• A finite automata (FA) is a collection of states, represented as circles.
• A finite automata (FA) is a collection of states, represented as circles.

• If we wanted to represent the FA mathematically, we would need to label each state and put the labels in a set $S = \{a, b, c\}$
• Every state **must** have an arrow coming from it for every character in the input alphabet.

• Mathematically, we need to indicate the input alphabet $A = \{0, 1\}$ and arrows. Since each arrow maps a (state, char) pair to a state, a function is a good representation $F: S \times A \rightarrow S$.

![Diagram of a finite automaton](image)

<table>
<thead>
<tr>
<th>$F$</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>a</td>
</tr>
<tr>
<td>b</td>
<td>c</td>
<td>b</td>
</tr>
<tr>
<td>c</td>
<td>a</td>
<td>c</td>
</tr>
</tbody>
</table>
• There **must** be exactly one start state indicated by an unlabeled arrow from nowhere. And zero or more accept states (also called final) indicated by double circles.

• Mathematically, we can call $S_i$ the start (initial) state, $S_i = a$, and let $Y = \{b, c\}$ be the set of accept states.
• Because you can represent a FA graphically or mathematically, these are identical FA.

\[ A = \{0, 1\} \]
\[ S = \{a, b, c\} \]
\[ S_i = a \]
\[ Y = \{b, c\} \]
\[ F \] shown below in table form

<table>
<thead>
<tr>
<th>( F )</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>a</td>
</tr>
<tr>
<td>b</td>
<td>c</td>
<td>b</td>
</tr>
<tr>
<td>c</td>
<td>a</td>
<td>c</td>
</tr>
</tbody>
</table>

• Note about state labels: They can be any name you wish and are optional in drawings.
FA Operation

• When presented an input:

  • Start at start state
  
  • Consume characters from left to right
  
  • Follow arrow for each character consumed
  
  • If end in accept state "accept", else "reject"
Example: Input is 10011. Start in the start state.
• Consume 1, follow arrow with 1, still in state "a"

Remaining after consume and move: 0011
• Consume 0, follow arrow with 0, now in state "b"

Remaining after consume and move: 011
• Consume 0, follow arrow with 0, now in state "c"

Remaining after consume and move: 11
• Consume 1, follow arrow with 1, still in state "c"

Remaining after consume and move: 1
• Consume 1, follow arrow with 1, still in state "c"

Remaining after consume and move: (empty)

• After consuming input, we end in an accept state.
  10011 is accepted by this FA.
  10011 is in the "language" of this machine.
Meaning of FA

• You can often design an FA to accept strings that are easily described in English.

• This FA accepts all strings over alphabet \{0, 1\} that don't have a multiple of three 0's.

• If this machine is called $M$, then $L(M) = \{ x | x \text{ is a string over alphabet } \{0,1\} \text{ and the number of 0's in } x \mod 3 = 0 \}$
Designing FA

- Give each state meaning. The only "memory" an FA has is the current state.
- Design the part of an FA that accepts good strings first.
- Make sure the FA is legal: one start state, arrow from each state for each character in the input alphabet.
- Test: try to find good string that's rejected; try to find bad string that's accepted. (This is how I grade FA's!)