Lecture 27: A (stronger?) computational model

Last time: DFA are limited

▶ They can’t even count!

Today: let’s make a more flexible/powerful kind of automaton, and see if they are more powerful

We’ll add two new features:

▶ “Angelic non-determinism”
▶ “Epsilon transitions”

Definitions to have handy for today: definition of a DFA, language of a DFA, \( \delta \) for DFA
“Angelic” non-determinism

Idea: Non-deterministic automata can have many options while processing. They “magically” choose the “correct” option (one that will lead them to an accepting state).

$M$ accepts $x$ if it is possible to process (all the characters of) $x$ from the start state to reach an accepting state.

Does $M$ accept $00$? $01$? $001$? $0101$? $101$?
\( \varepsilon \)-transitions

**Example:** Draw an NFA that recognizes the language

\[
L := \{ x \mid x \text{ contains } 010 \text{ or } x \text{ has an odd number of } 1s \}
\]
Formal definition of an ($\varepsilon$-) NFA

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

- $Q$ is the finite set of states
- $\Sigma$ is the alphabet
- $\delta : Q \times \Sigma \rightarrow Q$ is the transition function
- $q_0 \in Q$ is the start state
- $A \subseteq Q$ is the set of accepting states

**Defn:** An NFA $N$ is a 6-tuple $N = (Q, \Sigma, \delta, \varepsilon, q_0, A)$:

- $Q$ is a finite set
- $\Sigma$ is an alphabet
- $\delta : Q \times \Sigma \rightarrow 2^Q$
- $\varepsilon : Q \rightarrow 2^Q$
- $q_0 \in Q$
- $A \subseteq Q$

\[ \varepsilon(q_0) = \{ q_0, q_3 \} \]
\[ \varepsilon(q_4) = \emptyset \]

**Example:**
- $Q = \{ q_0, q_1, \ldots, q_6 \}$
- $\varepsilon = \{ q_0, q_3 \}$
- $\delta(q_3, 0) = \{ q_3, q_4 \}$
- $\delta(q_4, 0) = \emptyset$
The language of an NFA

Definitions for DFA:
- \( L(M) \) is the set of strings that \( M \) accepts
- \( M \) accepts \( x \) if processing \( x \) brings \( M \) to an accept state
- \( \delta(q, x) \) gives the state that \( M \) ends in after processing \( x \)

Definitions for NFA:
- \( L(N) \) is the set of strings that \( N \) accepts
- \( N \) accepts \( x \) if it is possible for \( x \) to bring \( M \) to an accept state
  - \( \hat{\delta}(q, x) \) gives the set of states reachable from \( q \) on input \( x \)
  - \( \delta(S, x) \) gives the set of states reachable from \( S \) using 0 or more \( \epsilon \)-transitions

\[
\delta(q, x) = \bigcup \hat{\delta}(q', x) \\
\delta(q_0, x) = \hat{\delta}(q_0, x)
\]

\( \hat{\delta}(q_0, x) = \{ \hat{\delta}(q_0, y) \} \)

\( \delta(q_0, a) \):

\( N \) accepts \( x \) means \( \exists q \in A \) with \( q \in \delta(q_0, x) \)

i.e. if \( A \nsubseteq \delta(q_0, x) \neq \emptyset \)

\( L(N) \) is the set of \( x \) accepted by \( N \)

\[
L(N) = \{ x \in \{0, 1\}^* \mid \delta(q_0, x) \cap A \neq \emptyset \}
\]
Are DFA more powerful than NFA?

**Claim:** If $L$ is DFA-recognizable, then $L$ is NFA-recognizable

**Proof:** Suppose $L$ is DFA-recognizable. We want to show that $L$ is NFA-recognizable. Let $M := (Q_M, \Sigma, \delta_M, q_0M, A_M)$ be a DFA that recognizes $L$. We wish to construct an NFA $N = (Q_N, \Sigma, \delta_N, \varepsilon_N, q_0N, A_N)$ with $L(N) = L(M) = L$.

For example, suppose $M$ is given as follows:

\[
M = \begin{align*}
&\begin{array}{c}
\circ \quad \circ \quad \circ \\
q_0 \quad q_1 \quad q_1
\end{array} \\
&\begin{array}{c}
0 \quad 1 \quad 0 \\
1 \quad 1 \quad 1
\end{array}
\]

\[
N = \begin{align*}
&\begin{array}{c}
\circ \quad \circ \quad \circ \\
q_0 \quad q_1 \quad q_1
\end{array} \\
&\begin{array}{c}
0 \quad 1 \quad 0 \\
1 \quad 1 \quad 1
\end{array}
\]

\[
L = \{ x \in \Sigma^* \mid x \text{ has odd } \#(s^2) \}
\]

\[
\delta_N = ?, \quad \varepsilon_N = ?
\]
Are NFA more powerful than DFA?