Three Subclasses of FAs*

- **EFA (Empty-string Finite Automata):**
  FA in which each transition label has length 0 or 1.

- **NFA (Nondeterministic Finite Automata):**
  FA in which each transition label has length 1.

- **DFA (Deterministic Finite Automata):**
  FA in which each transition label has length 1, and for every symbol $a$ in the alphabet of the automaton, there is exactly one transition labeled $a$ from each state. I.e., the transition relation is a total function $T : Q \times a \rightarrow Q$.

* This is Stoughton's terminology and classification. Many other authors use NFA for what Stoughton calls EFA, and do not give a distinct name for what Stoughton calls NFA.
Injections

Subset inclusion induces obvious injections:
- EFA → FA
- NFA → EFA
- DFA → NFA

These injections are Forlan functions:
- EFA.injToFA: efa → fa
- NFA.injToEFA: nfa → efa
- DFA.injToNFA: dfa → nfa

Transformations

Less obvious are the red transformations, which we focus on today.

With these transformations (and the injections) we can convert any of the four representations to any other, demonstrating that they all describe the same set of languages = the regular languages.
A Running Example

Here's an example we'll use throughout this lecture:

Which of the following strings are accepted, and why?
- aaa
- aba
- abaa
- abab
- aaba
- aabb
- aaaba

Transforming FA to EFA

Simply break every transition with a multi-symbol label $a_1,a_2,...,a_n$ into $n$ single-symbol transitions involving $n-1$ new non-final states:

Converting our running example:

FA to EFA
Transforming EFA to NFA: Notation

Some notation (mine, not Sipser’s or Stoughton’s)

- $p \xrightarrow{\varepsilon} q$ means $p$ goes to $q$ via a transition labeled $\varepsilon$
- $p \xrightarrow{\varepsilon*} q$ means $p$ goes to $q$ via one or more transitions labeled $\varepsilon$ (the transitive closure of $\xrightarrow{\varepsilon}$)
- $p \xrightarrow{\varepsilon*} q$ means $p$ goes to $q$ via zero or more transitions labeled $\varepsilon$ (the reflexive, transitive closure of $\xrightarrow{\varepsilon}$)

Example:

```
<table>
<thead>
<tr>
<th></th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>J</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Transforming EFA to NFA: Algorithm

- If $p \xrightarrow{\varepsilon} q$ and $q \xrightarrow{\varepsilon} r$ add a transition $p \xrightarrow{\varepsilon} r$.
- If $p \xrightarrow{\varepsilon*} q$, where $q$ is an accepting state, make $p$ an accepting state (since it accepts $\varepsilon$).

After performing the above steps everywhere, remove all $\varepsilon$ transitions.

Example:
**Converting EFA to NFA: More Formally**

**States:** $Q_{NFA} = Q_{EFA}$

**Alphabet:** $\Sigma_{NFA} = \Sigma_{EFA}$

**Start State:** $s_{NFA} = s_{EFA}$

**Final States:** $F_{NFA} = \{ p \mid p \xrightarrow{\varepsilon}^* q \text{ by } \delta_{EFA} \text{ and } q \text{ in } F_{EFA} \}$

**Transitions:**

$\delta_{NFA} = (Q_{NFA} \times \Sigma_{NFA}, P(Q_{NFA}),$

$\{(p, \sigma), \{r \mid p \xrightarrow{\varepsilon}^* q \text{ by } \delta_{EFA} \text{ and } r \in \delta_{EFA}(q, \sigma)\}\})$

*Note: some states may become unreachable, and so can be omitted:*

---

**EFA to NFA: Alternative Approaches**

Rather than taking $\varepsilon$ steps before a transition (Stoughton) could either:

1. take $\varepsilon$ steps after a transition (Sipser)
2. take $\varepsilon$ steps before and after a transition

But with (1) have problems defining start state, so must do at same time as NFA to DFA transformation (i.e., Sipser defines a single EFA-to-DFA transformation rather than separate EFA-to-NFA and NFA-to-DFA transformations).
Transforming EFA to NFA: Running Example

Current State of our Running Example

Which of the following strings are accepted, and why?

- aaa
- aba
- abaa
- abab
- aaba
- aabb
- aaaba
Transforming NFA to DFA: Idea

To construct a DFA from an NFA, we explore all possible labeled paths for a string in parallel and accept if any such path ends in a final state.

- Each state of DFA = set of states in NFA
- Start state of DFA = singleton set of start state of NFA
- A final state of DFA = any set of states from NFA containing a final state from the NFA
- A transition of DFA = set of all states in NFA that can be reached from a set of states by following transitions labeled by a symbol.

Some NFA transitions

\[ \{H,I,J,K\} \quad \{I,J,K\} \quad \{G,J\} \]

Some corresponding DFA transitions

\[ \{G\} \quad \{(H,I)\} \quad \{(I)\} \quad \{\{\}\} \]

NFA to DFA: The Subset Construction

The subset construction formalizes the sketch from the previous slide:

- Each state of DFA = set of states in NFA
  \[ Q_{DFA} = P(Q_{NFA}) \quad \text{(powerset of } Q_{NFA}) \]
- Alphabet of DFA = alphabet of NFA
  \[ \Sigma_{DFA} = \Sigma_{NFA} \]
- Start state of DFA = singleton set of start state of NFA
  \[ s_{DFA} = \{s_{NFA}\} \]
- A final state of DFA = any set of states from NFA containing a final state from the NFA
  \[ F_{DFA} = \{S \mid S \subseteq Q_{NFA} \text{ and } S \cap F_{NFA} \neq \emptyset\} \]
- A transition of DFA = set of all states in NFA that can be reached from a set of states by following transitions labeled by a symbol.
  \[ \delta_{DFA} = \{Q_{DFA} \times \Sigma_{DFA}, Q_{DFA}, \delta_{NFA}(q,\sigma) \mid q \in Q_{NFA}\} \]
Transforming Finite Automata

Transforming NFA to DFA: Running Example

NFA to DFA Example: \{A\}, a -> \{C,E\}
NFA to DFA Example: \{A\}, b → \{F\}

NFA to DFA Example: \{C,E\}, a → \{C\}
**NFA to DFA Example: \{C,E\}, b \rightarrow \{B,F\}**

- **NFA to DFA**

**NFA to DFA Example: \{C\}, a \rightarrow \{C\}**

- **NFA to DFA**
NFA to DFA Example: \{C\}, b \rightarrow \{F\}

NFA to DFA Example: \{F\}, a \rightarrow \{D\}
NFA to DFA Example: \{F\}, b \rightarrow \{\}

NFA to DFA Example: \{D\}, (a,b) \rightarrow \{D\}
NFA to DFA Example: \{B,F\}, \text{a} \to \{E,D\}

NFA to DFA Example: \{B,F\}, \text{b} \to \{\}
Transforming Finite Automata

NFA to DFA Example: \{E,D\}, \text{a} \rightarrow \{D\}

NFA to DFA

NFA to DFA Example: \{E,D\}, \text{b} \rightarrow \{B,D\}

NFA to DFA

Transforming Finite Automata 15-27

Transforming Finite Automata 15-28
NFA to DFA Example: \( \{B,D\}, a \rightarrow \{E,D\} \)

NFA to DFA Example: \( \{B,D\}, b \rightarrow \{D\} \)
Which of the following strings are accepted, and why?

- aaa
- aba
- abaa
- abab
- aaba
- aabb
- aaaba

Another Example

Transform this FA to EFA, NFA, and DFA.
Subset Construction: Discussion

- Given an NFA with $n$ states, how many reachable states could there be in the DFA in the worst case?

- Why does the subset construction algorithm terminate?

- Why is the subset construction algorithm correct?

Transformations in Forlan

All the transformations we studied today are implemented in Forlan:

```plaintext
faToEFA : fa -> efa
efaToNFA : efa -> nfa
nfaToDFA : nfa -> dfa
val faToDFA = nfaToDFA o efaToNFA o faToEFA
```