Equivalence of DFA and Regular Expressions

CSCI 3130 Formal Languages and Automata Theory

Siu On CHAN
Fall 2020

Chinese University of Hong Kong
Three ways of doing it

\[ L = \{ x \in \Sigma^* \mid x \text{ ends in } 01 \} \quad \Sigma = \{0, 1\} \]
They are equally powerful

DFA  NFA  regular expressions

regular languages
Examples: regular expression $\rightarrow$ NFA

$R_1 = 0$

$R_2 = 01$
Examples: regular expression → NFA

\[ R_3 = 0+01 \]

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

A regular expression over $\Sigma$ is an expression formed by the following rules:

- The symbols $\emptyset$ and $\varepsilon$ are regular expressions.
- Every symbol in $\Sigma$ is a regular expression.
  - If $\Sigma = \{0, 1\}$, 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

$RS$

$R + S$

$R^*$
Simplify the 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 $q_3$
- No arrows come into the start state $q_3$
- No arrows go out of the accepting state $q_3$
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
We will eliminate every state but the start and accepting states
State elimination

\[ \varepsilon + 10^* \rightarrow q_1 \rightarrow 0^*1 \rightarrow q_2 \]

\[ \downarrow \]

\[ (\varepsilon + 10^*)(0^*1)^*0^*11 \rightarrow q_2 \]

\[ \downarrow \]

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

Replace $u \!\!q\!\! v$ by $u \!\!R_1 \!\!R_2 \!\!R_3 \!\!+ \!\!R_4 \!\! v$

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

\[
\begin{align*}
q_0 & \xrightarrow{\varepsilon} q_1 \xrightarrow{1} q_2 \xrightarrow{\varepsilon} q_3 \\
q_0 & \xrightarrow{\varepsilon} q_2 \xrightarrow{0} q_1
\end{align*}
\]
Conversion example

After eliminating $q_1$: $q_0 \xrightarrow{\varepsilon} q_1 \xrightarrow{1} q_2 \xrightarrow{\varepsilon} q_3$

After eliminating $q_2$: $q_0 \xrightarrow{0 \cdot 1} q_2 \xrightarrow{\varepsilon} q_3$

$$00^*1 + 1$$

Check: $00^*1 + 1 = q_1 q_2 q_3$
Conversion example

After eliminating $q_1$:

After eliminating $q_2$:

Check: $0^*1(00^*1+1)^* \ ?$
Check your answer!

All strings ending in 1

$(0 + 1)^* 1$

Does every string ending in 1 have this form?

Yes
Check your answer!

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

$0^*1(00^*1 + 1)^*$

Always ends in 1

Does every string ending in 1 have this form?
Yes