Lecture 24: Deterministic finite automata (model computers)

Primary objects of study:

- Finite automata (model computers)
- Regular expressions (string patterns)
- Applications in computer architecture, string processing

Analysis techniques:

- Structural induction, translating between computational models, connecting step-by-step operation with end-to-end specifications
- Applications in programming language design, systems analysis

Theoretical results:

- Many of our simple computational models are equally powerful
- There are problems that our simple computers can’t solve
- In other courses, you would extend these techniques to fully general models
- Similar techniques and results are used to reason about efficient computation
An automaton by example

Here is an example deterministic finite automaton (DFA):

\[ \Sigma = \{0, 1\} \]
\[ \Sigma^* = 100101 \]

inputs: strings with characters in an alphabet \( \Sigma \)
outputs: "yes" or "no"

reject state.
accept state.

input: 100101

output: "yes"
\[ \Sigma = \{0, 1\} \]

Start

\[ S(q_0, 0) = q_0 \Rightarrow 0 \]

Exactly at most one transition from any state \( q \) on any character \( a \).

A DFA \( M = (Q, \Sigma, \delta, q_0, A) \) contains

Every automaton has:
- A set of states \( Q \) (finite)
- An alphabet \( \Sigma \)
- A transition function \( \delta: Q \times \Sigma \rightarrow Q \)
- A start state \( q_0 \in Q \)
- A set \( A \subseteq Q \) called the set of accepting states.
Language of a DFA

\[ L(M) = \{ 100101, 1, 10, \ldots \} \]
\[ = \{ x \mid x \text{ has an odd } \# \text{ of } 1s \} \]

Defn: The language of \( M \), written \( L(M) \), is the set of strings that \( M \) accepts.

\( x = 100101 \) is accepted
\( x = 0011 \) is rejected
Exercise: building an automaton

\[ L = \{ x \mid x \text{ does not contain } 010 \text{ as a substring} \} \]

\[ = \{ 0, 1, \ldots, 0110 \} \]

\[ A = \{ q_0, q_1, q_2, q_3 \} \]

\[ Q = \{ \{q_0, q_1, q_2, q_3\} \} \]

\[ \Sigma = \{ 0, 1 \} \]

\[ \delta(\langle q_2, 0 \rangle, 0) = \langle q_1, 0 \rangle \]

\[ q_0 = \delta(\langle \varepsilon, 0 \rangle, 1) \]

\[ A = \{ q_0, q_1, q_2, q_3 \} \]
Formal definitions

**Defn:** A deterministic finite automaton (or DFA) $M$ is a 5-tuple $M = (Q, \Sigma, \delta, q_0, A)$ where

- $Q$ is a finite set (elements $q \in Q$ are called states)
- $\Sigma$ is a finite set (elements $a \in \Sigma$ are called characters)
- $\delta : Q \times \Sigma \rightarrow Q$ is called the transition function
- $q_0 \in Q$ is called the start state
- $A \subseteq Q$ is the set of accepting states

$L(M) =$ set of strings accepted by $M$

$x$ is accepted by $M$ if we process $x$

starting in $q_0$

we end in $A$

**Defn:** The extended transition function $\hat{\delta} : Q \times \Sigma^* \rightarrow Q$ gives the state $M$ transitions to after processing $x$, starting in state $q_0$.

Formally:

$\hat{\delta}(q_0, \varepsilon) = q_0$

$x \in \Sigma^* \implies \hat{\delta}(q_0, x) = \hat{\delta}(\hat{\delta}(q_0, x), a)$

$x$ is a substring of $xa$

**Defn:** We say $M$ accepts $x$ if

$\hat{\delta}(q_0, x) \in A$. We say $M$ rejects $x$

otherwise.

$L(M) = \{ x \in \Sigma^* \mid \hat{\delta}(q_0, x) \in A \}$