18.6 DFA to Regular Expression
Back to Regular Languages

We saw the following two theorems previously.

<table>
<thead>
<tr>
<th>Theorem 18.1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>For every NFA $N$ over a finite alphabet $\Sigma$ there is DFA $M$ such that $L(M) = L(N)$.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Theorem 18.2.</th>
</tr>
</thead>
<tbody>
<tr>
<td>For every regular expression $r$ over finite alphabet $\Sigma$ there is a NFA $N$ such that $L(N) = L(r)$.</td>
</tr>
</tbody>
</table>

We claimed the following theorem which would prove equivalence of NFAs, DFAs and regular expressions.

<table>
<thead>
<tr>
<th>Theorem 18.3.</th>
</tr>
</thead>
<tbody>
<tr>
<td>For every DFA $M$ over a finite alphabet $\Sigma$ there is a regular expression $r$ such that $L(M) = L(r)$.</td>
</tr>
</tbody>
</table>
Back to Regular Languages

We saw the following two theorems previously.

**Theorem 18.1.**

For every NFA $N$ over a finite alphabet $\Sigma$ there is DFA $M$ such that $L(M) = L(N)$.

**Theorem 18.2.**

For every regular expression $r$ over finite alphabet $\Sigma$ there is a NFA $N$ such that $L(N) = L(r)$.

We claimed the following theorem which would prove equivalence of NFA's, DFA's and regular expressions.

**Theorem 18.3.**

For every DFA $M$ over a finite alphabet $\Sigma$ there is a regular expression $r$ such that $L(M) = L(r)$. 
DFA to Regular Expression

Given DFA $M = (Q, \Sigma, \delta, q_1, F)$ want to construct an equivalent regular expression $r$.

Idea:

- Number states of DFA: $Q = \{q_1, \ldots, q_n\}$ where $|Q| = n$.
- Define $L_{i,j} = \{w | \delta(q_i, w) = q_j\}$. Note $L_{i,j}$ is regular. Why?
- $L(M) = \bigcup_{q_i \in F} L_{1,i}$.
- Obtain regular expression $r_{i,j}$ for $L_{i,j}$.
- Then $r = \sum_{q_i \in F} r_{1,i}$ is regular expression for $L(M)$ – here the summation is the or operator.

Note: Using $q_1$ for start state is intentional to help in the notation for the recursion.
A recursive expression for $L_{i,j}$

Define $L_{i,j}^k$ be set of strings $w$ in $L_{i,j}$ such that the highest index state visited by $M$ on walk from $q_i$ to $q_j$ (not counting end points $i$ and $j$) on input $w$ is at most $k$.

**Claim:**

\[
L_{i,j}^0 = \{ a \in \Sigma \mid \delta(q_i, a) = q_i \}^* \\
L_{i,j}^0 = L_{i,i}^0 \{ a \in \Sigma \mid \delta(q_i, a) = q_j \} L_{j,j}^0 \quad \text{if } i \neq j \\
L_{i,j}^k = L_{i,j}^{k-1} \cup \left( L_{i,k}^{k-1} \cdot L_{k,k}^{k-1} \cdot L_{k,j}^{k-1} \right) \quad \text{if } i \neq j \\
L_{i,i}^k = \left( L_{i,i}^{k-1} \cup L_{i,k}^{k-1} \cdot L_{k,k}^{k-1} \cdot L_{k,i}^{k-1} \right)^* \\
L_{i,j} = L_{i,j}^n.
\]
A recursive expression for $L_{i,j}$

Claim:

\[
L^0_{i,i} = \{ a \in \Sigma \mid \delta(q_i, a) = q_i \}^* \\
L^0_{i,j} = L^0_{i,i} \{ a \in \Sigma \mid \delta(q_i, a) = q_j \} L^0_{j,j} \\
L^k_{i,j} = L^k_{i,j} \cup \left( L^k_{i,k} \cdot L^k_{k,k} \cdot L^k_{k,j} \right) \\
L^k_{i,i} = \left( L^k_{i,i} \cup L^k_{i,k} \cdot L^k_{k,k} \cdot L^k_{k,i} \right)^* \\
L_{i,j} = L^*_i j.
\]

Proof: by picture
A recursive expression for $L_{i,j}$

Claim:

$L_{i,i}^0 = \{ a \in \Sigma \mid \delta(q_i, a) = q_i \}^*$

$L_{i,j}^0 = L_{i,i}^0 \{ a \in \Sigma \mid \delta(q_i, a) = q_j \} L_{j,j}^0$ if $i \neq j$

$L_{i,j}^k = L_{i,j}^{k-1} \cup \left( L_{i,k}^{k-1} \cdot L_{k,k}^{k-1} \cdot L_{k,j}^{k-1} \right)$ if $i \neq j$

$L_{i,i}^k = \left( L_{i,i}^{k-1} \cup L_{i,k}^{k-1} \cdot L_{k,k}^{k-1} \cdot L_{k,i}^{k-1} \right)^*$

$L_{i,j} = L_{i,j}^n$.

The desired language is

$L(M) = \bigcup_{q_i \in F} L_{1,i} = \bigcup_{q_i \in F} L_{1,i}^n$
A regular expression for $L(M)$

\[
\begin{align*}
    r_{i,i}^0 &= \left( \sum_{a \in \Sigma : \delta(q_i, a) = q_i} a \right)^* \\
    r_{i,j}^0 &= r_{i,i}^0 \left( \sum_{a \in \Sigma : \delta(q_i, a) = q_j} a \right) r_{j,j}^0 & \text{if } i \neq j \\
    r_{i,j}^k &= r_{i,j}^{k-1} + r_{i,k}^{k-1} r_{k,k}^{k-1} r_{k,j}^{k-1} & i \neq j \\
    r_{i,i}^k &= \left( r_{i,i}^{k-1} + r_{i,k}^{k-1} \cdot r_{k,k}^{k-1} \cdot r_{k,i}^{k-1} \right)^* \\
    r_{i,j} &= r_{i,j}^n.
\end{align*}
\]

The desired regular expression is: \( \text{reg-expression}(M) = \sum_{q_i \in F} r_{1,i} = \sum_{q_i \in F} r_{1,i}^n. \)
Example

\( r_{1,1}^0 = r_{2,2}^0 = b^* \) \hspace{1cm} \( r_{1,2}^0 = r_{2,1}^0 = b^* ab^* \)

\[
\begin{align*}
    r_{1,1}^1 &= (r_{1,1}^0 + r_{1,1}^0 r_{1,1}^0 r_{1,1}^0)^* = b^* \\
    r_{2,2}^1 &= (r_{2,2}^0 + r_{2,1}^0 r_{1,1}^0 r_{1,2}^0)^* = (b^* + b^* ab^* b^* b^* ab^*)^* = (b^* + ab^* a)^* \\
    r_{1,2}^1 &= r_{1,2}^0 + r_{1,1}^0 r_{1,1}^0 r_{1,2}^0 = b^* ab^* + b^* b^* ab^* = b^* ab^*. \\
    r_{2,1}^1 &= r_{2,1}^0 + r_{2,1}^0 r_{1,1}^0 r_{1,1}^0 = b^* ab^* \\
    r_{1,1}^2 &= (r_{1,1}^1 + r_{1,2}^1 r_{2,1}^1 r_{1,1}^1)^* = \cdots \\
    r_{2,2}^1 &= \cdots
\end{align*}
\]
Example

\[ r_{1,1}^0 = r_{2,2}^0 = b^* \]
\[ r_{1,2}^0 = r_{2,1}^0 = b^*ab^* \]

\[ r_{1,1}^1 = (r_{1,1}^0 + r_{1,1}^0 r_{1,1}^0 r_{1,1}^0)^* = b^* \]
\[ r_{2,2}^1 = (r_{2,2}^0 + r_{2,1}^0 r_{1,1}^0 r_{1,2}^0)^* = (b^* + b^*ab^*b^*b^*ab^*)^* = (b^* + ab^*a)^* \]
\[ r_{1,2}^1 = r_{1,2}^0 + r_{1,1}^0 r_{1,1}^0 r_{1,2}^0 = b^*ab^* + b^*b^*ab^* = b^*ab^*. \]
\[ r_{2,1}^1 = r_{2,1}^0 + r_{2,1}^0 r_{1,1}^0 r_{1,1}^0 = b^*ab^* \]
\[ r_{1,1}^2 = (r_{1,1}^1 + r_{1,2}^1 r_{2,2}^1 r_{2,1}^1)^* = \cdots \]
\[ r_{2,2}^1 = \cdots \]
Example

\( r_1^0 = r_2^0 = b^* \)  \( r_1^0 = r_2^0 = b^* ab^* \)

\[
\begin{align*}
  r_{1,1}^1 &= (r_{1,1}^0 + r_{1,1}^0 r_{1,1}^0 r_{1,1}^0)^* = b^* \\
  r_{2,2}^1 &= (r_{2,2}^0 + r_{2,1}^0 r_{1,1}^0 r_{1,1}^0)^* = (b^* + b^* ab^* b^* b^* ab^*)^* = (b^* + ab^* a)^* \\
  r_{1,2}^1 &= r_{1,2}^0 + r_{1,1}^0 r_{1,1}^0 r_{1,2}^0 = b^* ab^* + b^* b^* ab^* = b^* ab^*. \\
  r_{2,1}^1 &= r_{2,1}^0 + r_{2,1}^0 r_{1,1}^0 r_{1,1}^0 = b^* ab^* \\
  r_{1,1}^2 &= (r_{1,1}^1 + r_{1,2}^1 r_{2,1}^1 r_{2,1}^1)^* = \cdots \\
  r_{2,2}^1 &= \cdots
\end{align*}
\]
Example

\[ r_{1,1}^0 = r_{2,2}^0 = b^* \]
\[ r_{1,2}^0 = r_{2,1}^0 = b^* ab^* \]

\[ r_{1,1}^1 = (r_{1,1}^0 + r_{1,1}^0 r_{1,1}^0 r_{1,1}^0)^* = b^* \]
\[ r_{2,2}^1 = (r_{2,2}^0 + r_{2,1}^0 r_{1,1}^0 r_{1,2}^0)^* = (b^* + b^* ab^* b^* b^* ab^*)^* = (b^* + ab^* a)^* \]
\[ r_{1,2}^1 = r_{1,2}^0 + r_{1,1}^0 r_{1,1}^0 r_{1,2}^0 = b^* ab^* + b^* b^* ab^* = b^* ab^* . \]
\[ r_{2,1}^1 = r_{2,1}^0 + r_{2,1}^0 r_{1,1}^0 r_{2,1}^0 = b^* ab^* \]
\[ r_{1,1}^2 = (r_{1,1}^1 + r_{1,2}^1 r_{2,2}^1 r_{2,1}^1)^* = \cdots \]
\[ r_{2,2}^1 = \cdots \]
Example

\[ r_{1,1}^0 = r_{2,2}^0 = b^* \]
\[ r_{1,2}^0 = r_{2,1}^0 = b^* ab^* \]

\[ r_{1,1}^1 = (r_{1,1}^0 + r_{1,1}^0 r_{1,1}^0 r_{1,1}^0)^* = b^* \]
\[ r_{2,2}^1 = (r_{2,2}^0 + r_{2,1}^0 r_{1,1}^0 r_{1,1}^0)^* = (b^* + b^* ab^* b^* b^* ab^*)^* = (b^* + ab^* a)^* \]
\[ r_{1,2}^1 = r_{1,2}^0 + r_{1,1}^0 r_{1,1}^0 r_{1,2}^0 = b^* ab^* + b^* b^* ab^* = b^* ab^*. \]
\[ r_{2,1}^1 = r_{1,2}^0 + r_{2,1}^0 r_{1,1}^0 r_{1,1}^0 = b^* ab^* \]
\[ r_{1,1}^2 = (r_{1,1}^1 + r_{1,2}^1 r_{2,2}^1 r_{2,1}^1)^* = \cdots \]
\[ r_{2,2}^1 = \cdots \]
Example

\[ r_{1,1}^0 = r_{2,2}^0 = b^* \quad \quad r_{1,2}^0 = r_{2,1}^0 = b^* ab^* \]

\[ r_{1,1}^1 = (r_{1,1}^0 + r_{1,1}^0 r_{1,1}^0 r_{1,1}^0)^* = b^* \]
\[ r_{2,2}^1 = (r_{2,2}^0 + r_{2,1}^0 r_{1,1}^0 r_{1,2}^0)^* = (b^* + b^* ab^* b^* b^* ab^*)^* = (b^* + ab^* a)^* \]
\[ r_{1,2}^1 = r_{1,2}^0 + r_{1,1}^0 r_{1,1}^0 r_{1,2}^0 = b^* ab^* + b^* b^* ab^* = b^* ab^*. \]
\[ r_{2,1}^1 = r_{1,2}^0 + r_{2,1}^0 r_{1,1}^0 r_{1,1}^0 = b^* ab^* \]
\[ r_{1,1}^2 = (r_{1,1}^1 + r_{1,2}^1 r_{2,2}^1 r_{2,1}^1)^* = \cdots \]
\[ r_{2,2}^1 = \cdots \]
Example

\[ r_{1,1}^0 = r_{2,2}^0 = b^* \quad r_{1,2}^0 = r_{2,1}^0 = b^*ab^* \]

\[ r_{1,1}^1 = (r_{1,1}^0 + r_{1,1}^0 r_{1,1}^0 r_{1,1}^0)^* = b^* \]
\[ r_{2,2}^1 = (r_{2,2}^0 + r_{2,1}^0 r_{1,1}^0 r_{1,1}^0 r_{1,2}^0)^* = (b^* + b^*ab*b^*b^*ab^*)^* = (b^* + ab^*a)^* \]
\[ r_{1,2}^1 = r_{1,2}^0 + r_{1,1}^0 r_{1,1}^0 r_{1,2}^0 = b^*ab^* + b^*b^*ab^* = b^*ab^*. \]
\[ r_{2,1}^1 = r_{2,1}^0 + r_{2,1}^0 r_{1,1}^0 r_{1,1}^0 = b^*ab^* \]
\[ r_{1,1}^2 = (r_{1,1}^1 + r_{1,2}^1 r_{2,2}^1 r_{2,1}^1)^* = \cdots \]
\[ r_{2,2}^1 = \cdots \]
Correctness

Similar to that of Floyd-Warshall algorithms for shortest paths via induction.

The length of the regular expression can be exponential in the size of the original DFA.
THE END

... (for now)