Nondeterministic Finite Automata

Nondeterminism
Subset Construction
Nondeterminism

- A *nondeterministic finite automaton* has the ability to be in several states at once.
- Transitions from a state on an input symbol can be to any set of states.
Nondeterminism – (2)

• Start in one start state.
• Accept if any sequence of choices leads to a final state.
• **Intuitively**: the NFA always “guesses right.”
Example: Moves on a Chessboard

• States = squares.
• Inputs = r (move to an adjacent red square) and b (move to an adjacent black square).
• Start state, final state are in opposite corners.
Example: Chessboard – (2)

Accept, since final state reached
Formal NFA

- A finite set of states, typically $Q$.
- An input alphabet, typically $\Sigma$.
- A transition function, typically $\delta$.
- A start state in $Q$, typically $q_0$.
- A set of final states $F \subseteq Q$. 
Transition Function of an NFA

- $\delta(q, a)$ is a set of states.
- Extend to strings as follows:
  - **Basis**: $\delta(q, \epsilon) = \{q\}$
  - **Induction**: $\delta(q, wa) = \text{the union over all states } p \text{ in } \delta(q, w) \text{ of } \delta(p, a)$
Language of an NFA

- A string $w$ is **accepted** by an NFA if $\delta(q_0, w)$ contains at least one final state.
- That is, **there exists** a sequence of valid transitions from $q_0$ to a final state given the input $w$.
- The language of the NFA is the set of strings it accepts.
Example NFA

- Set of all strings with two consecutive a’s or two consecutive b’s:

\[
\begin{align*}
&\text{State 0} \\
&\text{State 1} \\
&\text{State 2} \\
&\text{State 3}
\end{align*}
\]

- Note that some states have an empty transition on an a or b, and some have multiple transitions on a or b.
Example 2: Language of an NFA

• For our chessboard NFA we saw that rbb is accepted.

• If the input consists of only b’s, the set of accessible states alternates between {5} and {1,3,7,9}, so only even-length, nonempty strings of b’s are accepted.

• What about strings with at least one r?
Equivalence of DFA’s, NFA’s

• A DFA can be turned into an NFA that accepts the same language.
• If $\delta_D(q, a) = p$, let the NFA have $\delta_N(q, a) = \{p\}$.
• Then the NFA is always in a set containing exactly one state – the state the DFA is in after reading the same input.
Equivalence – (2)

• Surprisingly, for any NFA there is a DFA that accepts the same language.
• Proof is the *subset construction*.
• The number of states of the DFA can be exponential in the number of states of the NFA.
• Thus, NFA’s accept *exactly* the regular languages.
Subset Construction

• Given an NFA with states $Q$, inputs $\Sigma$, transition function $\delta_N$, state state $q_0$, and final states $F$, construct equivalent DFA with:
  • States $2^Q$ (Set of subsets of $Q$).
  • Inputs $\Sigma$.
  • Start state $\{q_0\}$.
  • Final states $= \text{all those with a member of } F$. 
Critical Point

• The DFA states have *names* that are sets of NFA states.
• But as a DFA state, an expression like \{p,q\} must be read as a single symbol, not as a set.
• **Analogy**: a class of objects whose values are sets of objects of another class.
The transition function $\delta_D$ is defined by:
$\delta_D(\{q_1, \ldots, q_k\}, a)$ is the union over all $i = 1, \ldots, k$ of $\delta_N(q_i, a)$.

**Example:** We’ll construct the DFA equivalent of our “chessboard” NFA.
### Example: Subset Construction

<table>
<thead>
<tr>
<th></th>
<th>r</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2,4</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>4,6</td>
<td>1,3,5</td>
</tr>
<tr>
<td>3</td>
<td>2,6</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>2,8</td>
<td>1,5,7</td>
</tr>
<tr>
<td>5</td>
<td>2,4,6,8</td>
<td>1,3,7,9</td>
</tr>
<tr>
<td>6</td>
<td>2,8</td>
<td>3,5,9</td>
</tr>
<tr>
<td>7</td>
<td>4,8</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>4,6</td>
<td>5,7,9</td>
</tr>
<tr>
<td>9</td>
<td>6,8</td>
<td>5</td>
</tr>
</tbody>
</table>

Alert: What we’re doing here is the *lazy* form of DFA construction, where we only construct a state if we are forced to.
Example: Subset Construction

<table>
<thead>
<tr>
<th></th>
<th>r</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2,4</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>4,6</td>
<td>{1}</td>
</tr>
<tr>
<td>3</td>
<td>2,6</td>
<td>1,3,5</td>
</tr>
<tr>
<td>4</td>
<td>2,8</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>2,4,6,8</td>
<td>1,3,7,9</td>
</tr>
<tr>
<td>6</td>
<td>2,8</td>
<td>3,5,9</td>
</tr>
<tr>
<td>7</td>
<td>4,8</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>4,6</td>
<td>5,7,9</td>
</tr>
<tr>
<td>9</td>
<td>6,8</td>
<td>5</td>
</tr>
</tbody>
</table>

*
**Example: Subset Construction**

<table>
<thead>
<tr>
<th></th>
<th>r</th>
<th>b</th>
<th>r</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2,4</td>
<td>5</td>
<td>{1}</td>
<td>{5}</td>
</tr>
<tr>
<td>2</td>
<td>4,6</td>
<td>1,3,5</td>
<td>{2,4}</td>
<td>{1,3,5,7}</td>
</tr>
<tr>
<td>3</td>
<td>2,6</td>
<td>5</td>
<td>{5}</td>
<td>{1,3,7,9}</td>
</tr>
<tr>
<td>4</td>
<td>2,8</td>
<td>1,5,7</td>
<td>{2,4,6,8}</td>
<td>{5}</td>
</tr>
<tr>
<td>5</td>
<td>2,4,6,8</td>
<td>1,3,7,9</td>
<td>{2,4,6,8}</td>
<td>{1,3,7,9}</td>
</tr>
<tr>
<td>6</td>
<td>2,8</td>
<td>3,5,9</td>
<td>{1,3,5,7}</td>
<td>{1,3,7,9}</td>
</tr>
<tr>
<td>7</td>
<td>4,8</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>4,6</td>
<td>5,7,9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>6,8</td>
<td>5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*
**Example:** Subset Construction

<table>
<thead>
<tr>
<th></th>
<th>r</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2,4</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>4,6</td>
<td>1,3,5</td>
</tr>
<tr>
<td>3</td>
<td>2,6</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>2,8</td>
<td>1,5,7</td>
</tr>
<tr>
<td>5</td>
<td>2,4,6,8</td>
<td>1,3,7,9</td>
</tr>
<tr>
<td>6</td>
<td>2,8</td>
<td>3,5,9</td>
</tr>
<tr>
<td>7</td>
<td>4,8</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>4,6</td>
<td>5,7,9</td>
</tr>
<tr>
<td>9</td>
<td>6,8</td>
<td>5</td>
</tr>
</tbody>
</table>

→ {1} → {2,4} → {1,3,5,7} → {2,4,6,8} → {1,3,7,9} → {2,4,6,8} → {1,3,5,7,9} → {1,3,7,9} → {1,3,5,7,9}
## Example: Subset Construction

<table>
<thead>
<tr>
<th></th>
<th>r</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2,4</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>4,6</td>
<td>1,3,5</td>
</tr>
<tr>
<td>3</td>
<td>2,6</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>2,8</td>
<td>1,5,7</td>
</tr>
<tr>
<td>5</td>
<td>2,4,6,8</td>
<td>1,3,7,9</td>
</tr>
<tr>
<td>6</td>
<td>2,8</td>
<td>3,5,9</td>
</tr>
<tr>
<td>7</td>
<td>4,8</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>4,6</td>
<td>5,7,9</td>
</tr>
<tr>
<td>9</td>
<td>6,8</td>
<td>5</td>
</tr>
</tbody>
</table>

\[ \text{r} \rightarrow \{1\} \quad \{2,4\} \quad \{5\} \]
\[ \{2,4\} \quad \{2,4,6,8\} \quad \{1,3,5,7\} \]
\[ \{5\} \quad \{2,4,6,8\} \quad \{1,3,7,9\} \]
\[ \{2,4,6,8\} \quad \{2,4,6,8\} \quad \{1,3,5,7,9\} \]
\[ \{1,3,5,7\} \quad \{2,4,6,8\} \quad \{1,3,5,7,9\} \]
\[ \{1,3,7,9\} \quad \{2,4,6,8\} \quad \{1,3,5,7,9\} \]
\[ \{1,3,5,7,9\} \quad \{2,4,6,8\} \quad \{1,3,5,7,9\} \]

*
**Example: Subset Construction**

<table>
<thead>
<tr>
<th></th>
<th>(r)</th>
<th>(b)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2,4</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>4,6</td>
<td>1,3,5</td>
</tr>
<tr>
<td>3</td>
<td>2,6</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>2,8</td>
<td>1,5,7</td>
</tr>
<tr>
<td>5</td>
<td>2,4,6,8</td>
<td>1,3,7,9</td>
</tr>
<tr>
<td>6</td>
<td>2,8</td>
<td>3,5,9</td>
</tr>
<tr>
<td>7</td>
<td>4,8</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>4,6</td>
<td>5,7,9</td>
</tr>
<tr>
<td>9</td>
<td>6,8</td>
<td>5</td>
</tr>
</tbody>
</table>

\[ \rightarrow \{1\} \quad \rightarrow \{2,4\} \quad \rightarrow \{5\} \]

\[ \rightarrow \{2,4\} \quad \rightarrow \{2,4,6,8\} \quad \rightarrow \{1,3,5,7\} \]

\[ \rightarrow \{2,4,6,8\} \quad \rightarrow \{2,4,6,8\} \quad \rightarrow \{1,3,7,9\} \]

\[ \rightarrow \{1,3,5,7\} \quad \rightarrow \{2,4,6,8\} \quad \rightarrow \{1,3,5,7,9\} \]

\[ \rightarrow \{1,3,7,9\} \quad \rightarrow \{2,4,6,8\} \quad \rightarrow \{1,3,5,7,9\} \]

\[ \rightarrow \{1,3,7,9\} \quad \rightarrow \{2,4,6,8\} \quad \rightarrow \{5\} \]

\[ \rightarrow \{1,3,5,7,9\} \]
Example: Subset Construction

<table>
<thead>
<tr>
<th></th>
<th>r</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2,4</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>4,6</td>
<td>1,3,5</td>
</tr>
<tr>
<td>3</td>
<td>2,6</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>2,8</td>
<td>1,5,7</td>
</tr>
<tr>
<td>5</td>
<td>2,4,6,8</td>
<td>1,3,7,9</td>
</tr>
<tr>
<td>6</td>
<td>2,8</td>
<td>3,5,9</td>
</tr>
<tr>
<td>7</td>
<td>4,8</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>4,6</td>
<td>5,7,9</td>
</tr>
<tr>
<td>9</td>
<td>6,8</td>
<td>5</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
\rightarrow \{&1\} & \{2,4\} & \{5\} \\
& \{2,4\} & \{2,4,6,8\} & \{1,3,5,7\} \\
& \{5\} & \{2,4,6,8\} & \{1,3,7,9\} \\
& \{2,4,6,8\} & \{2,4,6,8\} & \{1,3,5,7,9\} \\
& \{1,3,5,7\} & \{2,4,6,8\} & \{1,3,5,7,9\} \\
\star & \{1,3,7,9\} & \{2,4,6,8\} & \{5\} \\
\star & \{1,3,5,7,9\} & \{2,4,6,8\} & \{1,3,5,7,9\}
\end{align*}
\]
Proof of Equivalence: Subset Construction

• The proof is almost a pun.
• Show by induction on $|w|$ that
  \[
  \delta_N(q_0, w) = \delta_D(\{q_0\}, w)
  \]
• **Basis:** $w = \varepsilon$: $\delta_N(q_0, \varepsilon) = \delta_D(\{q_0\}, \varepsilon) = \{q_0\}$.
Induction

• Assume IH for strings shorter than \( w \).
• Let \( w = xa \); IH holds for \( x \).
• Let \( \delta_N(q_0, x) = \delta_D(\{q_0\}, x) = S \).
• Let \( T = \) the union over all states \( p \) in \( S \) of \( \delta_N(p, a) \).
• Then \( \delta_N(q_0, w) = \delta_D(\{q_0\}, w) = T \).
  • For NFA: the extension of \( \delta_N \).
  • For DFA: definition of \( \delta_D \) plus extension of \( \delta_D \).
    • That is, \( \delta_D(S, a) = T \); then extend \( \delta_D \) to \( w = xa \).
NFA’s With $\varepsilon$-Transitions

- We can allow state-to-state transitions on $\varepsilon$ input.
- These transitions are done spontaneously, without looking at the input string.
- A convenience at times, but still only regular languages are accepted.
Example: $\epsilon$-NFA

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>$\epsilon$</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>{E}</td>
<td>{B}</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>B</td>
<td>$\emptyset$</td>
<td>{C}</td>
<td>{D}</td>
</tr>
<tr>
<td>C</td>
<td>$\emptyset$</td>
<td>{D}</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>D</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>E</td>
<td>{F}</td>
<td>$\emptyset$</td>
<td>{B, C}</td>
</tr>
<tr>
<td>F</td>
<td>{D}</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
</tr>
</tbody>
</table>
Closure of States

• CL(q) = set of states you can reach from state q following only arcs labeled $\varepsilon$.

• Example: CL(A) = \{A\}; CL(E) = \{B, C, D, E\}.

• Closure of a set of states = union of the closure of each state.
Extended Delta

• **Basis:** \( \delta(q, \epsilon) = \text{CL}(q) \).

• **Induction:** \( \delta(q, xa) \) is computed as follows:
  1. Start with \( \delta(q, x) = S \).
  2. Take the union of \( \text{CL}(\delta(p, a)) \) for all \( p \) in \( S \).

• **Intuition:** \( \delta(q, w) \) is the set of states you can reach from \( q \) following a path labeled \( w \).

And notice that \( \delta(q, a) \) is *not* that set of states, for symbol \( a \).
Example:
Extended Delta

\[ \delta(A, \varepsilon) = CL(A) = \{A\} \]
\[ \delta(A, 0) = CL(\{E\}) = \{B, C, D, E\} \]
\[ \delta(A, 01) = CL(\{C, D\}) = \{C, D\} \]

*Language* of an \( \varepsilon \)-NFA is the set of strings \( w \) such that \( \delta(q_0, w) \) contains a final state.
Equivalence of NFA, $\epsilon$-NFA

• Every NFA is an $\epsilon$-NFA.
  • It just has no transitions on $\epsilon$.
• Converse requires us to take an $\epsilon$-NFA and construct an NFA that accepts the same language.
• We do so by combining $\epsilon$–transitions with the next transition on a real input.

**Warning:** This treatment is a bit different from that in the text.
Picture of $\varepsilon$-Transition Removal

Transitions on $\varepsilon$

Transitions on $\varepsilon$
Picture of $\varepsilon$-Transition Removal

Text goes from here

To here, and performs the subset construction

Transitions on $\varepsilon$

Transitions on $\varepsilon$
Picture of $\varepsilon$-Transition Removal

We'll go from here to here, with no subset construction.

Transitions on $\varepsilon$
Equivalence – (2)

• Start with an $\varepsilon$-NFA with states $Q$, inputs $\Sigma$, start state $q_0$, final states $F$, and transition function $\delta_E$.

• Construct an “ordinary” NFA with states $Q$, inputs $\Sigma$, start state $q_0$, final states $F'$, and transition function $\delta_N$. 
Equivalence – (3)

- Compute $\delta_N(q, a)$ as follows:
  1. Let $S = \text{CL}(q)$.
  2. $\delta_N(q, a)$ is the union over all $p$ in $S$ of $\delta_E(p, a)$.

- $F' = \text{the set of states } q \text{ such that } \text{CL}(q) \text{ contains a state of } F$.

- **Intuition**: $\delta_N$ incorporates $\epsilon$–transitions before using $a$ but not after.
Equivalence – (4)

• Prove by induction on $|w|$ that

$$CL(\delta_N(q_0, w)) = \delta_E(q_0, w).$$

• Thus, the $\epsilon$-NFA accepts $w$ if and only if the “ordinary” NFA does.
Interesting closures: \( \text{CL}(B) = \{B, D\}; \text{CL}(E) = \{B, C, D, E\} \)

Since closure of \( E \) includes \( B \) and \( C \); which have transitions on 1 to \( C \) and \( D \).

Since closures of \( B \) and \( E \) include final state \( D \).

Example: \( \epsilon \)-NFA-to-NFA

\[ \begin{array}{c|ccc}
\text{A} & 0 & 1 & \epsilon \\
\hline
\text{E} & \{E\} & \emptyset & \emptyset \\
\text{B} & \emptyset & \{C\} & \{D\} \\
\text{C} & \emptyset & \{D\} & \emptyset \\
\text{D} & \emptyset & \emptyset & \emptyset \\
\text{F} & \{D\} & \emptyset & \emptyset \\
\end{array} \]

\[ \begin{array}{c|ccc}
\text{A} & 0 & 1 & \epsilon \\
\hline
\text{E} & \{E\} & \emptyset & \emptyset \\
\text{B} & \emptyset & \{C\} & \emptyset \\
\text{C} & \emptyset & \{D\} & \emptyset \\
\text{D} & \emptyset & \emptyset & \emptyset \\
\text{E} & \{F\} & \emptyset & \{C, D\} \\
\text{F} & \{D\} & \emptyset & \emptyset \\
\end{array} \]

Since closure of \( E \) includes \( B \) and \( C \); which have transitions on 1 to \( C \) and \( D \).
Summary

• DFA’s, NFA’s, and ε–NFA’s all accept exactly the same set of languages: the regular languages.
• The NFA types are easier to design and may have exponentially fewer states than a DFA.
• But only a DFA can be implemented in linear time!