Equivalence of DFA and Regular Expressions

CSCI 3130 Formal Languages and Automata Theory

Siu On CHAN
Fall 2018

Chinese University of Hong Kong
Three ways of doing it

\[ L = \{ x \in \Sigma^* \mid x \text{ ends in } 01 \} \]

\[ \Sigma = \{0, 1\} \]
They are equally powerful

DFA  NFA  regular expressions

regular languages
Examples: regular expression → NFA

\[ R_1 = 0 \]

\[ R_2 = 01 \]
$R_3 = 0+01$

$R_4 = (0+01)^*$
In general, how do we convert a regular expression to an NFA?

A regular expression over Σ is an expression formed by the following rules:

- The symbols ∅ and ε are regular expressions.
- Every symbol in Σ is a regular expression.
  - If Σ = {0, 1}, then 0 and 1 are both regular expressions.
- If $R$ and $S$ are regular expressions, so are $R + S$, $RS$ and $R^*$.
General method when $\Sigma = \{0, 1\}$

<table>
<thead>
<tr>
<th>Regular expression</th>
<th>NFA</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\emptyset$</td>
<td>$q_0$</td>
</tr>
<tr>
<td>$\varepsilon$</td>
<td>$q_0$</td>
</tr>
<tr>
<td>$0$</td>
<td>$q_0 \xrightarrow{0} q_1$</td>
</tr>
<tr>
<td>$1$</td>
<td>$q_0 \xrightarrow{1} q_1$</td>
</tr>
</tbody>
</table>
General method

regular expression $\Rightarrow$ NFA

$RS$

$R + S$

$R^*$
Roadmap

regular expressions

2-state GNFA

GNFA

NFA
First we simplify the NFA so that

- It has exactly one accepting state
- No arrows come into the start state
- No arrows go out of the accepting state
First we simplify the NFA so that

- It has **exactly one** accepting state
- No arrows come into the start state
- No arrows go out of the accepting state
Simplify the NFA

- It has exactly one accepting state.
- No arrows come into the start state.
- No arrows go out of the accepting state.
Simplify the NFA

- It has exactly one accepting state ✓
- No arrows come into the start state ✓
- No arrows go out of the accepting state ✓
A generalized NFA is an NFA whose transitions are labeled by regular expressions, like

![Diagram of a generalized NFA with states $q_0$, $q_1$, and $q_2$, transitions labeled by $\varepsilon + 10^*$, $0^*1$, and $01$.

$q_0 \rightarrow q_1 \rightarrow q_2$]
We will eliminate every state but the start and accepting states
State elimination

\[ \begin{align*}
q_0 &\xrightarrow{\varepsilon + 10^*} q_1 \\
q_1 &\xrightarrow{0^*1} q_2 \\
q_0 &\xrightarrow{01} q_2
\end{align*} \]

\[ \Downarrow \]

\[ \begin{align*}
(\varepsilon + 10^*)(0^*1)^*0^*11 &\xrightarrow{01} q_2
\end{align*} \]

\[ \Downarrow \]

\[ \begin{align*}
(\varepsilon + 10^*)(0^*1)^*0^*11 + 01 &\xrightarrow{01} q_2
\end{align*} \]
To eliminate state $q$, for every pair of states $(u, v)$ such that $u \rightarrow q \rightarrow v$

Replace $u \rightarrow q \rightarrow v$ by $u \rightarrow R_1 R_2^* R_3 + R_4 \rightarrow v$

Remember to do this even when $u = v$
A 2-state GNFA is the same as a regular expression $R$. 
After eliminating $q_1$:

Check: $0 \varepsilon (00 \varepsilon 1 + 1) \varepsilon = q_1 q_3$
Conversion example

After eliminating $q_1$: 

After eliminating $q_2$: 

Check: $0\ast 1(00\ast 1 + 1)^\ast = q_1 q_2 0 1$
Conversion example

After eliminating $q_1$:

After eliminating $q_2$:

Check:

$$0^*1(00^*1 + 1)^* = ?$$
All strings ending in 1
(0+1)*1

Yes
All strings ending in 1

(0+1)*1

0*1(00*1 + 1)*

= 0*1(0*1)*

Always ends in 1

Does every string ending in 1 have this form?

Yes