
- Finite State Automata (FSA)

- The languages accepted by (defined by) these machines.
A FSA is a model of a very primitive computer:

- Input device that reads strings of characters (words)
- One memory location—possibly large
- Can output a yes/no determination
A FSA is deterministic. The future is determined by the current state and the next input character.

An FSA, $M$, has the following parts:

- $A$ — a set called the alphabet
- $S$ — a finite set — the set of states
- $f : S \times A \rightarrow S$, called the transition function.
- $q_0$ — an element of $S$ called the start state.
- $F$ — a subset of $S$, called the set of final states.
If $\delta(s, a) = t$ we intend that if the machine is in state $s$ when the character $a$ is input, then the machine moves to state $t$.

```cpp
    char c = nextChar();
    State = $\delta$(State, t);
```

Even for moderately-sized state sets, an FSA can describe a useful, albeit low-level, language.
Example: \[ A = \{a, b\}, \quad S = \{0, 1, 2\}; \]

\[ M: \]

\[
\begin{array}{c|ccc}
S & a & b & \\
\hline
0 & 1 & 0 & \\
1 & 2 & 1 & \\
2 & 2 & 1 & \\
\end{array}
\]

\[ q_0 = 0; \]

\[ F = \{2\} \]

On this input: \text{aba}

The machine follows these states: \((0, 1, 1, 2)\), so we say \(M\) accepts the word \text{aba}.

On this: \text{abb}

Follow this: \((0, 1, 1, 1)\)

\(1 \notin F\) of \(M\) rejects \text{abb}.
Example. \( S = \{0, 1\} \); \( A = \{0, 1\} \);

\[
N: \begin{array}{c|ccc}
\delta & 0 & 1 \\
\hline
0 & 0 & 1 \\
1 & 1 & 0 \\
\end{array} \quad q_0 = 0 \quad F = \{0\}.
\]

If \( N \) is in state \( q \):

- an input of 0 preserves the state;
- an input of 1 flips the state.

The words accepted by \( N \) are exactly those strings in \( 0 \& 1 \) with an even number of ones.

This example was easy because \( S \) is a semigroup. (FSAs \( \leftrightarrow \) semigroups)
Need a visual aid. (Modified digraphs.)

Draw $s$ to represent $s$ in $S$;

Use $\rightarrow$ (or whatever) to mark $q$;

Use $\bigcirc$ when $t \in F$;

Use $s \xrightarrow{a} t$ when $\delta(s, a) = t$. 
Back to First Example:

An 'a' moves as left, if possible.
A 'b' leaves us standing still.
Still on left, hope to end on right.

\[ M \text{ accepts } 01101, \text{ which have at two 'a's in them.} \]
2nd example:

\[ N : \]

\[ N \text{ is a parity checker.} \]

accepts odd number of ones

\[ (F = 0, 1, 3) \]

accepts all words in 0 and 1.