Deterministic and Nondeterministic Finite Automata

Last Time
- Sets Theory (Review?)
- Logic, Proofs (Review?)
- Words, and operations on them: $w_1 \circ w_2, w^i, w^*, w^+$
- Languages, and operations on them: $L_1 \circ L_2, L^i, L^*, L^+$

Today
- Deterministic Finite Automata (DFAs) and their languages
- Closure properties of DFA languages (the product construction)
- Nondeterministic Finite Automata (NFAs) and their languages
- Relating DFAs and NFAs (the subset construction)
Fibonacci as a Recursively Defined Set

The $n^{th}$ Fibonacci number $f(n)$:

$$
\begin{align*}
    f(0) &= 0 \\
    f(1) &= 1 \\
    f(n) &= f(n - 1) + f(n - 2), \text{ for } n \geq 2
\end{align*}
$$

As a recursively defined set (relation)

$$
\begin{align*}
    F_0 &= \emptyset \\
    F_{i+1} &= \{ \langle 0, 0 \rangle, \langle 1, 1 \rangle \} \\
    \cup \left\{ \langle n, f_{n_1} + f_{n_2} \rangle \mid \langle n_1, f_{n_1} \rangle \in F_i \text{ and } \langle n_2, f_{n_2} \rangle \in F_i \text{ and } n = n_1 + 1 = n_2 + 2 \right\}
\end{align*}
$$
Fibonacci as a Recursively Defined Set

\[
F_0 = \emptyset
\]
\[
F_{i+1} = \{\langle 0, 0 \rangle, \langle 1, 1 \rangle\} \\
\cup \left\{ \langle n, f_{n_1} + f_{n_2} \rangle \mid \langle n_1, f_{n_1} \rangle \in F_i \quad \text{and} \quad \langle n_2, f_{n_2} \rangle \in F_i \quad \text{and} \quad n = n_1 + 1 = n_2 + 2 \right\}
\]

For example:

\[
F_0 = \emptyset
\]
\[
F_1 = \{\langle 0, 0 \rangle, \langle 1, 1 \rangle\}
\]
\[
F_2 = \{\langle 0, 0 \rangle, \langle 1, 1 \rangle, \langle 2, 1 \rangle\}
\]
\[
F_3 = \{\langle 0, 0 \rangle, \langle 1, 1 \rangle, \langle 2, 1 \rangle, \langle 3, 2 \rangle\}
\]
\[
F_4 = \{\langle 0, 0 \rangle, \langle 1, 1 \rangle, \langle 2, 1 \rangle, \langle 3, 2 \rangle, \langle 4, 3 \rangle\}
\]
\[
F_5 = \]

Conventions

- \( \Sigma \) is an arbitrary alphabet. (In examples, \( \Sigma \) should be clear from context.)

- The variables \( a–e \) range over **letters** in \( \Sigma \).

- The variables \( u–z \) range over **words** over \( \Sigma^* \).

- The variables \( p–q \) range over **states** in \( Q \).
Recall

For any string $w$ and language $L$:

$$w \circ \varepsilon = w = \varepsilon \circ w \quad (1)$$

$$L \circ \{\varepsilon\} = L = \{\varepsilon\} \circ L \quad (2)$$

$$L^* = \{\varepsilon\} \cup L \circ L^* \quad (3)$$

$L^*$ is closed with respect to concatenation, for any $L$:

if $u \in L^*$ and $v \in L^*$ then $u \circ v \in L^*$
Finite Automata

... are “machines” for recognizing languages!

- They process input words a symbol at a time.
- An “accept light” flashes if the symbols read in so far are “OK”.

[Diagram of a finite automaton with an accept state and input symbols]
Formal Definition of Finite Automata

A finite automaton (DFA) is a quintuple \( \langle Q, \Sigma, q_0, \delta, A \rangle \) where:

- \( Q \) is a finite non-empty set of states;
- \( \Sigma \) is an alphabet;
- \( q_0 \in Q \) is the start state;
- \( \delta : Q \times \Sigma \rightarrow Q \) is the transition function; and
- \( A \subseteq Q \) is the set of accepting (final) states.
DFA Acceptance

Given a DFA $M = \langle Q, \Sigma, q_0, \delta, A \rangle$ and word $w \in \Sigma^*$:

- $M$ should **accept** $w$ if in processing $w$ a symbol at a time, $M$ goes to an accepting state.

- To formalize this we define a function

  $$\delta^* : Q \times \Sigma^* \rightarrow Q$$

  $\delta^*(q, w)$ should be the state reached from $q$ after processing $w$.

- How to define $\delta^*$?
Example of $\delta^*$

\[
\delta^*(0, aab) = \delta^*(\delta(0, a), ab) = \delta^*(2, ab)
\]
\[
= \delta^*(\delta(2, a), b) = \delta^*(3, b)
\]
\[
= \delta^*(\delta(3, b), \varepsilon) = \delta^*(1, \varepsilon)
\]
\[
= 1
\]

What is $\delta^*(0, abaa)$?
**Definition of $\delta^*$**

**Definition** Let $M = \langle Q, \Sigma, q_0, \delta, A \rangle$ be a DFA. Then $\delta^* : Q \times \Sigma^* \rightarrow Q$ is defined recursively:

$$
\delta^*(q, w) = \begin{cases} 
q & \text{if } w = \varepsilon \\
\delta^*(\delta(q, a), w') & \text{if } w = aw' \text{ and } a \in \Sigma
\end{cases}
$$

$\delta^*(q, w) = q'$ if $q'$ the state reached by processing $w$, starting from $q$. 
A DFA **accepts** a word if it reaches an accepting state after “consuming” the word.

**Definition** Let $M = \langle Q, \Sigma, q_0, \delta, A \rangle$ be a DFA.

- $M$ accepts $w \in \Sigma^*$ if $\delta^*(q_0, w) \in A$.
- $\mathcal{L}(M) = \{ w \in \Sigma^* | M \text{ accepts } w \}$ is the language accepted by $M$. 
Example: DFA for \( \{ w \in \{0, 1\}^* \mid w \text{ ends in 01} \} \)
Example: DFA for Valid Binary Numbers

- Must contain at least one digit.
- No leading 0s.