Equivalence of DFA and Regular Expressions
CSCI 3130 Formal Languages and Automata Theory

Siu On CHAN

Chinese University of Hong Kong

Fall 2015
Three ways of doing it

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

DFA

NFA

\[(0 + 1)^*01\]

regular expressions
They are equally powerful

DFA

NFA

regular expressions

regular languages
Roadmap

regular expressions → NFA → NFA without ε → DFA

4/19
Examples: regular expression $\rightarrow$ NFA

\[ R_1 = 0 \quad \xrightarrow{} \quad q_0 \xrightarrow{0} q_1 \]

\[ R_2 = 01 \quad \xrightarrow{} \quad q_0 \xrightarrow{0} q_1 \xrightarrow{1} q_2 \]
Examples: regular expression → NFA

\[ R_3 = 0 + 01 \]

\[ R_4 = (0 + 01)^* \]
Regular expressions

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 \( a \) in \( \Sigma \) is a regular expression.
- If \( R \) and \( S \) are regular expressions, so are \( R + S \), \( RS \) and \( R^* \).
General method

Regular expression $\Rightarrow$ NFA

$\emptyset \rightarrow q_0$

$\varepsilon \rightarrow q_0$

$a \in \Sigma \rightarrow q_0 \xrightarrow{a} q_1$
General method

regular expression $\rightarrow$ NFA

$RS$

$R + S$

$R^*$
Roadmap

regular expressions → NFA

NFA → regular expressions

✓
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
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
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](image_url)
We will **eliminate** every state but the start and accepting states.
State elimination

\[ (\varepsilon + 10^*) (01)^* 0^* 11 + 01 \]
State elimination: general method

To **eliminate** state $q$, for every pair of states $(u, v)$

Replace

```
    u  ->q  v
      |    |
      |    R2|
      |      |
      |    |
      |    R3|
      |      |
      v
```

by

```
    u  -> q  v
      |      |
      |    R1  R2*  R3  +  R4 |
      |      |
      v
```

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

$\begin{align*}
q_0 & \xrightarrow{R} q_1 \\
\text{2-state GNFA} & \xrightarrow{} \text{GNFA} \\
\text{regular expressions} & \xrightarrow{} \text{NFA}
\end{align*}$
Conversion example

Eliminate $q_1$:
Conversion example

Eliminate $q_1$:

Eliminate $q_2$:
Conversion example

Eliminate $q_1$:

Eliminate $q_2$:

Check:

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

All strings ending in 1

(0 + 1)*1
All strings ending in $1$ have the form $(0 + 1)^*1$.

Does every string ending in $1$ have this form?

Yes