Finite Automata: Informal Picture

Example: An “Electronic money” protocol

Ground Rules (continued):

- The store may ship goods to the customer.
- The store may redeem the money; that is, the money is sent back to the bank with a request that its value be added to the store’s bank account.
- The bank may transfer the money by creating a new, suitably encrypted money file and sending it to the store.
- Appropriate behavior of the three participants embodied in a protocol that could be represented as a finite automata ...
Finite Automata: Informal Picture

Example: An “Electronic money” protocol

Complete sets of transitions for the automata:

Finite automata

an important way to describe certain simple, but highly useful languages called “regular languages.”

A graph with a finite number of nodes called states (typically Q).

Arcs are labeled with one or more symbols from some alphabet (typically Σ).

One state is designated as the start state or initial state (typically q0); some states are final states or accepting states (typically F).
Finite Automata

Finite automata

- A *transition function* (typically $\delta$)
  - that takes a state and input symbol as arguments
  - that returns a state
  - where each “rule” of $\delta$ would be written as $\delta(q,a)=p$, where $q,p \in Q$ and $a \in \Sigma$.
  - intuitively, if a FA is in state $q \in Q$, and input $a \in \Sigma$ is received, then the FA goes to state $p \in Q$ (it is not necessarily the case that $q \neq p$).

Finite Automata

- A FA is represented as the five-tuple $A = (Q, \Sigma, \delta, q_0, F)$

Conventions:
- Input symbols typically $a, b$, etc., or digits.
- Strings of input symbols typically $u, v, \ldots, z$.
- States typically $q, p$, etc.

Deterministic Finite Automata

Definition

- A *deterministic finite automaton* consists of:
  - A finite set of states, often denoted $Q$
  - A finite set of input symbols, often denoted $\Sigma$
  - A transition function $\delta: Q \times \Sigma \rightarrow Q$
  - A start state $q_0 \in Q$
  - A set of final or accepting states $F \subseteq Q$
  - A DFA is also referred to using the five-tuple notation: $A = (Q, \Sigma, \delta, q_0, F)$

Example: Clamping Logic

Notes:

- We may think of an *accepting state* as representing a “1” output and nonaccepting states as representing a “0” output.
- A “clamping” circuit waits for a “1” input, and forever after makes a “1” output. However, to avoid clamping on spurious noise, we’ll design a FA that waits for two 1’s in a row, and “clamps” only then.
Example: Clamping Logic

In general, we can think of a state as representing a summary of the history of what has been seen on the input so far. So we need:

- state $q_0$, the start state, says that the most recent input (if there was one) was not a 1, and we have never seen two 1's in a row.
- state $q_1$, indicates we have never seen 11, but the previous input was 1.
- state $q_2$ is the only accepting state, indicating that we have at some time seen 11.

Thus, DFA $A = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\})$
where $\delta$ is given by:

<table>
<thead>
<tr>
<th>$\delta$</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\rightarrow$</td>
<td>$q_0$</td>
<td>$q_0$</td>
</tr>
<tr>
<td></td>
<td>$q_1$</td>
<td>$q_0$</td>
</tr>
<tr>
<td></td>
<td>$*$</td>
<td>$q_2$</td>
</tr>
</tbody>
</table>

by marking the start state with $\rightarrow$ and accepting states with $*$, the transition table that defines $\delta$ also specifies the entire DFA.

Example: HTML documents

Example:

On the next slide is a DFA that scans HTML documents, looking for a list of what could be title-author pairs, perhaps in a reading list for some literature course.

The DFA accepts whenever it finds the end of a list item.
Deterministic Finite Automata

The Language of a DFA

The language of a DFA $A = (Q, \Sigma, \delta, q_0, F)$, denoted $L(A)$, is defined by

$L(A) = \{ w \mid \Delta(q_0, w) \in F \}$

That is, the language of $A$ is the set of strings $w \in \Sigma^*$ that take the start state $q_0$ to one of the accepting states.

If $L$ is $L(A)$ for some DFA $A$, then we say that $L$ is a regular language.

Extending $\delta$ to Strings

If $\delta$ is a transition function for a DFA $A$, the extended transition function constructed from $\delta$, called $\Delta$, describes what happens when we start in any state and follow any sequence of inputs.

By definition, for DFA $A = (Q, \Sigma, \delta, q_0, F)$

- $\Delta(q, \varepsilon) = q$
- For $w \in \Sigma^*$, if $w = xa$ where $a \in \Sigma$, then $\Delta(q, w) = \delta(\Delta(q, x), a)$

Deterministic Finite Automata

Nondeterministic Finite Automata

A nondeterministic finite automata is similar to a DFA, except for $\delta$: $Q \times \Sigma \rightarrow 2^Q$

NFA that accepts all and only the strings of $\{0,1\}$ that end in 01:
Nondeterministic Finite Automata

**Definition**
- A **nondeterministic finite automaton** consists of:
  - A finite set of states, often denoted \( Q \)
  - A finite set of input symbols, often denoted \( \Sigma \)
  - A start state \( q_0 \in Q \)
  - A set of final or accepting states \( F \subseteq Q \)
  - A transition function \( \delta : Q \times \Sigma \rightarrow 2^Q \)

A NFA is also referred to using the five-tuple notation:

\[
A = (Q, \Sigma, \delta, q_0, F)
\]

**Representations**

\[
\begin{array}{c|cc}
\delta & 0 & 1 \\
\hline
q_0 & \{q_0, q_1\} & \{q_0\} \\
q_1 & \emptyset & \{q_2\} \\
* & q_2 & \emptyset
\end{array}
\]

The Language of an NFA

The **language** of an NFA \( A = (Q, \Sigma, \delta, q_0, F) \), denoted \( L(A) \), is defined by

\[
L(A) = \{ w \mid \Delta(q_0, w) \cap F \neq \emptyset \}
\]

That is, the language of \( A \) is the set of strings \( w \in \Sigma^* \) such that \( \Delta(q_0, w) \) contains at least one accepting state.

---

**Extending \( \delta \) to Strings**

- By definition, for NFA \( A = (Q, \Sigma, \delta, q_0, F) \)
  - \( \Delta(q, \epsilon) = \{ q \} \)
  - For \( w \in \Sigma^* \), if
    - \( w = \epsilon a \) where \( a \in \Sigma \); and
    - \( \Delta(q, x) = \{ p_1, p_2, \ldots, p_k \} \)
      where \( N_i \delta(p_i, a) = \{ r_1, r_2, \ldots, r_m \} \)
      then \( \Delta(q, w) = \{ r_1, r_2, \ldots, r_m \} \)
Nondeterministic Finite Automata

- Equivalence of DFAs and NFAs
  - Every language that can be described by some NFA can also be described by some DFA.
  - The DFA can have exponentially many states.
- Can be proven via \textit{subset construction} ...
  - Start from NFA \( N = (Q_N, \Sigma, \delta_N, q_0, F_N) \)
  - Goal: describe DFA \( D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D) \) such that \( L(D) = L(N) \).

\textbf{Theorem 2.11}

If \( D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D) \) is a DFA constructed from NFA \( N = (Q_N, \Sigma, \delta_N, q_0, F_N) \) by the \textit{subset construction}, then \( L(D) = L(N) \).

- Proof (by induction on \( |w| \)): \textit{See textbook} ...

\textbf{Theorem 2.12}

A language \( L \) is accepted by some DFA if and only if \( L \) is accepted by some NFA.

- Proof: \textit{See textbook} ...
Chapter 2: Finite Automata

Nondeterministic Finite Automata

Example
- Design an NFA to accept strings over \( \Sigma = \{1, 2, 3\} \) such that the last symbol appears previously, without any intervening higher symbol; e.g., \( w_{11}, w_{21112}, w_{312123} \).
- Use start state to mean “I guess I have not seen the symbol that matches the ending symbol.”
- Use three other states to represent a guess that the matching symbol has been seen, and remembers what symbol that is.

Note: Only 15 of the 32 possible states are accessible.

A Bad Case for the Subset Construction
- Recall: Given an NFA, the equivalent DFA can have exponentially many states. Why?
- Consider the NFA \( N \) where \( L(N) \) is the set of all strings in \( \{0, 1\} \) such that the \( n \)th symbol from the end is 1.
- Note: This NFA has no equivalent DFA with fewer than \( 2^n \) states.

An Application: Text Search
Finding Strings in Text
- Given a set of words, find all documents that contain one (or all) of those words.
- Inverted indexes
  - for each word, maintain a list of all the places where that word occurs.
- Automaton-based techniques
  - Suitable for:
    - Dynamic, rapidly changing search repositories
    - news analyst, financial analyst
    - “shopping robot” searching for current prices
    - Documents cannot be cataloged

Note: Only 15 of the 32 possible states are accessible.
An Application: Text Search

NFA for Text Search

From Figure 2.16 of IATLC, Hopcroft, Motwani, & Ullman, 2001.

An NFA that searches for the words “web” and “ebay”.

DFA to Recognize a Set of Keywords

From Figure 2.17 of IATLC, Hopcroft, Motwani, & Ullman, 2001.

An application: Text Search

Automaton-based string matching algorithms

Knuth-Morris-Pratt algorithm


Boyer-Moore algorithm


Both algorithms have sub-algorithms that build the automata for recognizing the target text patterns.

Finite Automata with ε-transitions

Uses of ε-transitions

- Allow ε to be a label on arcs
- Example:

An ε-NFA accepting decimal numbers

Uses of ε-transitions

- Allow ε to be a label on arcs
- Example:
Finite Automata with ε-transitions

Additional examples:

001 is accepted by the ε-NFA above by the path $q,s,r,s$, with label $001\varepsilon = 001$.

Finite Automata with ε-transitions

Formal Notation for an ε-NFA

- An ε-NFA consists of:
  - A finite set of states, often denoted $Q$
  - A finite set of input symbols, often denoted $\Sigma$
  - A start state $q_0 \in Q$
  - A set of final or accepting states $F \subseteq Q$
  - A transition function $\delta: Q \times \Sigma \cup \{\varepsilon\} \rightarrow 2^Q$

An ε-NFA is also referred to using the five-tuple notation:

$$E = (Q, \Sigma, \delta, q_0, F)$$

ε-Closures

- Given a state $q \in Q$ of ε-NFA $E = (Q, \Sigma, \delta, q_0, F)$.
- The ε-closure of $q$, denoted $ECLOSE(q)$, is defined recursively as:
  - $q \in ECLOSE(q)$
  - If $p \in ECLOSE(q)$ and $\delta(p, \varepsilon) = r$, then $r \in ECLOSE(q)$

Using ε-transitions to help recognize keywords.
Finite Automata with \(\varepsilon\)-transitions

### \(\varepsilon\)-Closures

**Example:**

\[
\begin{align*}
\text{ECLOSE}(q_0) &= \{ q_0, q_1 \} \\
\text{ECLOSE}(q_3) &= \{ q_3, q_5 \}
\end{align*}
\]

From Figure 2.18 of *IATLC*, Hopcroft, Motwani, & Ullman, 2001.

### Extended Transition Function, \(\Delta\), for \(\varepsilon\)-NFAs

- By definition, for an \(\varepsilon\)-NFA \(E = (Q, \Sigma, \delta, q_0, F)\)
  - \(\Delta(q, \varepsilon) = \text{ECLOSE}(q)\)
  - For \(w=xa\) where \(a \in \Sigma\), to compute \(\Delta(q, w)\):
    - Let \(\Delta(q, x) = \{ p_1, p_2, \ldots, p_k \} \)
    - Let \(N = \{ 1 \leq i \leq k \} \delta(p_i, a) = \{ r_1, r_2, \ldots, r_m \} \)
    - Then \(\Delta(q, w) = N \text{ECLOSE}(r_j)\)

### Language of an \(\varepsilon\)-NFA

- By definition, for an \(\varepsilon\)-NFA \(E = (Q, \Sigma, \delta, q_0, F)\), the *language* of \(E\), denoted \(L(E)\), is defined as
  \[
  L(E) = \{ w \in \Sigma^* \mid \Delta(q_0, w) \not\in \emptyset \}
  \]
Finite Automata with $\varepsilon$-transitions

**Eliminating-$\varepsilon$-transitions continued...**

For all $a \in \Sigma$ and $S \in Q_D$, compute $\delta_D(S,a)$ by

- Let $S = \{p_1, p_2, \ldots, p_k\}$
- Let $N_{1 \leq i \leq k} \delta(p_i,a) = \{r_1, r_2, \ldots, r_m\}$
- Then $\delta_D(S,a) = N_{1 \leq j \leq m} \text{ECLOSE}(r_j)$

**Theorem 2.22**

A language $L$ is accepted by some $\varepsilon$-NFA if and only if $L$ is accepted by some DFA.

**Copyright and Intellectual Property Notice**

This document and its contents are the Intellectual Property (IP) of Dr. Benjoe A. Juliano of the Department of Computer Science at California State University, Chico (CSUC). Dr. Juliano claims exclusive moral rights of ownership under current Copyright Laws (Title 17 of the United States Code and 1998 Digital Millennium Copyright Act) and IP Policies/Guidelines (CSUC EM83-08, EM87-07, and Article 39 of the CFACSU Contract) including, but not limited to:

- the exclusive right to copy, reproduce, and/or distribute this document;
- the right to be identified as the creator of this work (the right of attribution);
- the right to take action against false attribution; and
- the right to object to derogatory treatment of this work (the right of integrity).