Determinizing NFAs

(Material taken from “Theory of Computation” by Michael Sipser)
Determinizing an NFA

1. Remove ε-transitions
2. Convert NFA without ε-transitions to DFA
Removing $\varepsilon$-Transitions

- Input: NFA $M \equiv (Q, \Sigma, \delta, q, F)$ with $\varepsilon$-transitions
- Output: NFA $M' \equiv (Q, \Sigma - \{\varepsilon\}, \delta', q, F')$ without $\varepsilon$-transitions

Algorithm:

1. Compute epsilon closure for every state in $Q$
   - $\text{eclose}(s) =$ set of all states reachable from $s$ using zero or more $\varepsilon$-transitions
2. $F' \leftarrow F$; Add $p$ to $F'$ iff $\text{eclose}(p)$ contains some state in $F$
3. Add an edge from $p$ to $q$ labelled ‘$a$’ iff there is an edge labelled ‘$a$’ in $M$ from some state in $\text{eclose}(p)$ to $q$
4. Delete all edges labelled ‘$\varepsilon$’; Use new edges to compute $\delta'$
Example 1

• Remove $\epsilon$-transitions from the following NFA
Example 1: Step 1

• Compute epsilon closures

<table>
<thead>
<tr>
<th>s</th>
<th>eclose(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{1, 3}</td>
</tr>
<tr>
<td>2</td>
<td>{2}</td>
</tr>
<tr>
<td>3</td>
<td>{3}</td>
</tr>
</tbody>
</table>
Example 1: Step 2

Compute $F'$

- Initially, $F' = \{1\}$ (M has only one final state: 1)
- There doesn’t exist a state $p$ (other than 1) such that $\text{eclose}(p)$ contains 1
- No extra states get added to $F'$
Example 1: Step 3

Add extra edges according to $\text{eclose}$

- 1 is the only state whose $\text{eclose}$ has another state, i.e., 3
- For every edge labelled ‘a’ from 3 to some state $q$, add an edge labelled ‘a’ from 1 to $q$
- There is only one outgoing edge from 3 (to 1 labelled ‘a’)
- Add a self loop in 1 labelled ‘a’
Example 1: Step 4

- Remove edges labelled $\varepsilon$
Example 2

- Remove $\varepsilon$-transitions from the following NFA
Example 2: Step 1

- Compute epsilon closures

```
<table>
<thead>
<tr>
<th>s</th>
<th>eclose(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{1, 2, 3}</td>
</tr>
<tr>
<td>2</td>
<td>{2}</td>
</tr>
<tr>
<td>3</td>
<td>{3}</td>
</tr>
<tr>
<td>4</td>
<td>{4}</td>
</tr>
<tr>
<td>5</td>
<td>{3, 5}</td>
</tr>
</tbody>
</table>
```
Example 2: Step 2

Compute $F'$

- Initially, $F' = \{2, 5\}$
- 1 is the only state (other than 2 and 5) such that $\text{eclose}(1)$ contains either 2 or 5
- Add 1 to $F'$
- $F' = \{1, 2, 5\}$
Example 2: Step 3

Add extra edges according to eclose

- 1 and 5 are the only states whose eclose has other states, i.e., \{2, 3\} and \{3\}, respectively
- Because of edge 2 -> 2 labelled ‘a’, add edge 1 -> 2 labelled ‘a’. Similarly add edges 1 -> 4 labelled ‘a’, and 5 -> 4 labelled ‘a’
Example 2: Step 4

- Remove edges labelled $\varepsilon$
- Also remove state 3 because it has no incoming edge
Converting NFA w/o $\varepsilon$-Transitions to DFA

- Input: NFA $M \equiv (Q, \Sigma, \delta, q, F)$ without $\varepsilon$-transitions
- Output: DFA $M' \equiv (Q', \Sigma, \delta', q', F')$

Algorithm:
1. $Q' = P(Q)$ ($P(Q)$ is the power set of $Q$, i.e., the set of all subsets of $Q$)
2. For each $R \in Q'$ and $a \in \Sigma$, $\delta'(R, a) = \bigcup_{r \in R} \delta(r, a)$
3. $q' = \{q\}$
4. $F' = \{R \in Q' \mid R \text{ contains a final state of } M\}$
Example 1

- Convert the following NFA to a DFA
Example 1: Step 1

- Create states in $Q'$ from $P(Q)$

{}  
{1}  
{2}  
{1, 2}  
{3}  
{1, 3}  
{2, 3}  
{1, 2, 3}
Example 1: Step 2

- Create transitions in \( \delta' \) by agglomerating transitions in \( \delta \)
Example 1: Step 3

• Identify initial state
Example 1: Step 4

- Identify final states, and remove states with no incoming edges
Example 2

• Convert the following NFA to a DFA
Example 2

• Solution