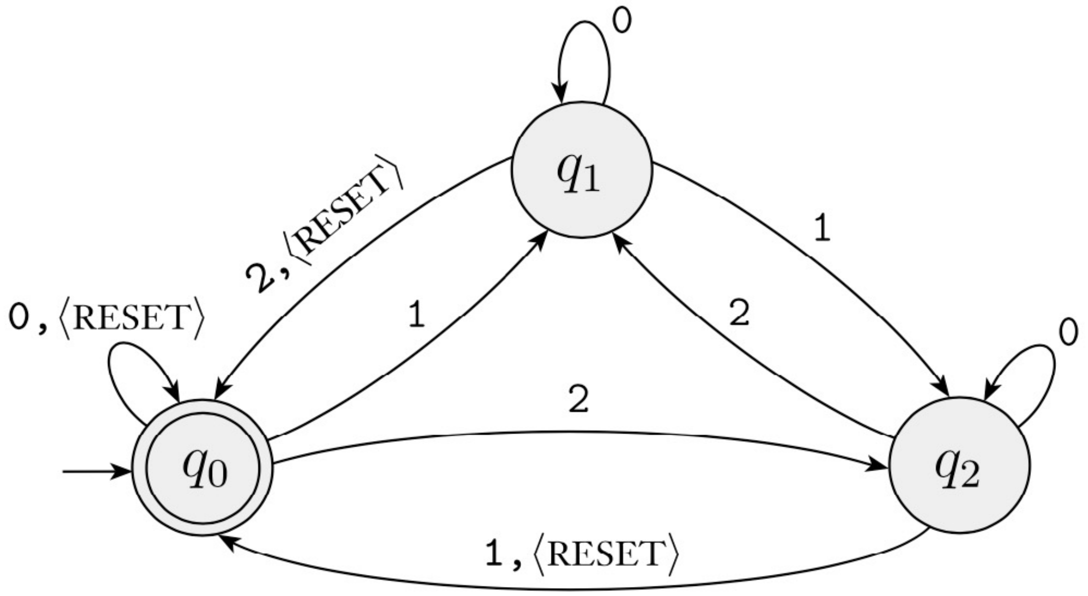
Question: what strings does M_4 accept?

Consider:

a, b, aa, bb, bab, ab, ba, bbba

Any idea about M_4 accepts in general?

Examples of finite automata – M_5



- M_5 has three states, and a four-symbol input alphabet Σ = {<RESET>, 0, 1, 2}.
- M_5 keeps a running count of the sum of the numerical input symbols it reads, modulo 3.
- Every time it receives the <RESET> symbol, it resets the count to 0.
- It accepts if the sum is 0 modulo 3 (IOW if the sum is a multiple of 3)

Formal definition of computation

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a finite automaton and let $w = w_1 w_2 \cdots w_n$ be a string where each w_i is a member of the alphabet Σ. Then M ***accepts*** w if a sequence of states $r_0, r_1, \ldots, r_n$ in Q exists with three conditions:

1. $r_0 = q_0$,
2. $\delta(r_i, w_{i+1}) = r_{i+1}$, for $i = 0, \ldots, n-1$, and
3. $r_n \in F$.

Condition 1 says that the machine starts in the start state. Condition 2 says that the machine goes from state to state according to the transition function. Condition 3 says that the machine accepts its input if it ends up in an accept state. We say that M ***recognizes language*** A if $A = \{w \mid M \text{ accepts } w\}$.

Regular languages

DEFINITION 1.16

A language is called a ***regular language*** if some finite automaton recognizes it.