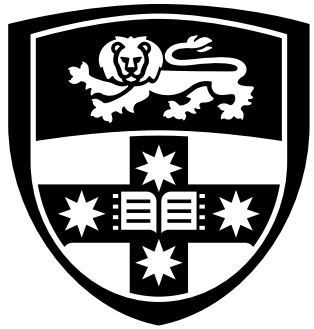
COMP2022: Formal Languages and Logic

2017, Semester 1, Week 3

Joseph Godbehere

Adapted from slides by A/Prof Kalina Yacef

March 21, 2017



THE UNIVERSITY OF
SYDNEY

Outline

- **Regular Expressions**
- Equivalence of FA and Regular Expressions
- Proving if a language is, *or is not*, regular

Regular Expressions

In arithmetic we use operations to build up expressions, describing numbers such as

$$(5 + 3) \times 4 \text{ (value is 32)}$$
$$4/2 - 1 \text{ (value is 1)}$$

With regular languages, we use regular operations to build up expressions describing languages such as

$(0 \mid 1)0^\star$ (strings starting with 0 or 1, then any number of 0's)

$(ab)^\star a$ (any number of ab's, followed by a)

Regular Expressions vs Finite Automata

- RE: algebraic description of the strings in a language
- FA: a machine-like description

- RE serve as the input language for many systems which process strings
  - Search commands
    - UNIX grep
    - Web browsers
    - Text editors
    - Text formatting systems
    - ...
  - Lexical-analyser generators (Lex, Flex, etc.)

- Like FA, RE define exactly the class of *regular languages*

Definition of a Regular Expression (RE)

- Basic expressions and the language they describe
  - If a is any symbol of the input alphabet, then a is a RE and its language $L(a) = \{a\}$
  - ε is a RE and $L(\varepsilon) = \{\varepsilon\}$
  - $\emptyset$ is a RE and $L(\emptyset) = \emptyset$

Definition of a Regular Expression (RE)

- Basic expressions and the language they describe
  - If a is any symbol of the input alphabet, then a is a RE and its language $L(a) = \{a\}$
  - ε is a RE and $L(\varepsilon) = \{\varepsilon\}$
  - $\emptyset$ is a RE and $L(\emptyset) = \emptyset$
- Operators on RE. If R and S are RE, then
  - Union: $R \mid S$ (sometimes denoted $R + S$) is a RE
  - Concatenation: RS (sometimes denoted $R \circ S$) is a RE
  - Star Closure: $R^\star$ is a RE

Definition of a Regular Expression (RE)

- Basic expressions and the language they describe
  - If a is any symbol of the input alphabet, then a is a RE and its language $L(a) = \{a\}$
  - ε is a RE and $L(\varepsilon) = \{\varepsilon\}$
  - $\emptyset$ is a RE and $L(\emptyset) = \emptyset$
- Operators on RE. If R and S are RE, then
  - Union: $R \mid S$ (sometimes denoted $R + S$) is a RE
  - Concatenation: RS (sometimes denoted $R \circ S$) is a RE
  - Star Closure: $R^\star$ is a RE
- Precedence of operators: $^\star, \circ, \mid$ to avoid excessive parentheses
  - e.g. $R \mid ST^\star = (R \mid (S(T^\star)))$, similar to how $8 + 21/3^2 = (8 + (21/(3^2)))$
  - Use parentheses to change the order of operations

Language of a Regular Expression

Let x, y be symbols from the input alphabet

$$
\begin{aligned}
L(x) &= \{x\} \\
L(\varepsilon) &= \{\varepsilon\} \\
L(\emptyset) &= \{\emptyset\} \\
L(x \mid y) &= \{x, y\} \\
L(x^\star) &= \{\varepsilon, x, xx, xxx, ...\} \\
L(xy) &= \{xy\}
\end{aligned}
$$

Operators on RE: Union $R \mid S$

Definition: $L(R \mid S) = L(R) \cup L(S)$

Operators on RE: Union $R \mid S$

Definition: $L(R \mid S) = L(R) \cup L(S)$

$L(a \mid b) =$

Operators on RE: Union $R \mid S$

Definition: $L(R \mid S) = L(R) \cup L(S)$

$$L(a \mid b) = L(a) \cup L(b) = \{a\} \cup \{b\} = \{a, b\}$$

$$L(b \mid a) =$$

Operators on RE: Union $R \mid S$

Definition: $L(R \mid S) = L(R) \cup L(S)$

$$L(a \mid b) = L(a) \cup L(b) = \{a\} \cup \{b\} = \{a, b\}$$
$$L(b \mid a) = L(b) \cup L(a) = \{b\} \cup \{a\} = \{a, b\} = L(a \mid b)$$
$$L((a \mid b) \mid c) =$$

Operators on RE: Union $R \mid S$

Definition: $L(R \mid S) = L(R) \cup L(S)$

$$L(a \mid b) = L(a) \cup L(b) = \{a\} \cup \{b\} = \{a, b\}$$
$$L(b \mid a) = L(b) \cup L(a) = \{b\} \cup \{a\} = \{a, b\} = L(a \mid b)$$
$$L((a \mid b) \mid c) = L(a \mid b) \cup L(c)$$
$$= $$

Operators on RE: Union $R \mid S$

Definition: $L(R \mid S) = L(R) \cup L(S)$

$$
\begin{aligned}
L(a \mid b) &= L(a) \cup L(b) = \{a\} \cup \{b\} = \{a, b\} \\
L(b \mid a) &= L(b) \cup L(a) = \{b\} \cup \{a\} = \{a, b\} = L(a \mid b) \\
L((a \mid b) \mid c) &= L(a \mid b) \cup L(c) \\
&= \{a, b\} \cup \{c\} \\
&=
\end{aligned}
$$

Operators on RE: Union $R \mid S$

Definition: $L(R \mid S) = L(R) \cup L(S)$

$$
\begin{aligned}
L(a \mid b) &= L(a) \cup L(b) = \{a\} \cup \{b\} = \{a, b\} \\
L(b \mid a) &= L(b) \cup L(a) = \{b\} \cup \{a\} = \{a, b\} = L(a \mid b) \\
L((a \mid b) \mid c) &= L(a \mid b) \cup L(c) \\
&= \{a, b\} \cup \{c\} \\
&= \{a, b, c\} \\
L(a \mid (b \mid c)) &=
\end{aligned}
$$

Operators on RE: Union $R \mid S$

Definition: $L(R \mid S) = L(R) \cup L(S)$

$$
\begin{aligned}
L(a \mid b) &= L(a) \cup L(b) = \{a\} \cup \{b\} = \{a, b\} \\
L(b \mid a) &= L(b) \cup L(a) = \{b\} \cup \{a\} = \{a, b\} = L(a \mid b) \\
L((a \mid b) \mid c) &= L(a \mid b) \cup L(c) \\
&= \{a, b\} \cup \{c\} \\
&= \{a, b, c\} \\
L(a \mid (b \mid c)) &= L(a) \cup L(b \mid c) \\
&=
\end{aligned}
$$

Operators on RE: Union $R \mid S$

Definition: $L(R \mid S) = L(R) \cup L(S)$

$$
\begin{aligned}
L(a \mid b) &= L(a) \cup L(b) = \{a\} \cup \{b\} = \{a, b\} \\
L(b \mid a) &= L(b) \cup L(a) = \{b\} \cup \{a\} = \{a, b\} = L(a \mid b) \\
L((a \mid b) \mid c) &= L(a \mid b) \cup L(c) \\
&= \{a, b\} \cup \{c\} \\
&= \{a, b, c\} \\
L(a \mid (b \mid c)) &= L(a) \cup L(b \mid c) \\
&= \{a\} \cup \{b, c\} \\
&=
\end{aligned}
$$

Operators on RE: Union $R \mid S$

Definition: $L(R \mid S) = L(R) \cup L(S)$

$$
\begin{aligned}
L(a \mid b) &= L(a) \cup L(b) = \{a\} \cup \{b\} = \{a, b\} \\
L(b \mid a) &= L(b) \cup L(a) = \{b\} \cup \{a\} = \{a, b\} = L(a \mid b) \\
L((a \mid b) \mid c) &= L(a \mid b) \cup L(c) \\
&= \{a, b\} \cup \{c\} \\
&= \{a, b, c\} \\
L(a \mid (b \mid c)) &= L(a) \cup L(b \mid c) \\
&= \{a\} \cup \{b, c\} \\
&= \{a, b, c\}
\end{aligned}
$$

Operators on RE: Union $R \mid S$

Definition: $L(R \mid S) = L(R) \cup L(S)$

$$
\begin{aligned}
L(a \mid b) &= L(a) \cup L(b) = \{a\} \cup \{b\} = \{a, b\} \\
L(b \mid a) &= L(b) \cup L(a) = \{b\} \cup \{a\} = \{a, b\} = L(a \mid b) \\
L((a \mid b) \mid c) &= L(a \mid b) \cup L(c) \\
&= \{a, b\} \cup \{c\} \\
&= \{a, b, c\} \\
L(a \mid (b \mid c)) &= L(a) \cup L(b \mid c) \\
&= \{a\} \cup \{b, c\} \\
&= \{a, b, c\}
\end{aligned}
$$

Because it is associative we can write $L(a \mid b \mid c)$

Operators on RE: Concatenation RS

Definition: $L(RS) = \{rs \mid r \in L(R) \text{ and } s \in L(S)\}$

Operators on RE: Concatenation RS

Definition: $L(RS) = \{rs \mid r \in L(R) \text{ and } s \in L(S)\}$

$L(ab) =$

Operators on RE: Concatenation RS

Definition: $L(RS) = \{rs \mid r \in L(R) \text{ and } s \in L(S)\}$

$L(ab) = \{rs \mid r \in L(a) \text{ and } s \in L(b)\} = \{ab\}$

$L(ba) =$

Operators on RE: Concatenation RS

Definition: $L(RS) = \{rs \mid r \in L(R) \text{ and } s \in L(S)\}$

$$L(ab) = \{rs \mid r \in L(a) \text{ and } s \in L(b)\} = \{ab\}$$
$$L(ba) = \{rs \mid r \in L(b) \text{ and } s \in L(a)\} = \{ba\}$$
$$L((ab)c) =$$

Operators on RE: Concatenation RS

Definition: $L(RS) = \{rs \mid r \in L(R) \text{ and } s \in L(S)\}$

$$L(ab) = \{rs \mid r \in L(a) \text{ and } s \in L(b)\} = \{ab\}$$
$$L(ba) = \{rs \mid r \in L(b) \text{ and } s \in L(a)\} = \{ba\}$$
$$L((ab)c) = \{rs \mid r \in L(ab) \text{ and } s \in L(c)\}$$
$$=$$

Operators on RE: Concatenation RS

Definition: $L(RS) = \{rs \mid r \in L(R) \text{ and } s \in L(S)\}$

$$
\begin{aligned}
L(ab) &= \{rs \mid r \in L(a) \text{ and } s \in L(b)\} = \{ab\} \\
L(ba) &= \{rs \mid r \in L(b) \text{ and } s \in L(a)\} = \{ba\} \\
L((ab)c) &= \{rs \mid r \in L(ab) \text{ and } s \in L(c)\} \\
&= \{rs \mid r \in \{ab\} \text{ and } s \in \{c\}\} \\
&=
\end{aligned}
$$

Operators on RE: Concatenation RS

Definition: $L(RS) = \{rs \mid r \in L(R) \text{ and } s \in L(S)\}$

$$
\begin{aligned}
L(ab) &= \{rs \mid r \in L(a) \text{ and } s \in L(b)\} = \{ab\} \\
L(ba) &= \{rs \mid r \in L(b) \text{ and } s \in L(a)\} = \{ba\} \\
L((ab)c) &= \{rs \mid r \in L(ab) \text{ and } s \in L(c)\} \\
&= \{rs \mid r \in \{ab\} \text{ and } s \in \{c\}\} \\
&= \{abc\} \\
L(a(bc)) &=
\end{aligned}
$$

Operators on RE: Concatenation RS

Definition: $L(RS) = \{rs \mid r \in L(R) \text{ and } s \in L(S)\}$

$$
\begin{aligned}
L(ab) &= \{rs \mid r \in L(a) \text{ and } s \in L(b)\} = \{ab\} \\
L(ba) &= \{rs \mid r \in L(b) \text{ and } s \in L(a)\} = \{ba\} \\
L((ab)c) &= \{rs \mid r \in L(ab) \text{ and } s \in L(c)\} \\
&= \{rs \mid r \in \{ab\} \text{ and } s \in \{c\}\} \\
&= \{abc\} \\
L(a(bc)) &= \{rs \mid r \in L(a) \text{ and } s \in L(bc)\} \\
&=
\end{aligned}
$$

Operators on RE: Concatenation RS

Definition: $L(RS) = \{rs \mid r \in L(R) \text{ and } s \in L(S)\}$

$$
\begin{aligned}
L(ab) &= \{rs \mid r \in L(a) \text{ and } s \in L(b)\} = \{ab\} \\
L(ba) &= \{rs \mid r \in L(b) \text{ and } s \in L(a)\} = \{ba\} \\
L((ab)c) &= \{rs \mid r \in L(ab) \text{ and } s \in L(c)\} \\
&= \{rs \mid r \in \{ab\} \text{ and } s \in \{c\}\} \\
&= \{abc\} \\
L(a(bc)) &= \{rs \mid r \in L(a) \text{ and } s \in L(bc)\} \\
&= \{rs \mid r \in \{a\} \text{ and } s \in \{bc\}\} \\
&=
\end{aligned}
$$

Operators on RE: Concatenation RS

Definition: $L(RS) = \{rs \mid r \in L(R) \text{ and } s \in L(S)\}$

$$
\begin{aligned}
L(ab) &= \{rs \mid r \in L(a) \text{ and } s \in L(b)\} = \{ab\} \\
L(ba) &= \{rs \mid r \in L(b) \text{ and } s \in L(a)\} = \{ba\} \\
L((ab)c) &= \{rs \mid r \in L(ab) \text{ and } s \in L(c)\} \\
&= \{rs \mid r \in \{ab\} \text{ and } s \in \{c\}\} \\
&= \{abc\} \\
L(a(bc)) &= \{rs \mid r \in L(a) \text{ and } s \in L(bc)\} \\
&= \{rs \mid r \in \{a\} \text{ and } s \in \{bc\}\} \\
&= \{abc\}
\end{aligned}
$$

Operators on RE: Concatenation RS

Definition: $L(RS) = \{rs \mid r \in L(R) \text{ and } s \in L(S)\}$

$$
\begin{aligned}
L(ab) &= \{rs \mid r \in L(a) \text{ and } s \in L(b)\} = \{ab\} \\
L(ba) &= \{rs \mid r \in L(b) \text{ and } s \in L(a)\} = \{ba\} \\
L((ab)c) &= \{rs \mid r \in L(ab) \text{ and } s \in L(c)\} \\
&= \{rs \mid r \in \{ab\} \text{ and } s \in \{c\}\} \\
&= \{abc\} \\
L(a(bc)) &= \{rs \mid r \in L(a) \text{ and } s \in L(bc)\} \\
&= \{rs \mid r \in \{a\} \text{ and } s \in \{bc\}\} \\
&= \{abc\}
\end{aligned}
$$

Because it is associative we can write $L(abc)$

Union and Concatenation

$$L((a \mid b)c) =$$

$$L((ab) \mid c) =$$

Union and Concatenation

$$L((a \mid b)c) = \{rs \mid r \in L(a \mid b) \text{ and } s \in L(c)\}$$
$$=$$

$$L((ab) \mid c) =$$

Union and Concatenation

$$\begin{aligned} L((a \mid b)c) &= \{rs \mid r \in L(a \mid b) \text{ and } s \in L(c)\} \\ &= \{rs \mid r \in (\{a\} \cup \{b\}) \text{ and } s \in \{c\}\} \\ &= \end{aligned}$$

$$L((ab) \mid c) =$$

Union and Concatenation

$$\begin{aligned} L((a \mid b)c) &= \{rs \mid r \in L(a \mid b) \text{ and } s \in L(c)\} \\ &= \{rs \mid r \in (\{a\} \cup \{b\}) \text{ and } s \in \{c\}\} \\ &= \{rs \mid r \in \{a, b\} \text{ and } s \in \{c\}\} \\ &= \end{aligned}$$

$$L((ab) \mid c) =$$

Union and Concatenation

$$\begin{aligned} L((a \mid b)c) &= \{rs \mid r \in L(a \mid b) \text{ and } s \in L(c)\} \\ &= \{rs \mid r \in (\{a\} \cup \{b\}) \text{ and } s \in \{c\}\} \\ &= \{rs \mid r \in \{a, b\} \text{ and } s \in \{c\}\} \\ &= \{ac, bc\} \end{aligned}$$

$$L((ab) \mid c) =$$

Union and Concatenation

$$\begin{aligned} L((a \mid b)c) &= \{rs \mid r \in L(a \mid b) \text{ and } s \in L(c)\} \\ &= \{rs \mid r \in (\{a\} \cup \{b\}) \text{ and } s \in \{c\}\} \\ &= \{rs \mid r \in \{a, b\} \text{ and } s \in \{c\}\} \\ &= \{ac, bc\} \end{aligned}$$

$$\begin{aligned} L((ab) \mid c) &= L(ab) \cup L(c) \\ &= \end{aligned}$$

Union and Concatenation

$$\begin{aligned} L((a \mid b)c) &= \{rs \mid r \in L(a \mid b) \text{ and } s \in L(c)\} \\ &= \{rs \mid r \in (\{a\} \cup \{b\}) \text{ and } s \in \{c\}\} \\ &= \{rs \mid r \in \{a, b\} \text{ and } s \in \{c\}\} \\ &= \{ac, bc\} \end{aligned}$$

$$\begin{aligned} L((ab) \mid c) &= L(ab) \cup L(c) \\ &= \{rs \mid r \in L(a) \text{ and } s \in L(b)\} \cup L(c) \\ &= \end{aligned}$$

Union and Concatenation

$$\begin{aligned} L((a \mid b)c) &= \{rs \mid r \in L(a \mid b) \text{ and } s \in L(c)\} \\ &= \{rs \mid r \in (\{a\} \cup \{b\}) \text{ and } s \in \{c\}\} \\ &= \{rs \mid r \in \{a, b\} \text{ and } s \in \{c\}\} \\ &= \{ac, bc\} \end{aligned}$$

$$\begin{aligned} L((ab) \mid c) &= L(ab) \cup L(c) \\ &= \{rs \mid r \in L(a) \text{ and } s \in L(b)\} \cup L(c) \\ &= \{rs \mid r \in \{a\} \text{ and } s \in \{b\}\} \cup \{c\} \\ &= \end{aligned}$$

Union and Concatenation

$$\begin{aligned} L((a \mid b)c) &= \{rs \mid r \in L(a \mid b) \text{ and } s \in L(c)\} \\ &= \{rs \mid r \in (\{a\} \cup \{b\}) \text{ and } s \in \{c\}\} \\ &= \{rs \mid r \in \{a, b\} \text{ and } s \in \{c\}\} \\ &= \{ac, bc\} \end{aligned}$$

$$\begin{aligned} L((ab) \mid c) &= L(ab) \cup L(c) \\ &= \{rs \mid r \in L(a) \text{ and } s \in L(b)\} \cup L(c) \\ &= \{rs \mid r \in \{a\} \text{ and } s \in \{b\}\} \cup \{c\} \\ &= \{ab\} \cup \{c\} \\ &= \{ab, c\} \end{aligned}$$

Operators on RE: Star closure $R^\star$

$R^\star$ means 0 or more occurrences of R

$L(R^\star) = \{\varepsilon\} \cup L(R) \cup L(RR) \cup L(RRR) \cup ...$

$$L(0^\star) = \{\varepsilon, 0, 00, 000, ...\}$$
$$L((01)^\star) = \{\varepsilon, 01, 0101, 010101, ...\}$$
$$L(\emptyset^\star) =$$

Operators on RE: Star closure $R^\star$

$R^\star$ means 0 or more occurrences of R

$L(R^\star) = \{\varepsilon\} \cup L(R) \cup L(RR) \cup L(RRR) \cup ...$

$$L(0^\star) = \{\varepsilon, 0, 00, 000, ...\}$$
$$L((01)^\star) = \{\varepsilon, 01, 0101, 010101, ...\}$$
$$L(\emptyset^\star) = \{\varepsilon\} \cup L(\emptyset) \cup ... = \{\varepsilon\}$$
$$L(a \mid bc^\star) =$$

Operators on RE: Star closure $R^\star$

$R^\star$ means 0 or more occurrences of R

$L(R^\star) = \{\varepsilon\} \cup L(R) \cup L(RR) \cup L(RRR) \cup ...$

$$L(0^\star) = \{\varepsilon, 0, 00, 000, ...\}$$
$$L((01)^\star) = \{\varepsilon, 01, 0101, 010101, ...\}$$
$$L(\emptyset^\star) = \{\varepsilon\} \cup L(\emptyset) \cup ... = \{\varepsilon\}$$
$$L(a \mid bc^\star) = \{a, b, bc, bcc, bccc, ...\}$$
$$L(a \mid (bc)^\star) =$$

Operators on RE: Star closure $R^\star$

$R^\star$ means 0 or more occurrences of R

$L(R^\star) = \{\varepsilon\} \cup L(R) \cup L(RR) \cup L(RRR) \cup ...$

$$
\begin{aligned}
L(0^\star) &= \{\varepsilon, 0, 00, 000, ...\} \\
L((01)^\star) &= \{\varepsilon, 01, 0101, 010101, ...\} \\
L(\emptyset^\star) &= \{\varepsilon\} \cup L(\emptyset) \cup ... = \{\varepsilon\} \\
L(a \mid bc^\star) &= \{a, b, bc, bcc, bccc, ...\} \\
L(a \mid (bc)^\star) &= \{a, \varepsilon, bc, bcbc, bcbcbc, ...\} \\
L((a \mid b)c^\star) &=
\end{aligned}
$$

Operators on RE: Star closure $R^\star$

$R^\star$ means 0 or more occurrences of R

$L(R^\star) = \{\varepsilon\} \cup L(R) \cup L(RR) \cup L(RRR) \cup ...$

$$
\begin{aligned}
L(0^\star) &= \{\varepsilon, 0, 00, 000, ...\} \\
L((01)^\star) &= \{\varepsilon, 01, 0101, 010101, ...\} \\
L(\emptyset^\star) &= \{\varepsilon\} \cup L(\emptyset) \cup ... = \{\varepsilon\} \\
L(a \mid bc^\star) &= \{a, b, bc, bcc, bccc, ...\} \\
L(a \mid (bc)^\star) &= \{a, \varepsilon, bc, bcbc, bcbcbc, ...\} \\
L((a \mid b)c^\star) &= \{a, b, ac, bc, acc, bcc, accc, bccc, ...\}
\end{aligned}
$$

Some properties of RE

Let R, S, T be regular expressions

- Union properties:
  - $R \mid S = S \mid R$ (commutative)
  - $R \mid \emptyset = \emptyset \mid R = R$
  - $R \mid R = R$
  - $(R \mid S) \mid T = R \mid (S \mid T)$ (associative)
- Concatenation properties:
- Union and concatenation are *distributive* when combined:

Some properties of RE

Let R, S, T be regular expressions

- Union properties:
  - $R \mid S = S \mid R$ (commutative)
  - $R \mid \emptyset = \emptyset \mid R = R$
  - $R \mid R = R$
  - $(R \mid S) \mid T = R \mid (S \mid T)$ (associative)
- Concatenation properties:
  - $R\emptyset = \emptyset R = \emptyset$
  - $R\varepsilon = \varepsilon R = R$
  - $(RS)T = R(ST)$ (associative)
- Union and concatenation are *distributive* when combined:

Some properties of RE

Let R, S, T be regular expressions

- Union properties:
  - $R \mid S = S \mid R$ (commutative)
  - $R \mid \emptyset = \emptyset \mid R = R$
  - $R \mid R = R$
  - $(R \mid S) \mid T = R \mid (S \mid T)$ (associative)
- Concatenation properties:
  - $R\emptyset = \emptyset R = \emptyset$
  - $R\varepsilon = \varepsilon R = R$
  - $(RS)T = R(ST)$ (associative)
- Union and concatenation are *distributive* when combined:
  - $R(S \mid T) = RS \mid RT$
  - $(S \mid T)R = SR \mid TR$

Some properties of RE

Let R, S, T be regular expressions

- Star Closure properties:
  - $\emptyset^\star = \varepsilon^\star = \varepsilon$

Some properties of RE

Let R, S, T be regular expressions

- Star Closure properties:
  - $\emptyset^\star = \varepsilon^\star = \varepsilon$
  - $R^\star = R^\star R^\star = (R^\star)^\star = R \mid R^\star$

Some properties of RE

Let R, S, T be regular expressions

- Star Closure properties:
  - $\emptyset^\star = \varepsilon^\star = \varepsilon$
  - $R^\star = R^\star R^\star = (R^\star)^\star = R \mid R^\star$
  - $R^\star = \varepsilon \mid R^\star = (\varepsilon \mid R)^\star = (\varepsilon \mid R)R^\star = \varepsilon \mid RR^\star$

Some properties of RE

Let R, S, T be regular expressions

- Star Closure properties:
  - $\emptyset^\star = \varepsilon^\star = \varepsilon$
  - $R^\star = R^\star R^\star = (R^\star)^\star = R \mid R^\star$
  - $R^\star = \varepsilon \mid R^\star = (\varepsilon \mid R)^\star = (\varepsilon \mid R)R^\star = \varepsilon \mid RR^\star$
  - $RR^\star = R^\star R$

Some properties of RE

Let R, S, T be regular expressions

- Star Closure properties:
  - $\emptyset^\star = \varepsilon^\star = \varepsilon$
  - $R^\star = R^\star R^\star = (R^\star)^\star = R \mid R^\star$
  - $R^\star = \varepsilon \mid R^\star = (\varepsilon \mid R)^\star = (\varepsilon \mid R)R^\star = \varepsilon \mid RR^\star$
  - $RR^\star = R^\star R$
  - $(R \mid S)^\star = (R^\star \mid S^\star)^\star = (R^\star S^\star)^\star = (R^\star S)^\star R^\star = R^\star(SR^\star)^\star$

Some properties of RE

Let R, S, T be regular expressions

- Star Closure properties:
  - $\emptyset^\star = \varepsilon^\star = \varepsilon$
  - $R^\star = R^\star R^\star = (R^\star)^\star = R \mid R^\star$
  - $R^\star = \varepsilon \mid R^\star = (\varepsilon \mid R)^\star = (\varepsilon \mid R)R^\star = \varepsilon \mid RR^\star$
  - $RR^\star = R^\star R$
  - $(R \mid S)^\star = (R^\star \mid S^\star)^\star = (R^\star S^\star)^\star = (R^\star S)^\star R^\star = R^\star(SR^\star)^\star$
  - $R(SR)^\star = (RS)^\star R$

Some properties of RE

Let R, S, T be regular expressions

- Star Closure properties:
  - $\emptyset^\star = \varepsilon^\star = \varepsilon$
  - $R^\star = R^\star R^\star = (R^\star)^\star = R \mid R^\star$
  - $R^\star = \varepsilon \mid R^\star = (\varepsilon \mid R)^\star = (\varepsilon \mid R)R^\star = \varepsilon \mid RR^\star$
  - $RR^\star = R^\star R$
  - $(R \mid S)^\star = (R^\star \mid S^\star)^\star = (R^\star S^\star)^\star = (R^\star S)^\star R^\star = R^\star(SR^\star)^\star$
  - $R(SR)^\star = (RS)^\star R$
  - $(R^\star S)^\star = \varepsilon \mid (R \mid S)^\star S$

Some properties of RE

Let R, S, T be regular expressions

- Star Closure properties:
  - $\emptyset^\star = \varepsilon^\star = \varepsilon$
  - $R^\star = R^\star R^\star = (R^\star)^\star = R \mid R^\star$
  - $R^\star = \varepsilon \mid R^\star = (\varepsilon \mid R)^\star = (\varepsilon \mid R)R^\star = \varepsilon \mid RR^\star$
  - $RR^\star = R^\star R$
  - $(R \mid S)^\star = (R^\star \mid S^\star)^\star = (R^\star S^\star)^\star = (R^\star S)^\star R^\star = R^\star(SR^\star)^\star$
  - $R(SR)^\star = (RS)^\star R$
  - $(R^\star S)^\star = \varepsilon \mid (R \mid S)^\star S$
  - $(RS^\star)^\star = \varepsilon \mid R(R \mid S)^\star$

Examples of RE

- All possible strings which can be formed with a, b, or c
- Strings over $\{0, 1\}$ which end with 01
- Traffic lights?
- Identifiers for Java programs?

Examples of RE

- All possible strings which can be formed with a, b, or c
  $(a \mid b \mid c)^\star$
- Strings over $\{0, 1\}$ which end with 01
- Traffic lights?
- Identifiers for Java programs?

Examples of RE

- All possible strings which can be formed with a, b, or c
  $(a \mid b \mid c)^\star$
- Strings over $\{0, 1\}$ which end with 01
  $(0 \mid 1)^\star 01$
- Traffic lights?
- Identifiers for Java programs?

Examples of RE

- All possible strings which can be formed with a, b, or c
  $(a \mid b \mid c)^\star$
- Strings over $\{0, 1\}$ which end with 01
  $(0 \mid 1)^\star 01$
- Traffic lights?
  $(red \mid red \circ amber \mid amber \mid green)$
- Identifiers for Java programs?

Examples of RE

- All possible strings which can be formed with a, b, or c
  $(a \mid b \mid c)^\star$
- Strings over $\{0, 1\}$ which end with 01
  $(0 \mid 1)^\star 01$
- Traffic lights?
  $(red \mid red \circ amber \mid amber \mid green)$
- Identifiers for Java programs?
  $(letter \mid \$ \mid \_)(letter \mid digit \mid \$ \mid \_)^\star$

Examples of RE

- $a^\star ba^\star$ represents
- $((a \mid b)(a \mid b))^\star$ represents
- $ab^\star a \mid ba^\star b \mid a \mid b$ represents
- $a^\star\emptyset$ represents

Examples of RE

- $a^\star ba^\star$ represents
  Strings over $\{a, b\}$ with exactly one b
- $((a \mid b)(a \mid b))^\star$ represents
- $ab^\star a \mid ba^\star b \mid a \mid b$ represents
- $a^\star\emptyset$ represents

Examples of RE

- $a^\star ba^\star$ represents
  Strings over $\{a, b\}$ with exactly one b
- $((a \mid b)(a \mid b))^\star$ represents
  Strings over $\{a, b\}$ with even length
- $ab^\star a \mid ba^\star b \mid a \mid b$ represents
- $a^\star \emptyset$ represents

Examples of RE

- $a^\star ba^\star$ represents
  Strings over $\{a, b\}$ with exactly one b
- $((a \mid b)(a \mid b))^\star$ represents
  Strings over $\{a, b\}$ with even length
- $ab^\star a \mid ba^\star b \mid a \mid b$ represents
  Strings of a's starting and ending with b, or strings of b's starting and ending with a, or a single a or b
- $a^\star\emptyset$ represents

Examples of RE

- $a^\star ba^\star$ represents
  Strings over $\{a, b\}$ with exactly one b
- $((a \mid b)(a \mid b))^\star$ represents
  Strings over $\{a, b\}$ with even length
- $ab^\star a \mid ba^\star b \mid a \mid b$ represents
  Strings of a's starting and ending with b, or strings of b's starting and ending with a, or a single a or b
- $a^\star\emptyset$ represents
  The empty language, $\emptyset$

Outline

- Regular Expressions
- **Equivalence of FA and Regular Expressions**
- Proving if a language is, *or is not*, regular

Equivalence of RE and FA

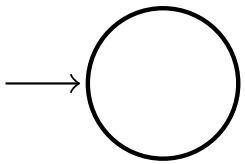
Theorem:
A language is regular if and only if a regular expression describes it.

Proof:
Show the equivalence of RE and FA (RE ⇔ FA)

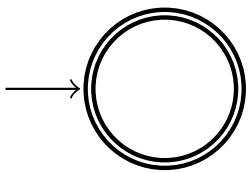
1. RE ⇒ FA:
   Show that for each RE, there exists an NFA which recognises the same language
2. FA ⇒ RE:
   Show that for each NFA, there exists a RE which recognises the same language

From RE to FA: atomic cases

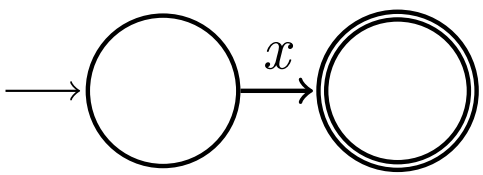
Automaton for $\emptyset$



Automaton for ε



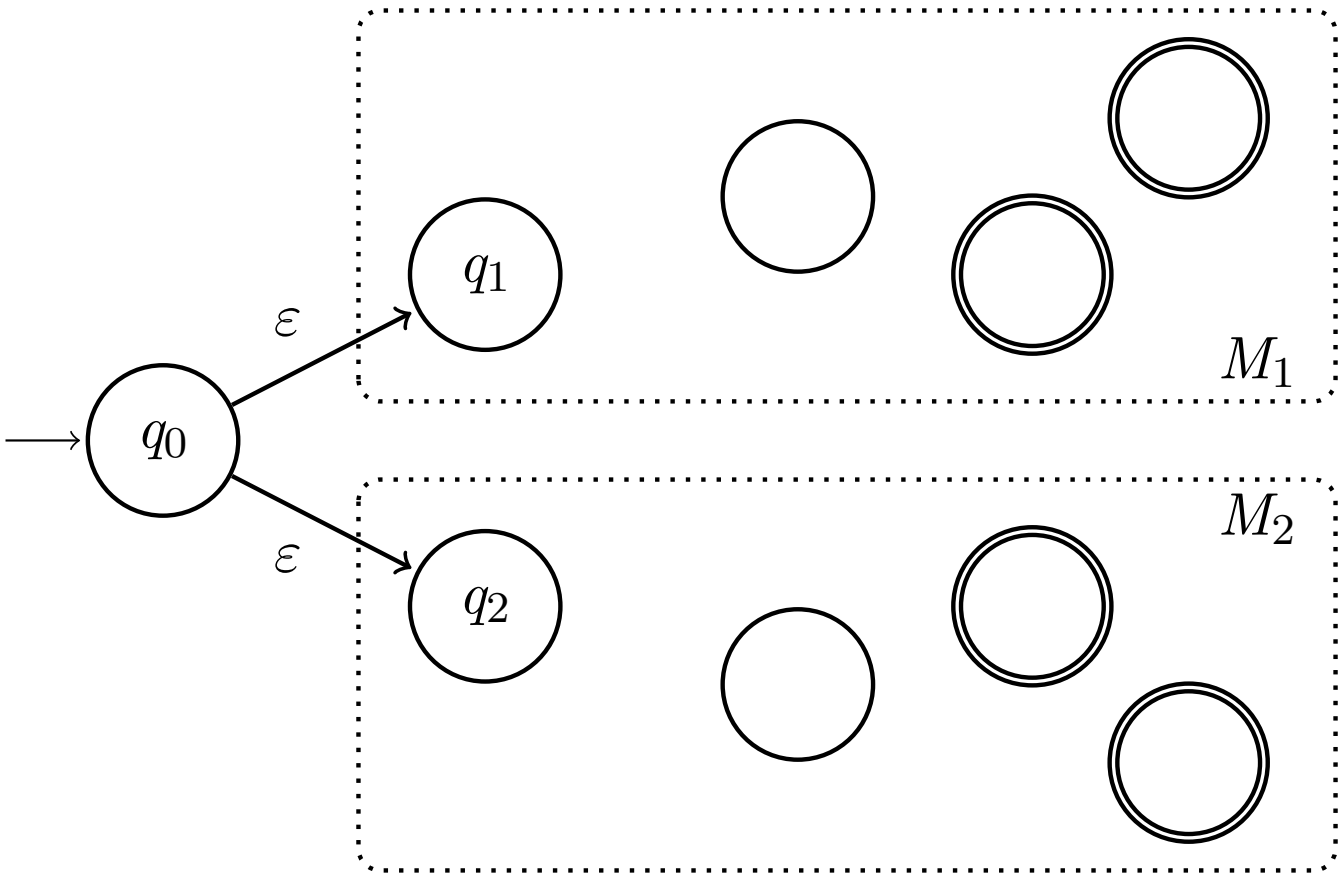
Automaton for $x \in \Sigma$



NFA for regular operations: Union

Let M_1 and M_2 be automata recognising $L(R_1)$ and $L(R_2)$

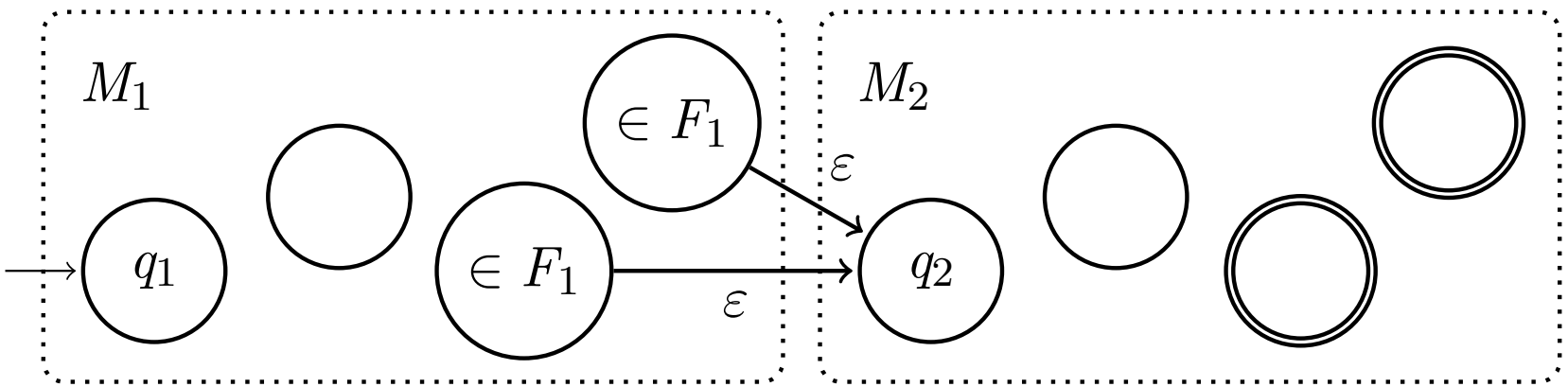
Then an automaton M for $R_1 \mid R_2$ is:



NFA for regular operations: Concatenation

Let M_1 and M_2 be automata recognising $L(R_1)$ and $L(R_2)$

Then an automaton M for $R_1 \circ R_2$ is:

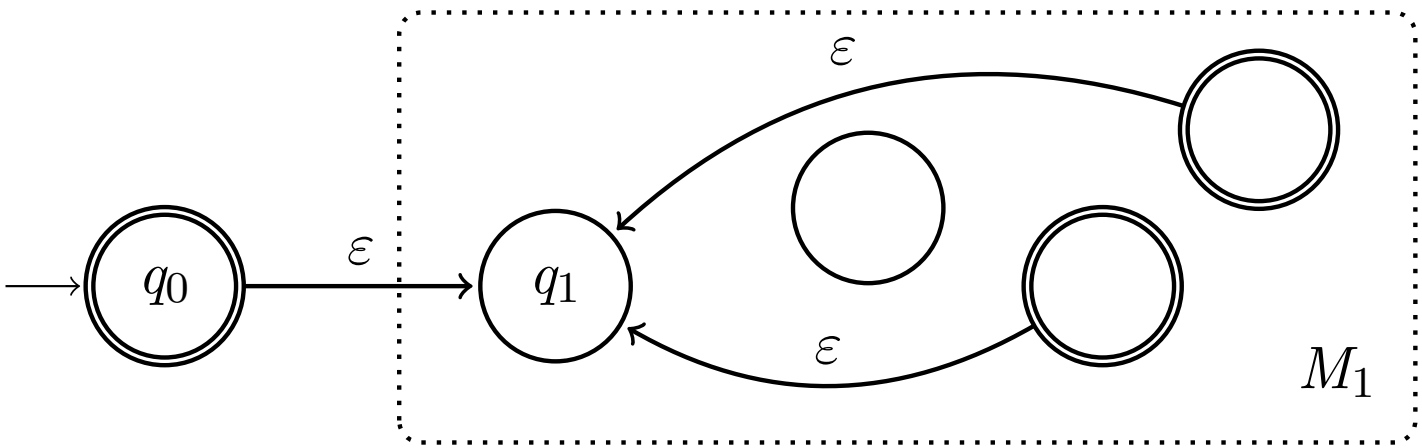


Reminder: the accept states of M_1 are *not* accept states in M

NFA for regular operations: Star closure

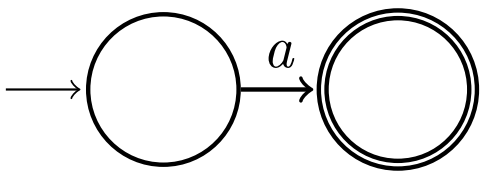
Let M_1 be an automaton recognising $L(R_1)$

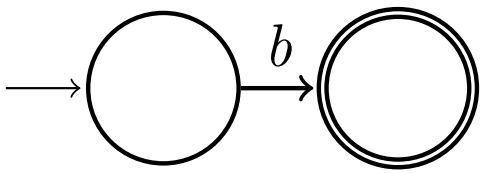
Then an automaton M for $R_1^\star$ is:

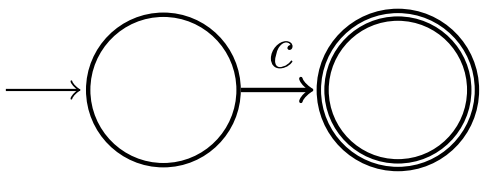


Example: $a \mid bc^\star$: from RE to NFA

1. Construct automata for the atomic regular expressions a, b, c

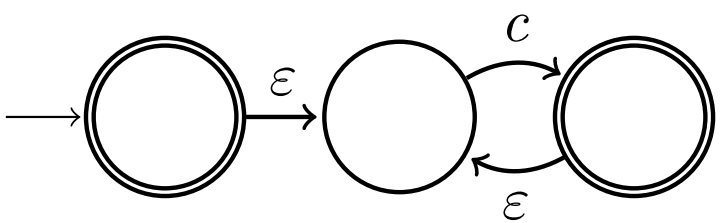






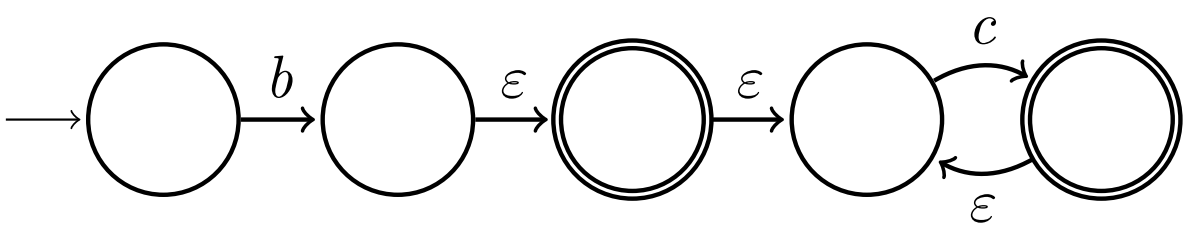
Example: $a \mid bc^\star$: from RE to NFA

1. Construct automata for the atomic regular expressions a, b, c
2. Use the Star Closure operation to find an automaton for $c^\star$



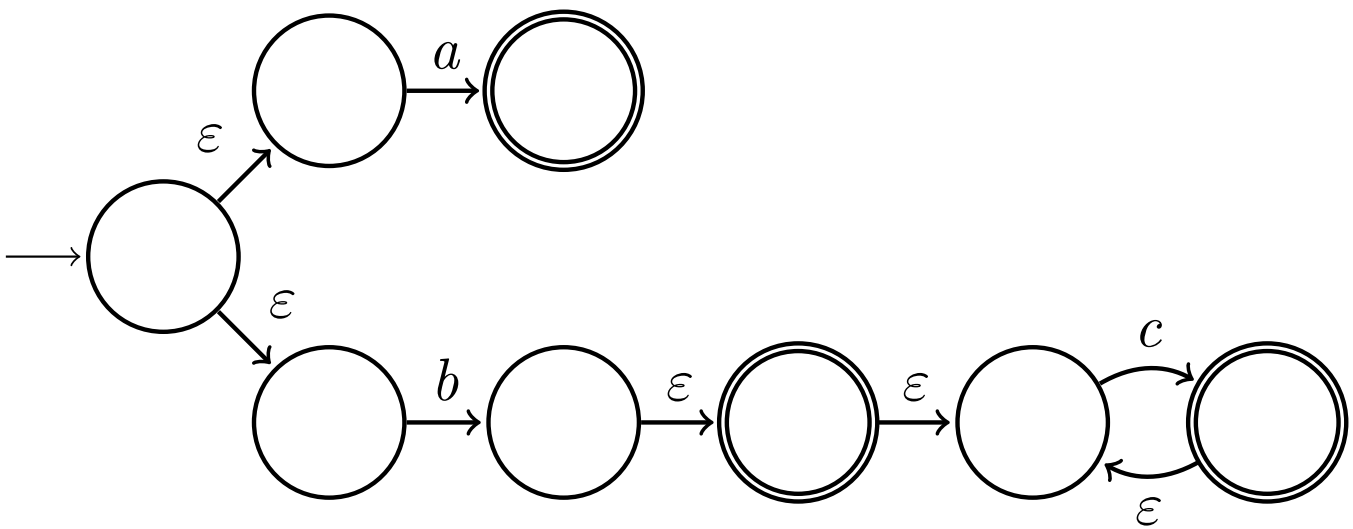
Example: $a \mid bc^\star$: from RE to NFA

1. Construct automata for the atomic regular expressions a, b, c
2. Use the Star Closure operation to find an automaton for $c^\star$
3. Use the Concatenation operation to find an automaton for $bc^\star$



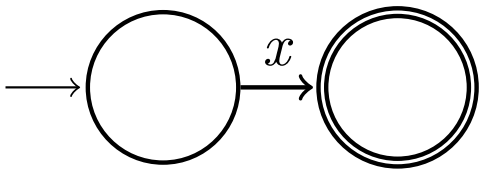
Example: $a \mid bc^\star$: from RE to NFA

1. Construct automata for the atomic regular expressions a, b, c
2. Use the Star Closure operation to find an automaton for $c^\star$
3. Use the Concatenation operation to find an automaton for $bc^\star$
4. Use the Union operation to find an automaton for $a|bc^\star$

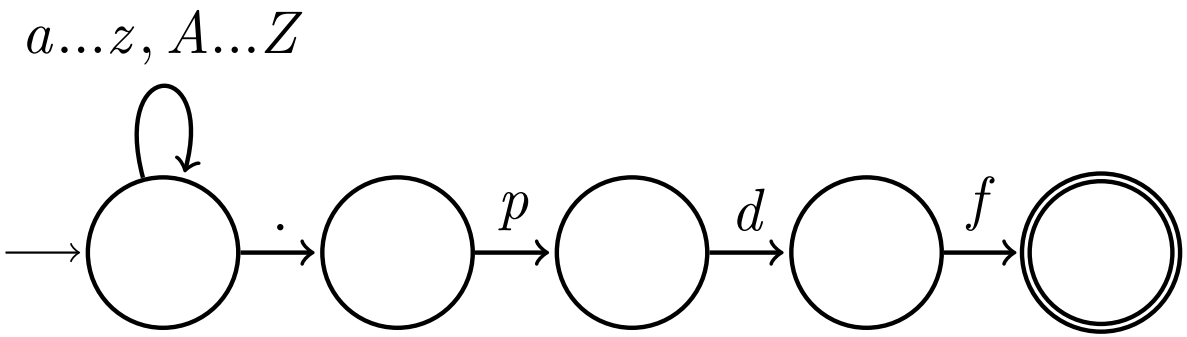


From NFA to RE: Simple examples

This automaton gives the RE x

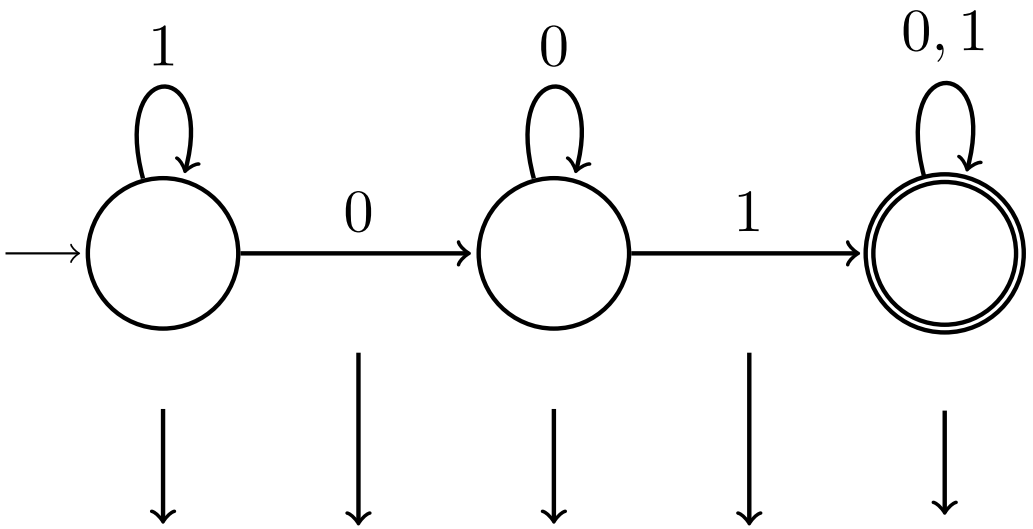


This one gives $(a \mid b \mid ... \mid Y \mid Z)^\star.pdf$



From NFA to RE: Simple examples

A more complex one:



$1^\star 00^\star 1(0 \mid 1)^\star$

Concept

Algorithm to convert an NFA to a RE:

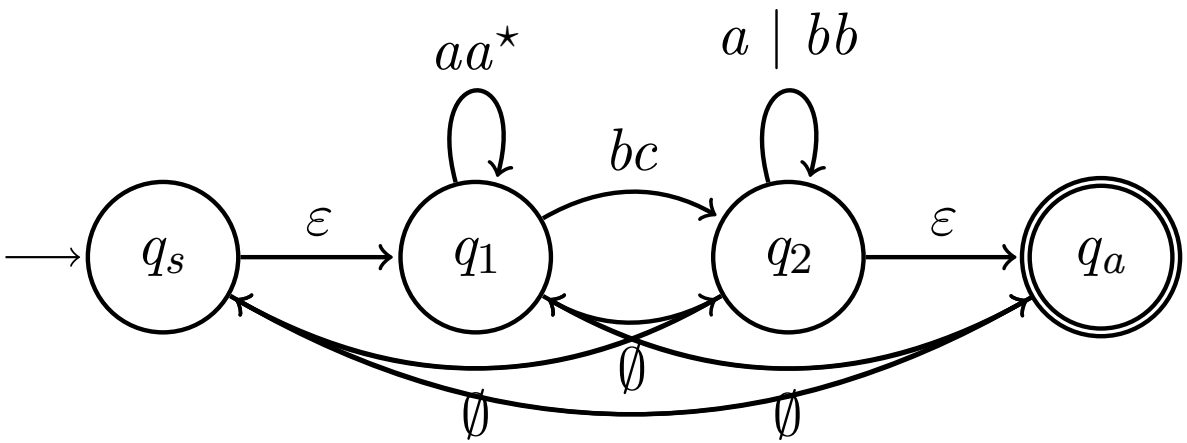
1. Convert the NFA to a Generalised NFA (GNFA)
   - Only one start state, no incoming transitions
   - Only one accept state, no outgoing transitions
   - Transitions described by RE
2. Progressively *eliminate* all states between the start and accept state
3. The transition between the start and the accept state is now a regular expression describing L

Generalised NFA (GNFA)

The start state q_s is non-accepting and has no incoming transitions

The *only* accept state q_a has no outgoing transitions, and $q_s \neq q_a$

There is exactly one transition between each ordered pair of states, labelled with a RE.



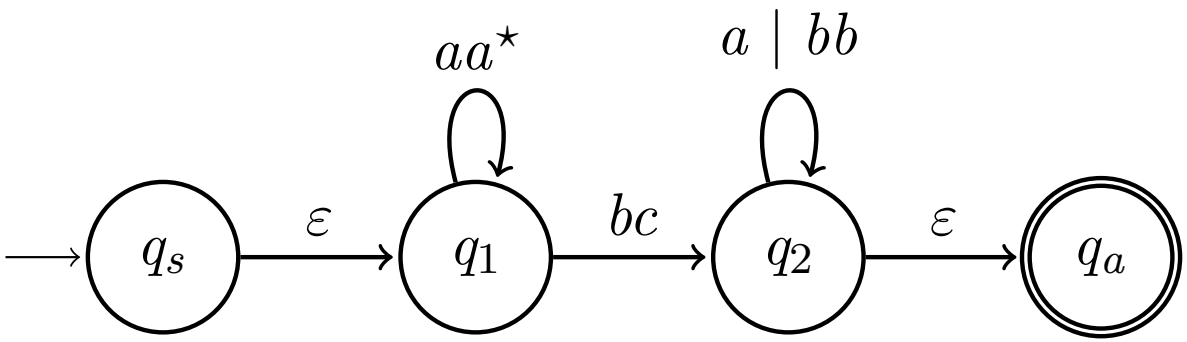
$\{abc, abca, abcbb, aaaaaabc\} \subseteq L$

Generalised NFA (GNFA)

The start state q_s is non-accepting and has no incoming transitions

The *only* accept state q_a has no outgoing transitions, and $q_s \neq q_a$

There is exactly one transition between each ordered pair of states, labelled with a RE.



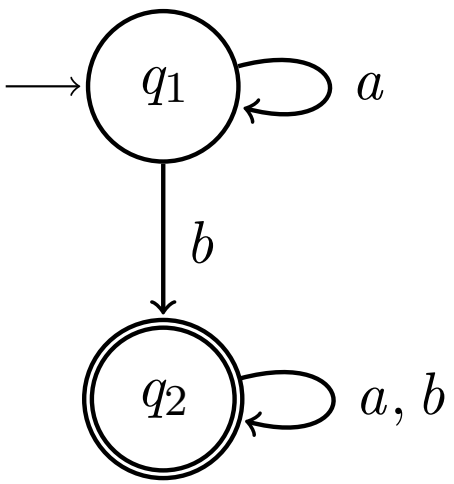
We do not show the $\emptyset$ transitions (why not?)
$\{abc, abca, abcbb, aaaaaabc\} \subseteq L$

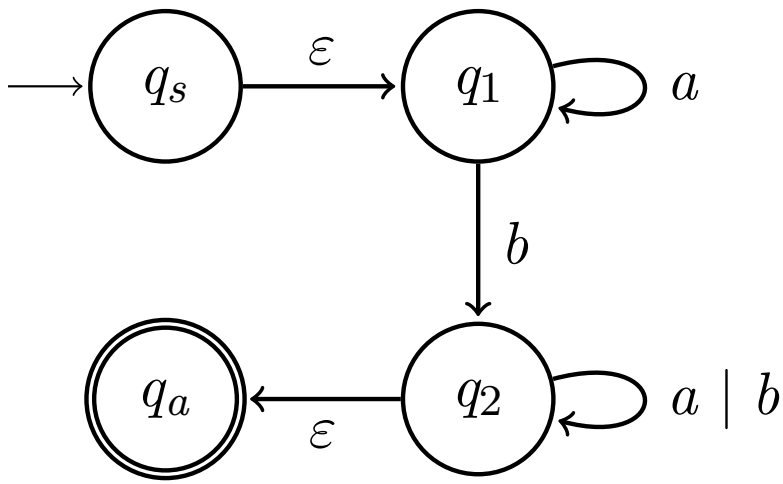
Converting an NFA to a GNFA

Create a new non-accepting start state q_s, with a ε-transition to the original start state.

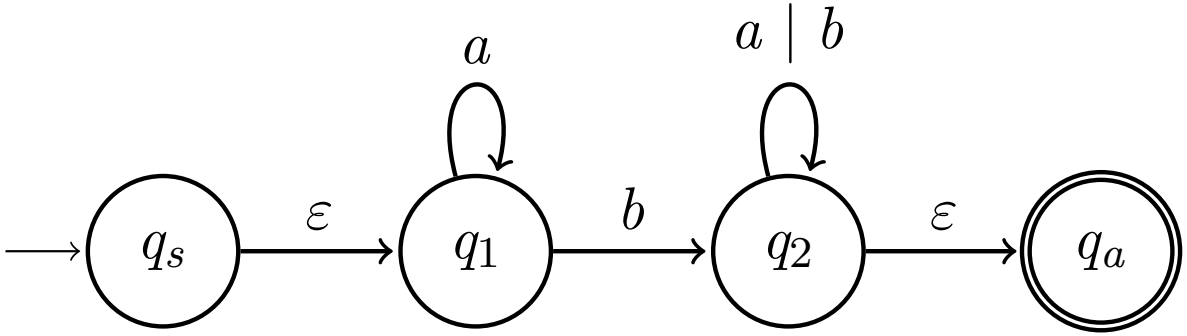
Create a new accept state q_a with ε-transitions from the original accept states, which are no longer accepting.

Label the transitions between every ordered pair (q_i, q_j) of states from the NFA as the union of the atomic RE describing each transition from q_i to q_j in the NFA.





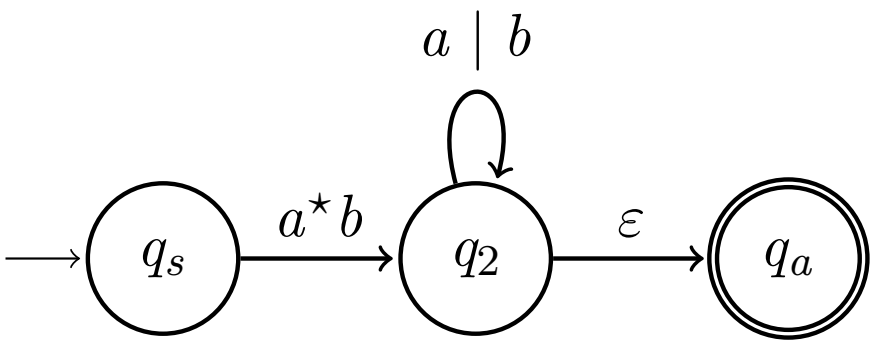
Simple example: eliminate state q_1



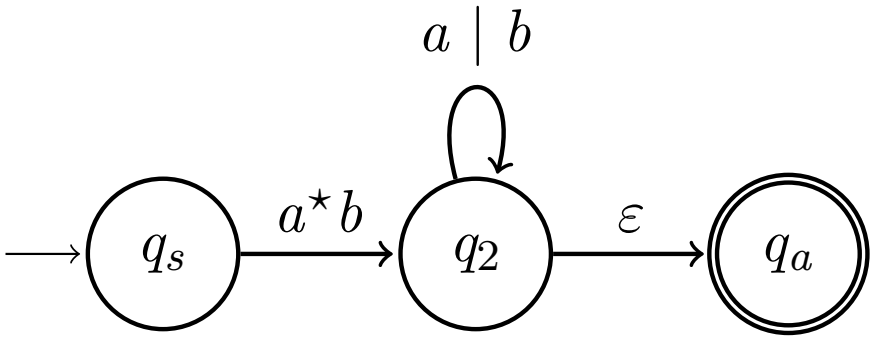
There is only one way to pass through q_1:

$q_s \to q_1$	ε
$q_1 \to q_1$ any number of times	$a^\star$
$q_1 \to q_2$	b

We set the transition $q_s \to q_2$ to the union of the new RE and it's old value ($\emptyset$). This is $\varepsilon a^\star b \mid \emptyset$, which simplifies to $a^\star b$

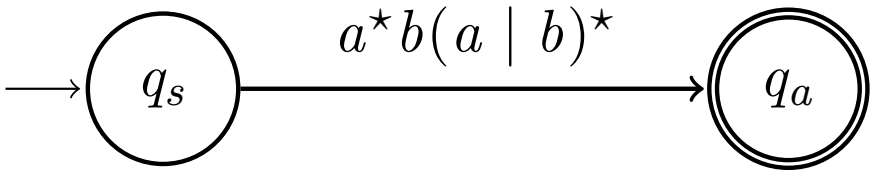


Simple example: eliminate state q_2



There is only one way to pass through q_2:

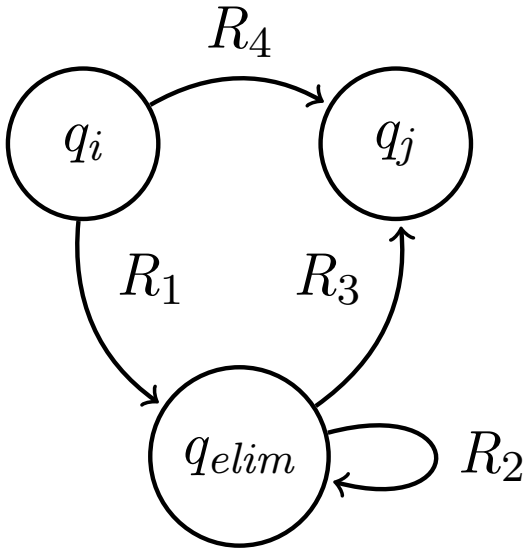
$q_s \to q_2$	$a^\star b$
$q_2 \to q_2$ any number of times	$(a\|b)^\star$
$q_2 \to q_a$	ε



The language of the original automaton is $a^\star b(a \mid b)^\star$

General method to eliminate state q_{elim}

Consider each pair (q_i, q_j) where $q_i, q_j \in Q \setminus \{q_{elim}\}$



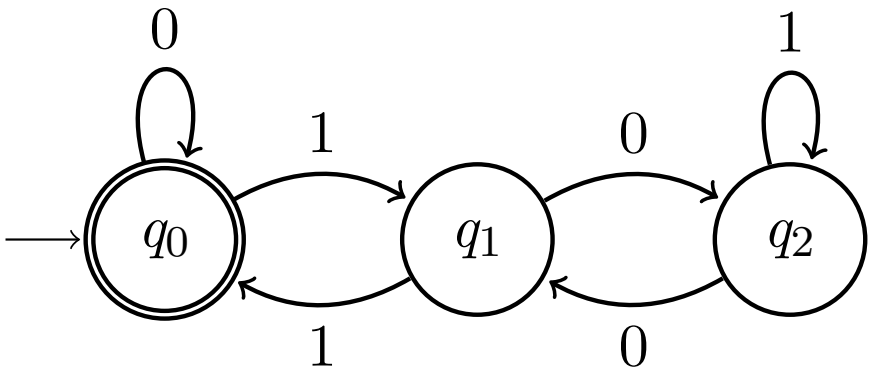
Replace the transition from q_i to q_j with the expression

$$((R_1)(R_2)^\star(R_3) \mid (R_4))$$

Note:

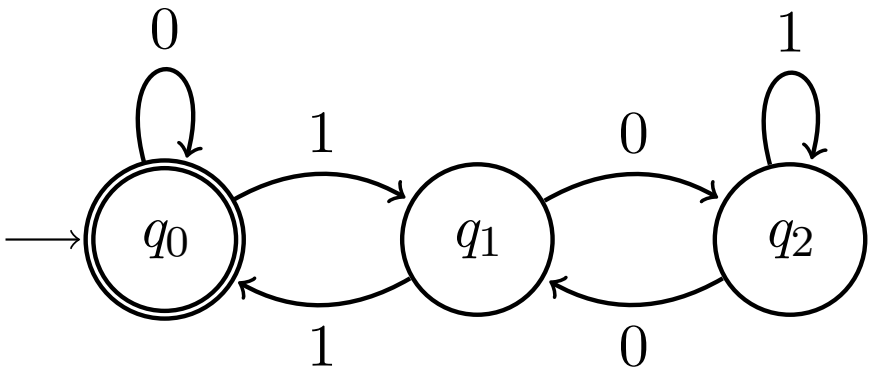
- Possibly $q_i = q_j$
- Recall that pairs are ordered, so we also consider (q_j, q_i)
- If there is no transition R_x, then $R_x = \emptyset$

Example 2: divisible by 3

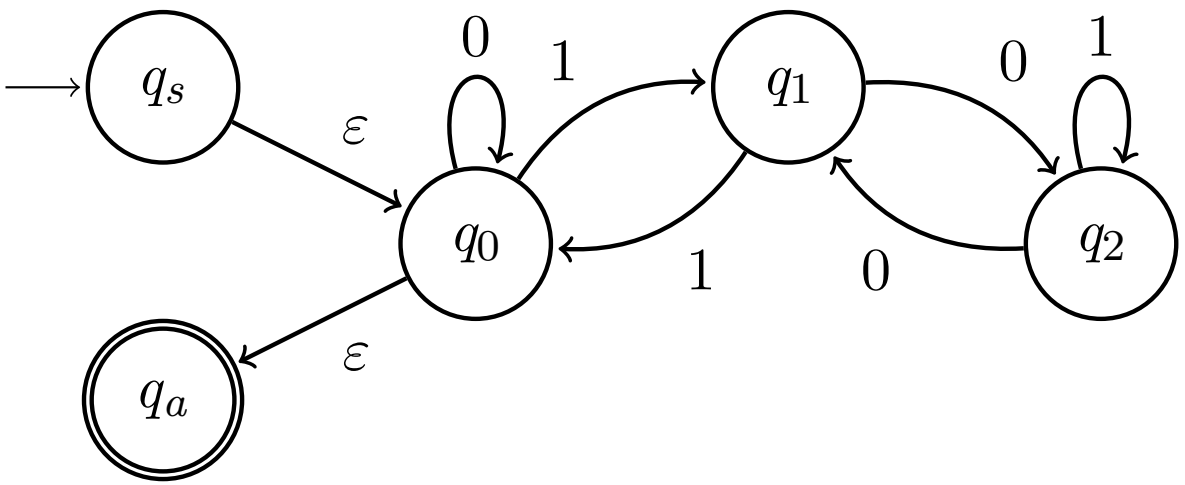


Convert to GNFA:

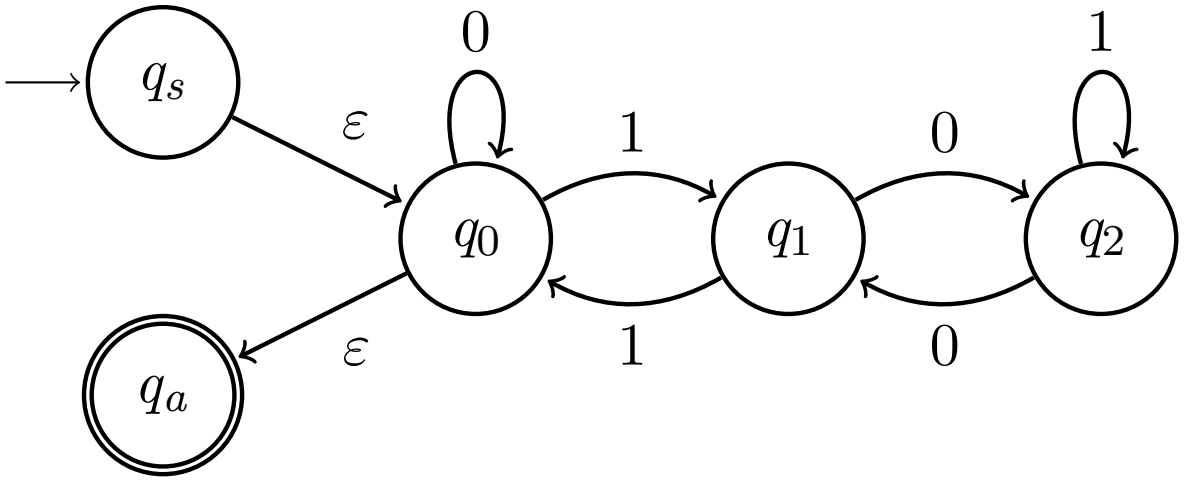
Example 2: divisible by 3



Convert to GNFA:



Example 2: divisible by 3

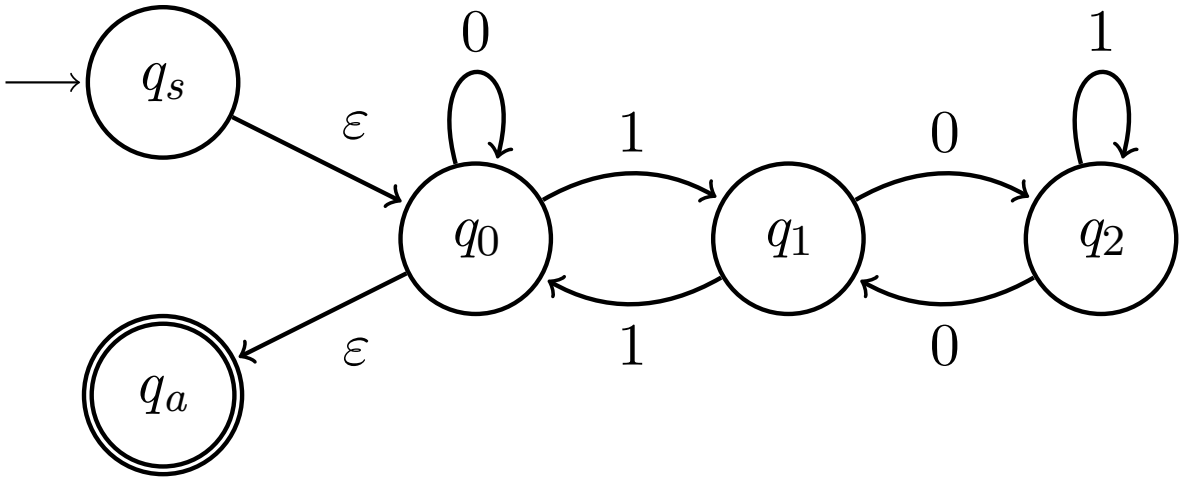


We can eliminate the states in any order. Eliminate q_0.

All pairs of states with transitions which are not $\emptyset$ through q_0:

-
-
-
-

Example 2: divisible by 3

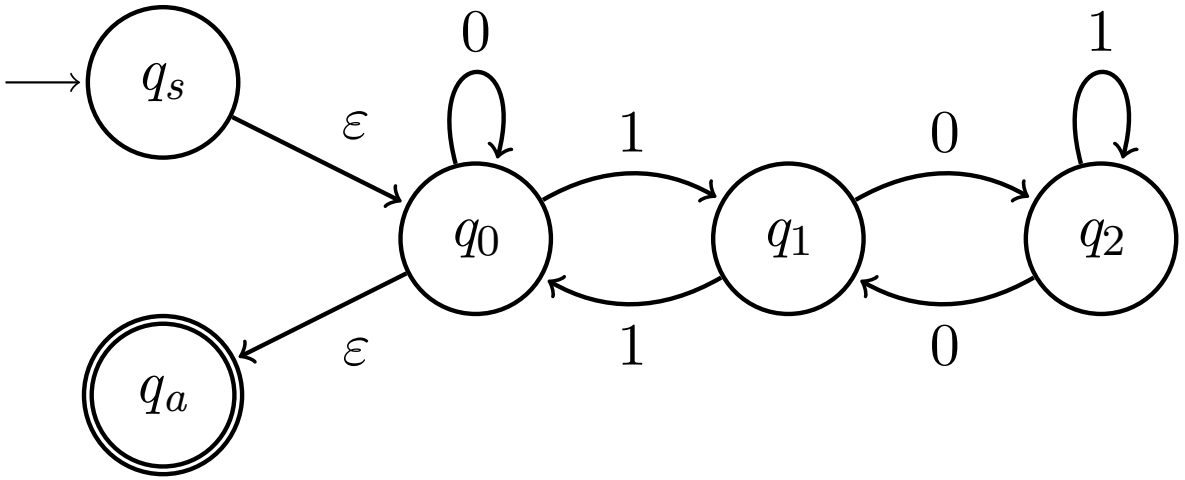


We can eliminate the states in any order. Eliminate q_0.

All pairs of states with transitions which are not $\emptyset$ through q_0:

- (q_s, q_1) :
- (q_s, q_a) :
- (q_1, q_1) :
- (q_1, q_a) :

Example 2: divisible by 3

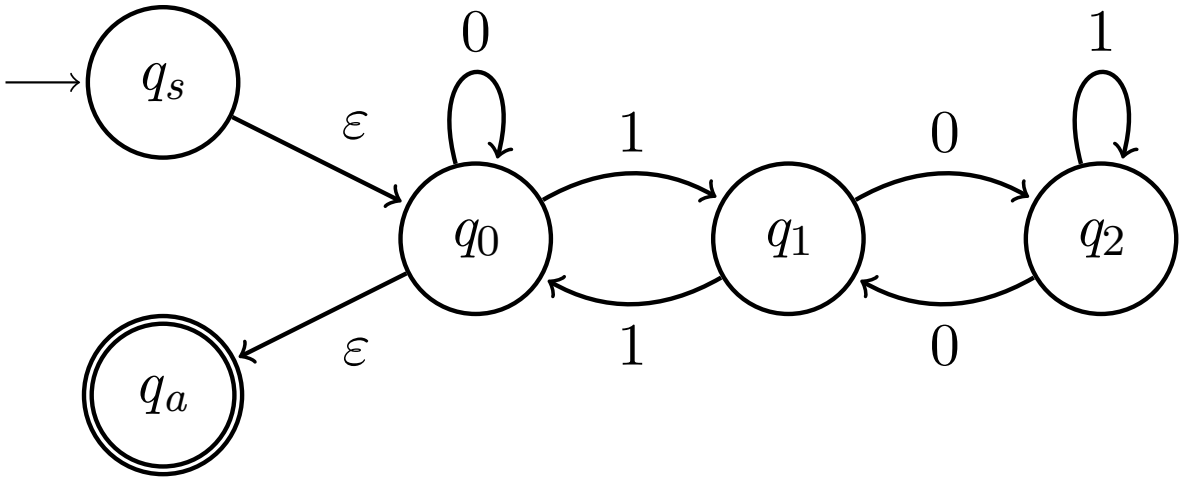


We can eliminate the states in any order. Eliminate q_0.

All pairs of states with transitions which are not $\emptyset$ through q_0:

- $(q_s, q_1) : R_1 = \varepsilon$,
- $(q_s, q_a) :$
- $(q_1, q_1) :$
- $(q_1, q_a) :$

Example 2: divisible by 3

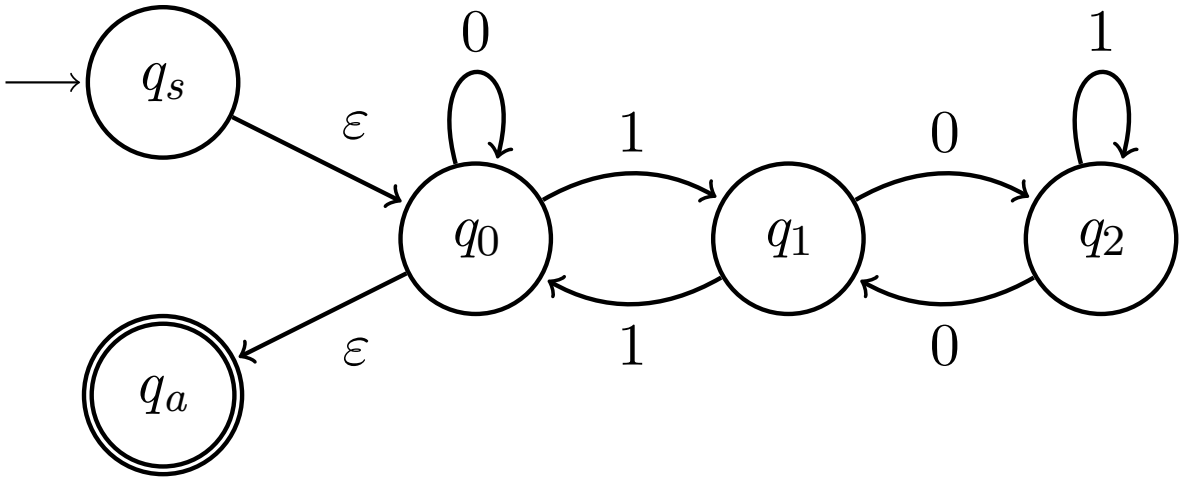


We can eliminate the states in any order. Eliminate q_0.

All pairs of states with transitions which are not $\emptyset$ through q_0:

- $(q_s, q_1) : R_1 = \varepsilon, R_2 = 0,$
- $(q_s, q_a) :$
- $(q_1, q_1) :$
- $(q_1, q_a) :$

Example 2: divisible by 3

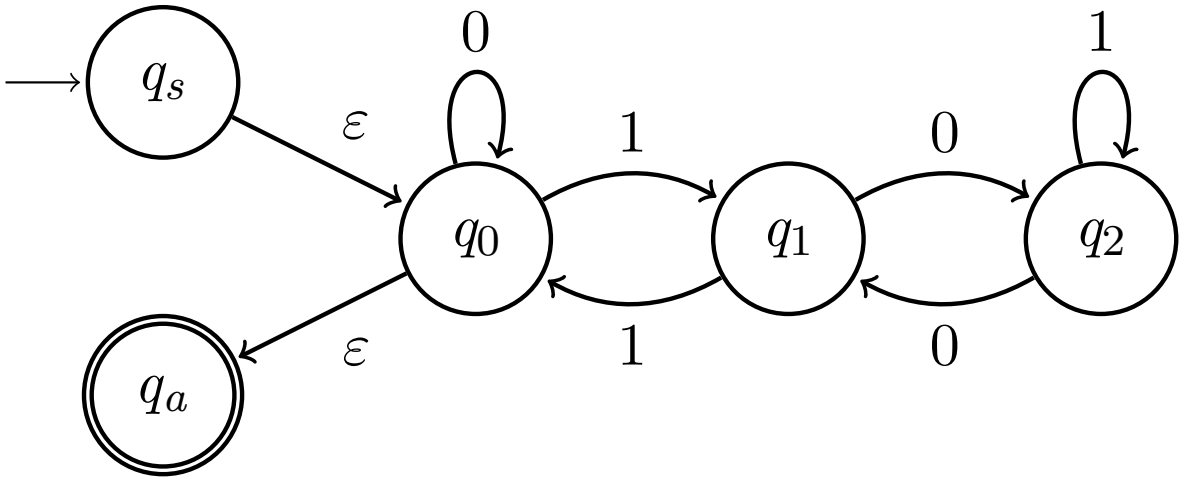


We can eliminate the states in any order. Eliminate q_0.

All pairs of states with transitions which are not $\emptyset$ through q_0:

- $(q_s, q_1) : R_1 = \varepsilon, R_2 = 0, R_3 = 1,$
- $(q_s, q_a) :$
- $(q_1, q_1) :$
- $(q_1, q_a) :$

Example 2: divisible by 3

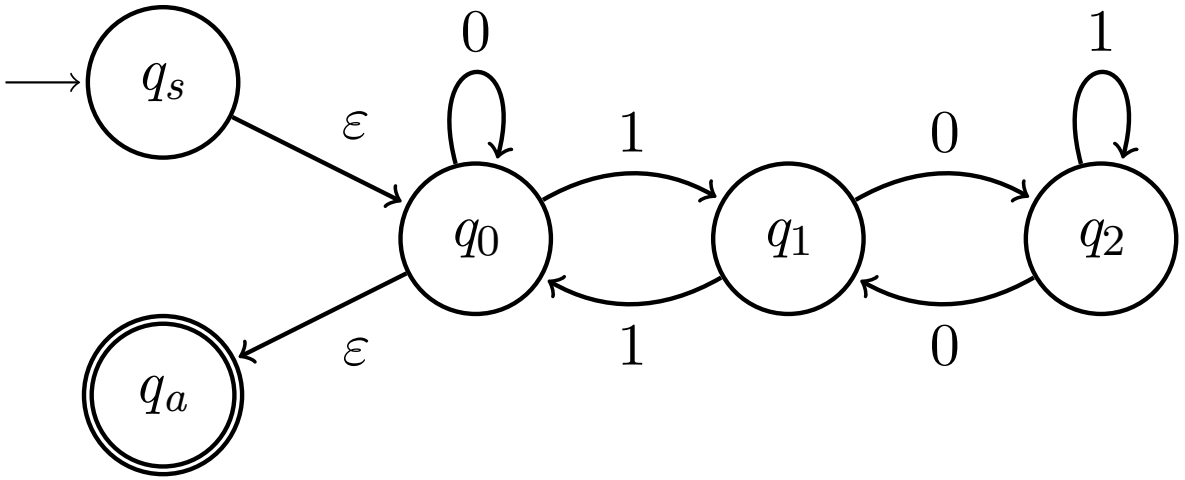


We can eliminate the states in any order. Eliminate q_0.

All pairs of states with transitions which are not $\emptyset$ through q_0:

- $(q_s, q_1) : R_1 = \varepsilon, R_2 = 0, R_3 = 1, R_4 = \emptyset$
- $(q_s, q_a) :$
- $(q_1, q_1) :$
- $(q_1, q_a) :$

Example 2: divisible by 3

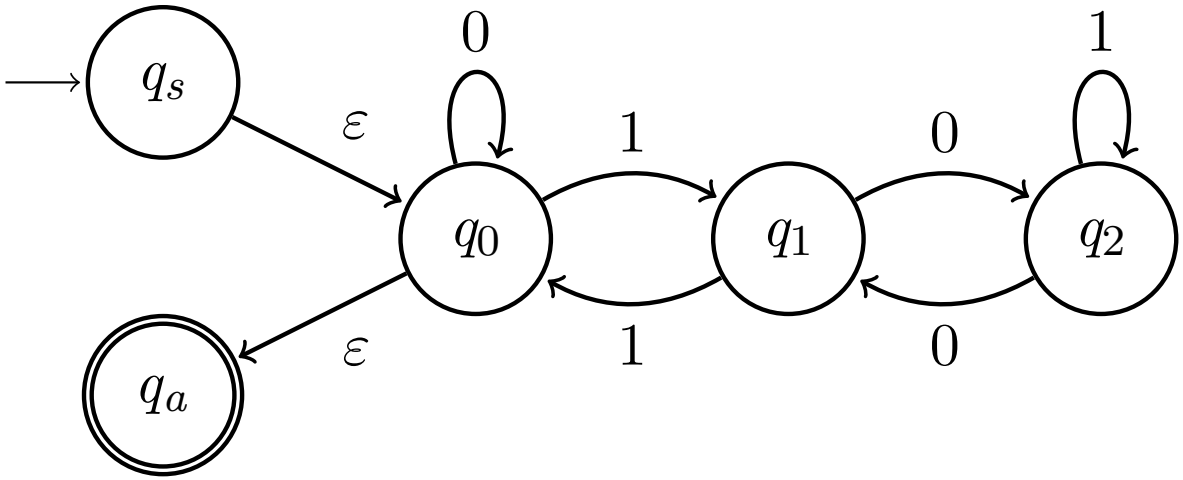


We can eliminate the states in any order. Eliminate q_0.

All pairs of states with transitions which are not $\emptyset$ through q_0:

- $(q_s, q_1) : R_1 = \varepsilon, R_2 = 0, R_3 = 1, R_4 = \emptyset \Rightarrow \varepsilon 0^\star 1 \mid \emptyset$
- (q_s, q_a) :
- (q_1, q_1) :
- (q_1, q_a) :

Example 2: divisible by 3

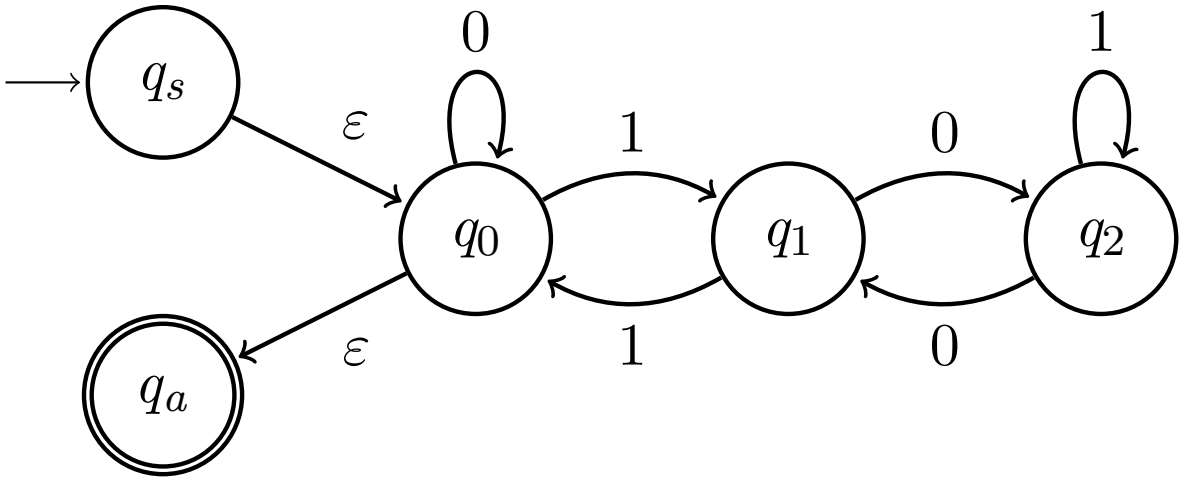


We can eliminate the states in any order. Eliminate q_0.

All pairs of states with transitions which are not $\emptyset$ through q_0:

- $(q_s, q_1) : R_1 = \varepsilon, R_2 = 0, R_3 = 1, R_4 = \emptyset \Rightarrow \varepsilon 0^\star 1 \mid \emptyset = 0^\star 1$
- (q_s, q_a) :
- (q_1, q_1) :
- (q_1, q_a) :

Example 2: divisible by 3

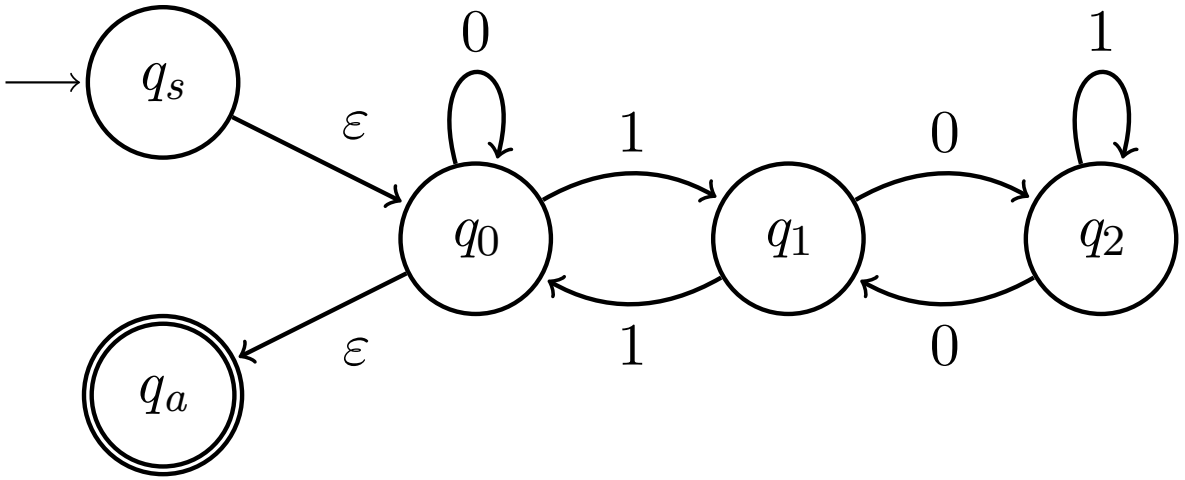


We can eliminate the states in any order. Eliminate q_0.

All pairs of states with transitions which are not $\emptyset$ through q_0:

- $(q_s, q_1) : R_1 = \varepsilon, R_2 = 0, R_3 = 1, R_4 = \emptyset \Rightarrow \varepsilon 0^\star 1 \mid \emptyset = 0^\star 1$
- $(q_s, q_a) : R_1 = \varepsilon,$
- $(q_1, q_1) :$
- $(q_1, q_a) :$

Example 2: divisible by 3

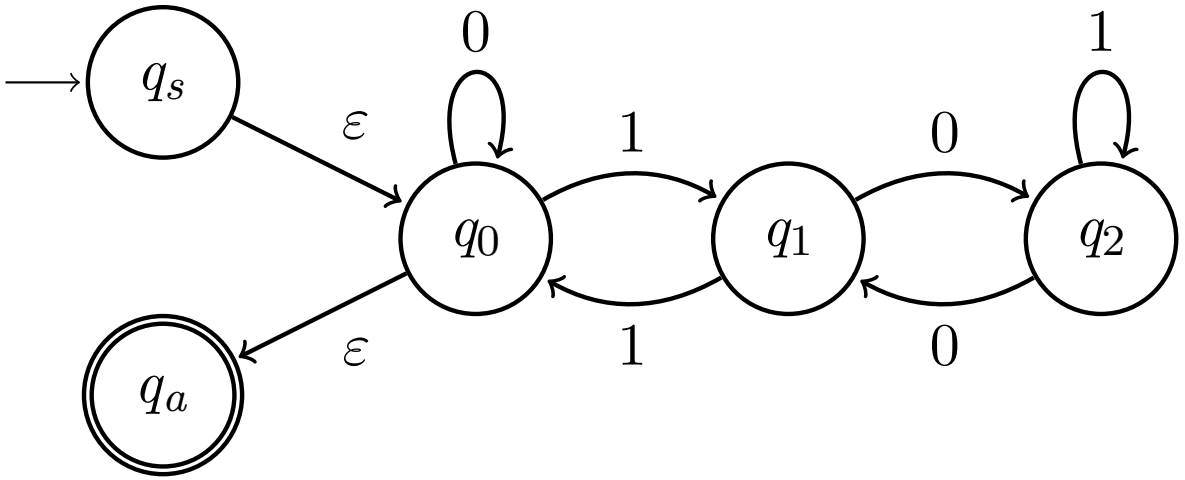


We can eliminate the states in any order. Eliminate q_0.

All pairs of states with transitions which are not $\emptyset$ through q_0:

- $(q_s, q_1) : R_1 = \varepsilon, R_2 = 0, R_3 = 1, R_4 = \emptyset \Rightarrow \varepsilon 0^\star 1 \mid \emptyset = 0^\star 1$
- $(q_s, q_a) : R_1 = \varepsilon, R_2 = 0,$
- $(q_1, q_1) :$
- $(q_1, q_a) :$

Example 2: divisible by 3

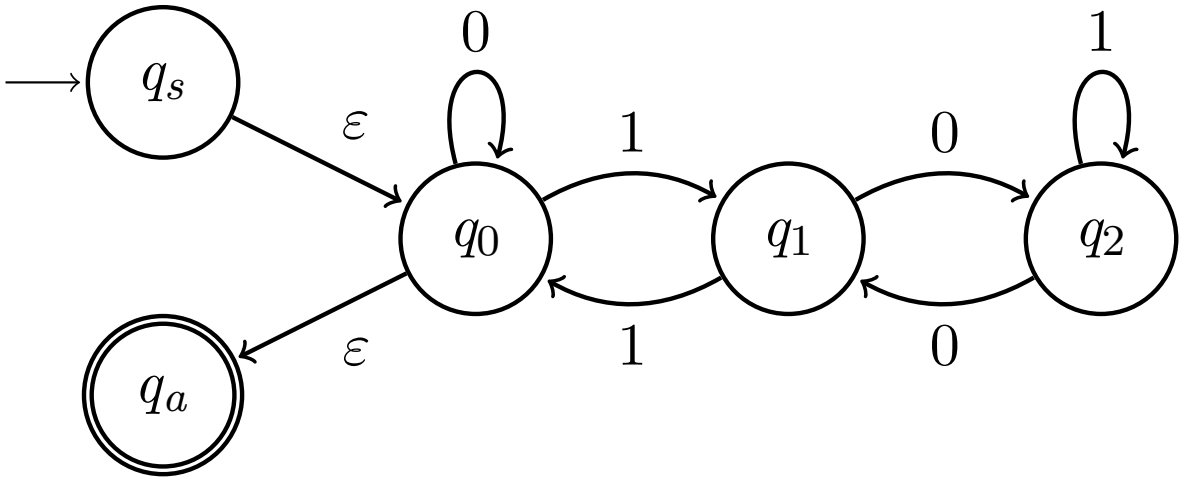


We can eliminate the states in any order. Eliminate q_0.

All pairs of states with transitions which are not $\emptyset$ through q_0:

- $(q_s, q_1) : R_1 = \varepsilon, R_2 = 0, R_3 = 1, R_4 = \emptyset \Rightarrow \varepsilon 0^\star 1 \mid \emptyset = 0^\star 1$
- $(q_s, q_a) : R_1 = \varepsilon, R_2 = 0, R_3 = \varepsilon,$
- $(q_1, q_1) :$
- $(q_1, q_a) :$

Example 2: divisible by 3

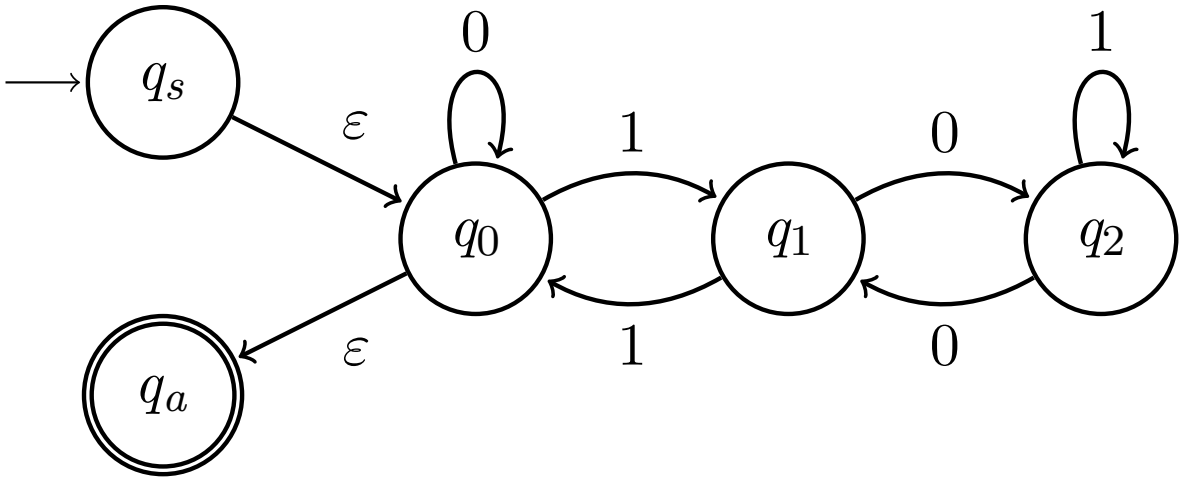


We can eliminate the states in any order. Eliminate q_0.

All pairs of states with transitions which are not $\emptyset$ through q_0:

- $(q_s, q_1) : R_1 = \varepsilon, R_2 = 0, R_3 = 1, R_4 = \emptyset \Rightarrow \varepsilon 0^\star 1 \mid \emptyset = 0^\star 1$
- $(q_s, q_a) : R_1 = \varepsilon, R_2 = 0, R_3 = \varepsilon, R_4 = \emptyset$
- $(q_1, q_1) :$
- $(q_1, q_a) :$

Example 2: divisible by 3

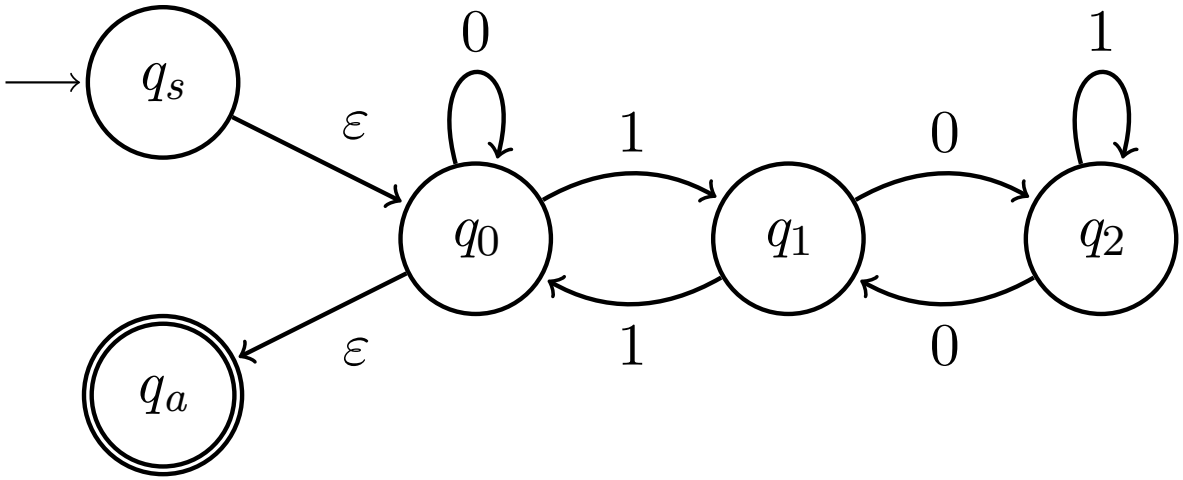


We can eliminate the states in any order. Eliminate q_0.

All pairs of states with transitions which are not $\emptyset$ through q_0:

- $(q_s, q_1) : R_1 = \varepsilon, R_2 = 0, R_3 = 1, R_4 = \emptyset \Rightarrow \varepsilon 0^\star 1 \mid \emptyset = 0^\star 1$
- $(q_s, q_a) : R_1 = \varepsilon, R_2 = 0, R_3 = \varepsilon, R_4 = \emptyset \Rightarrow \varepsilon 0^\star \varepsilon \mid \emptyset$
- $(q_1, q_1) :$
- $(q_1, q_a) :$

Example 2: divisible by 3

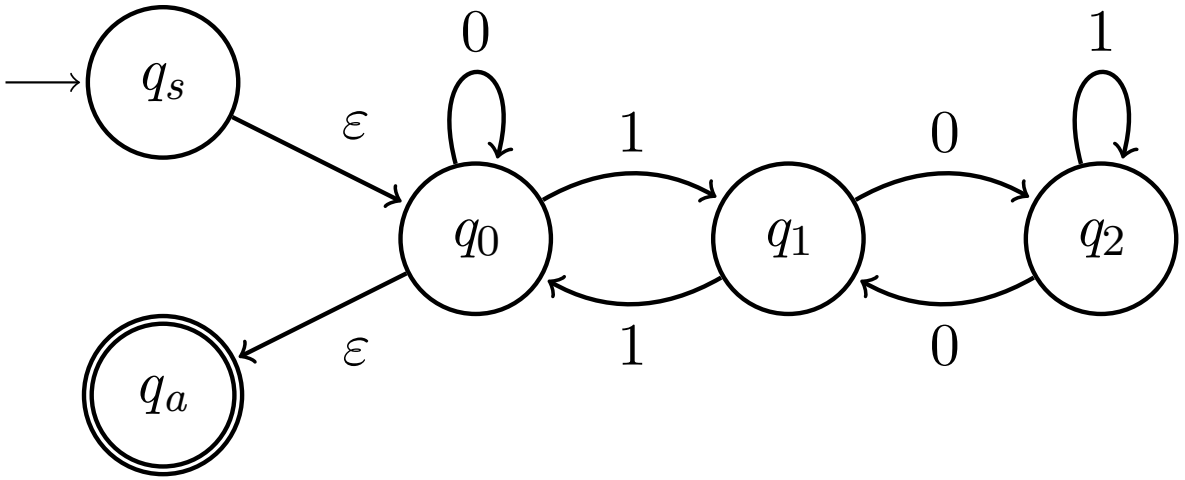


We can eliminate the states in any order. Eliminate q_0.

All pairs of states with transitions which are not $\emptyset$ through q_0:

- $(q_s, q_1) : R_1 = \varepsilon, R_2 = 0, R_3 = 1, R_4 = \emptyset \Rightarrow \varepsilon 0^\star 1 \mid \emptyset = 0^\star 1$
- $(q_s, q_a) : R_1 = \varepsilon, R_2 = 0, R_3 = \varepsilon, R_4 = \emptyset \Rightarrow \varepsilon 0^\star \varepsilon \mid \emptyset = 0^\star$
- $(q_1, q_1) :$
- $(q_1, q_a) :$

Example 2: divisible by 3

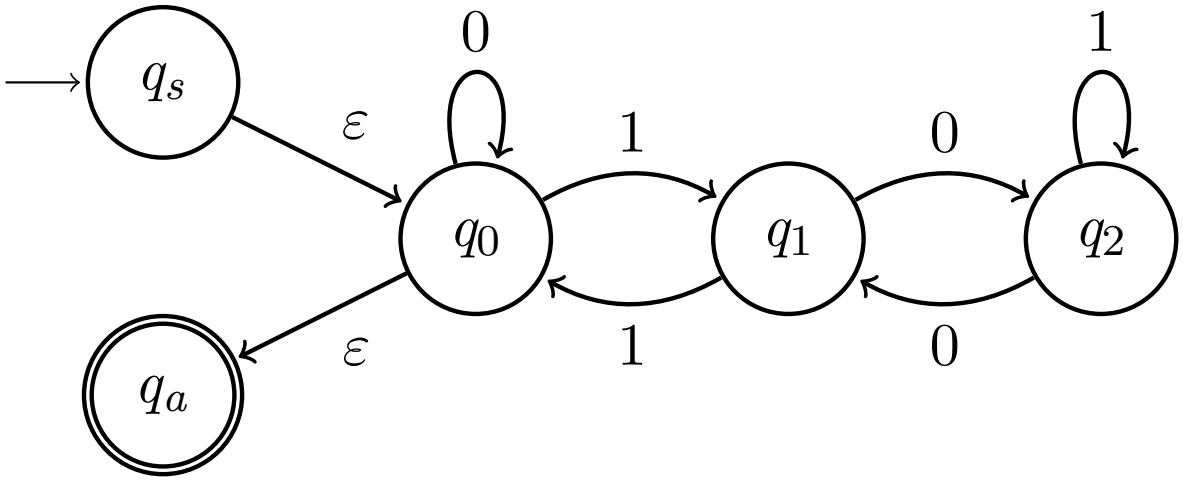


We can eliminate the states in any order. Eliminate q_0.

All pairs of states with transitions which are not $\emptyset$ through q_0:

- $(q_s, q_1) : R_1 = \varepsilon, R_2 = 0, R_3 = 1, R_4 = \emptyset \Rightarrow \varepsilon 0^\star 1 \mid \emptyset = 0^\star 1$
- $(q_s, q_a) : R_1 = \varepsilon, R_2 = 0, R_3 = \varepsilon, R_4 = \emptyset \Rightarrow \varepsilon 0^\star \varepsilon \mid \emptyset = 0^\star$
- $(q_1, q_1) : R_1 = 1, R_2 = 0, R_3 = 1, R_4 = \emptyset \Rightarrow 10^\star 1 \mid \emptyset = 10^\star 1$
- $(q_1, q_a) :$

Example 2: divisible by 3

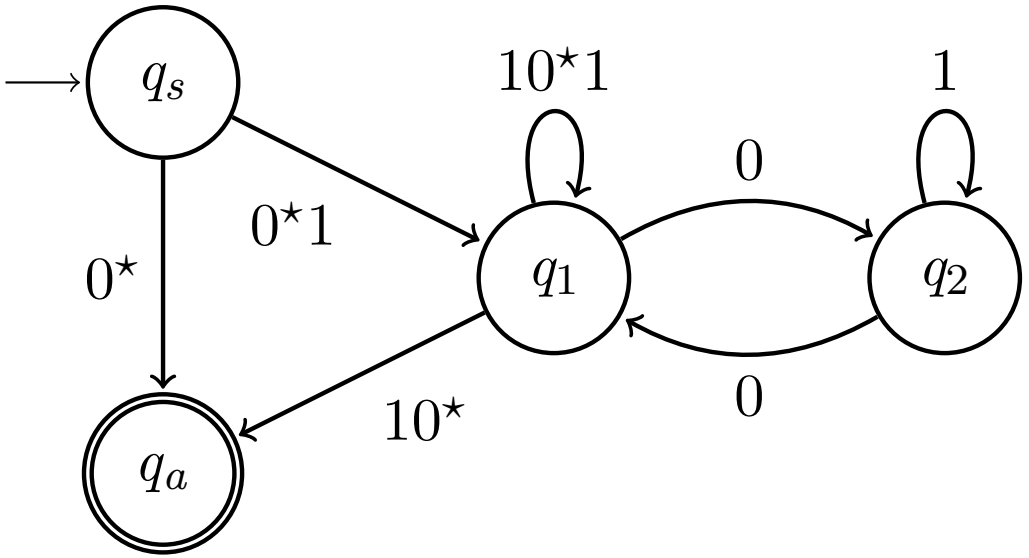


We can eliminate the states in any order. Eliminate q_0.

All pairs of states with transitions which are not $\emptyset$ through q_0:

- $(q_s, q_1) : R_1 = \varepsilon, R_2 = 0, R_3 = 1, R_4 = \emptyset \Rightarrow \varepsilon 0^\star 1 \mid \emptyset = 0^\star 1$
- $(q_s, q_a) : R_1 = \varepsilon, R_2 = 0, R_3 = \varepsilon, R_4 = \emptyset \Rightarrow \varepsilon 0^\star \varepsilon \mid \emptyset = 0^\star$
- $(q_1, q_1) : R_1 = 1, R_2 = 0, R_3 = 1, R_4 = \emptyset \Rightarrow 10^\star 1 \mid \emptyset = 10^\star 1$
- $(q_1, q_a) : R_1 = 1, R_2 = 0, R_3 = \varepsilon, R_4 = \emptyset \Rightarrow 10^\star \varepsilon \mid \emptyset = 10^\star$

Example 2: divisible by 3

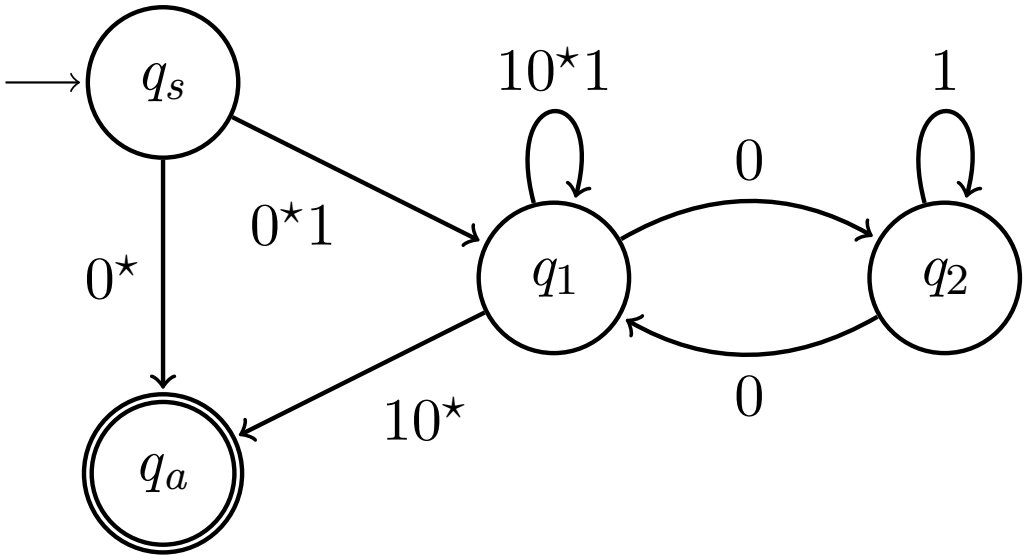


We can eliminate the states in any order. Eliminate q_2.

All pairs of states with transitions which are not $\emptyset$ through q_2:

- ▶

Example 2: divisible by 3

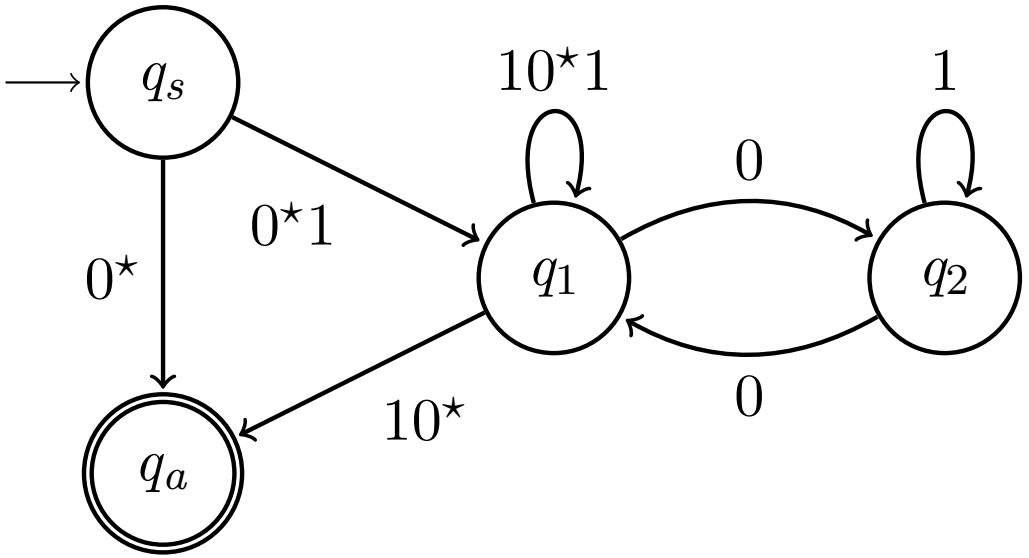


We can eliminate the states in any order. Eliminate q_2.

All pairs of states with transitions which are not $\emptyset$ through q_2:

- (q_1, q_1) :

Example 2: divisible by 3

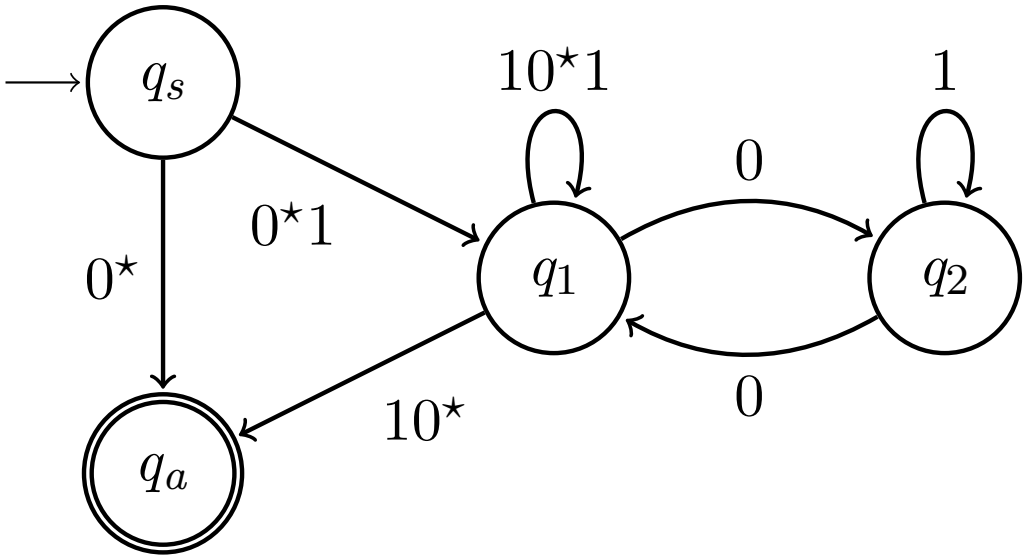


We can eliminate the states in any order. Eliminate q_2.

All pairs of states with transitions which are not $\emptyset$ through q_2:

- $(q_1, q_1) : R_1 = 0,$

Example 2: divisible by 3

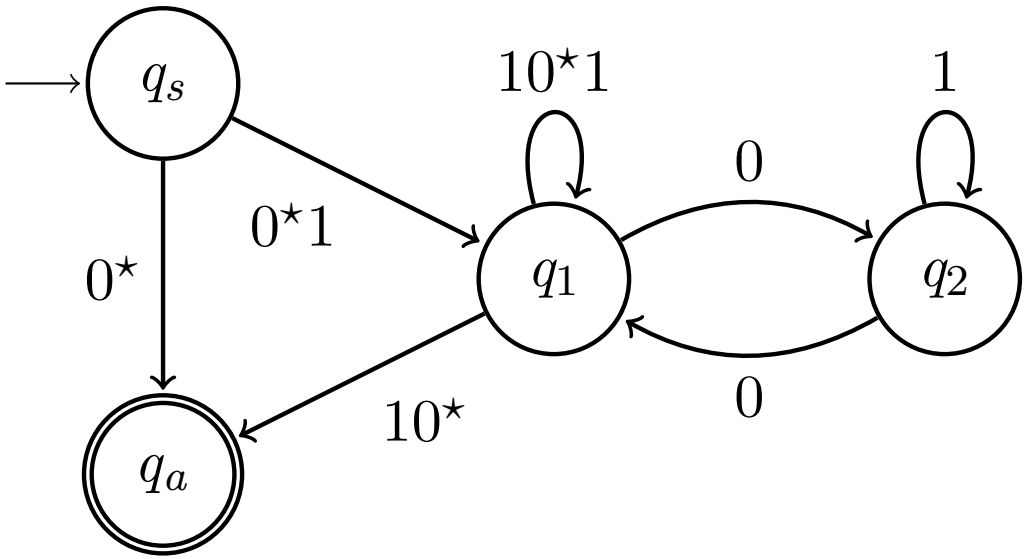


We can eliminate the states in any order. Eliminate q_2.

All pairs of states with transitions which are not $\emptyset$ through q_2:

- $(q_1, q_1) : R_1 = 0, R_2 = 1,$

Example 2: divisible by 3

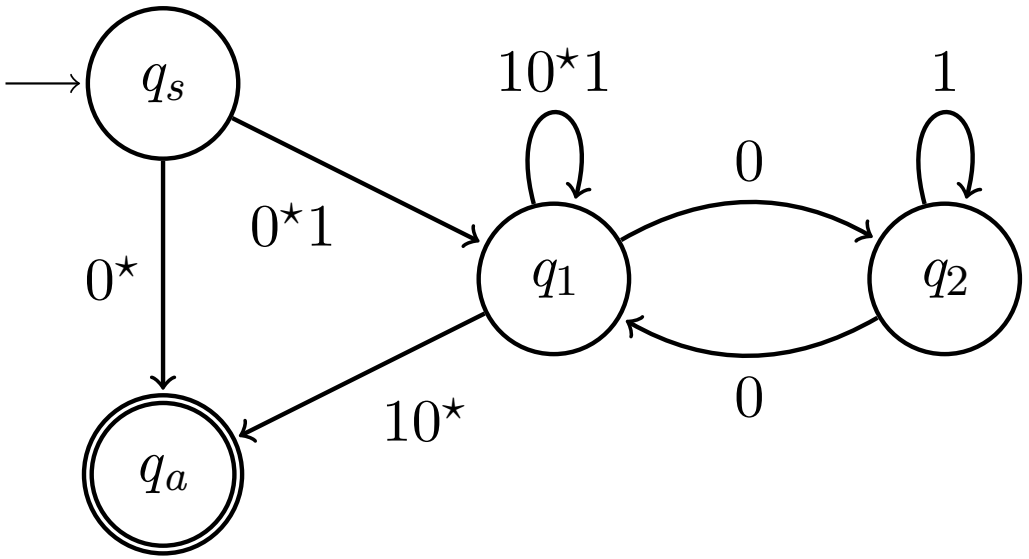


We can eliminate the states in any order. Eliminate q_2.

All pairs of states with transitions which are not $\emptyset$ through q_2:

- $(q_1, q_1) : R_1 = 0, R_2 = 1, R_3 = 0,$

Example 2: divisible by 3

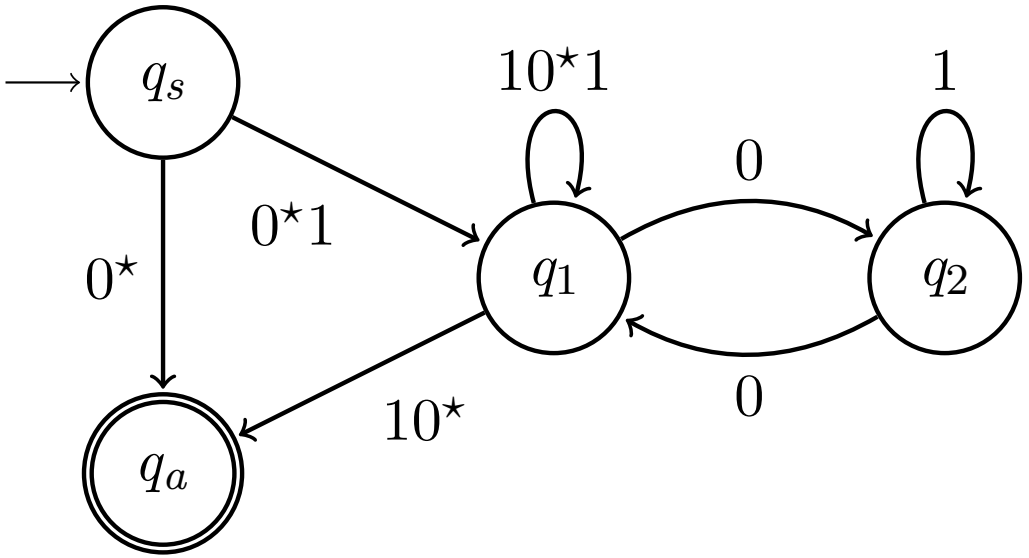


We can eliminate the states in any order. Eliminate q_2.

All pairs of states with transitions which are not $\emptyset$ through q_2:

- $(q_1, q_1) : R_1 = 0, R_2 = 1, R_3 = 0, R_4 = 10^\star1$

Example 2: divisible by 3



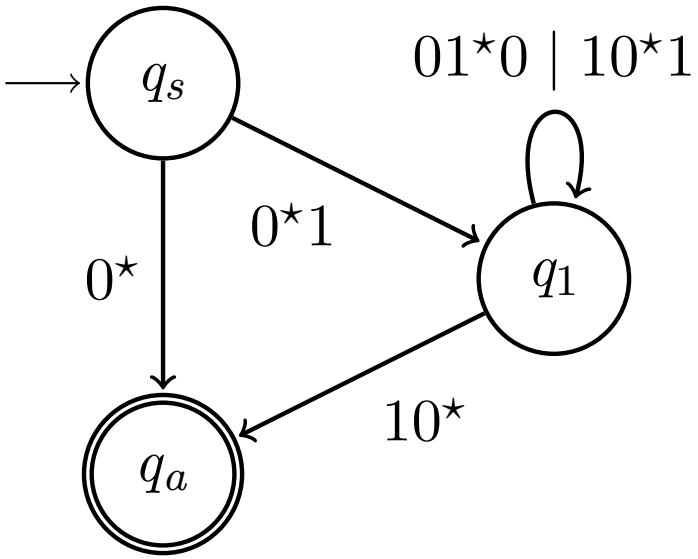
We can eliminate the states in any order. Eliminate q_2.

All pairs of states with transitions which are not $\emptyset$ through q_2:

- $(q_1, q_1) : R_1 = 0, R_2 = 1, R_3 = 0, R_4 = 10^\star1 \Rightarrow 01^\star0 \mid 10^\star1$

Much easier!

Example 2: divisible by 3

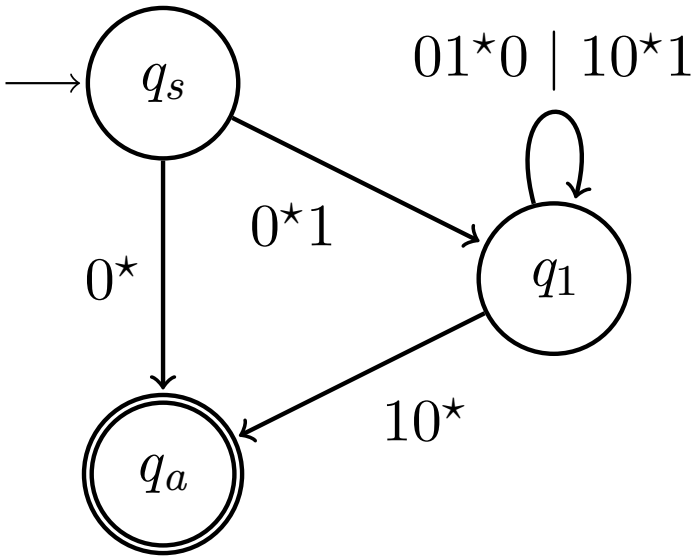


There is only q_1 left to remove.

All pairs of states with transitions which are not $\emptyset$ through q_1:

- ▶

Example 2: divisible by 3

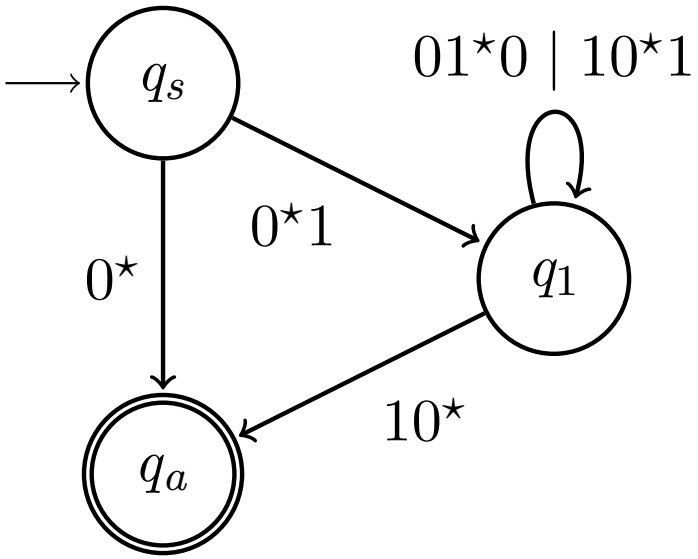


There is only q_1 left to remove.

All pairs of states with transitions which are not $\emptyset$ through q_1:

- (q_s, q_a) :

Example 2: divisible by 3

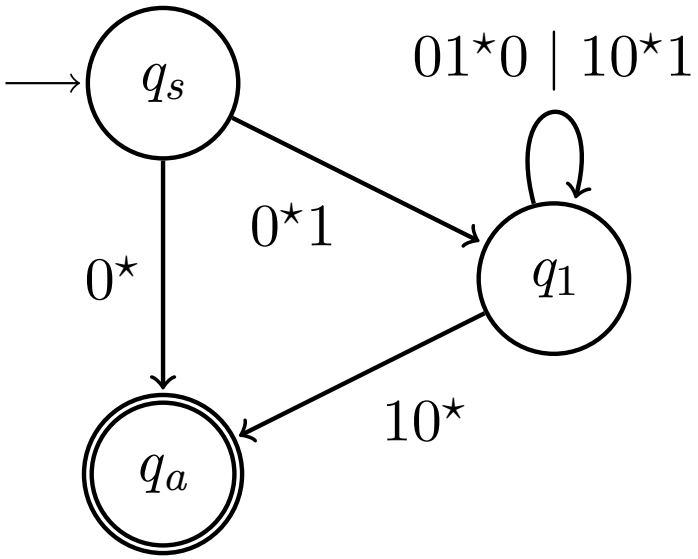


There is only q_1 left to remove.

All pairs of states with transitions which are not $\emptyset$ through q_1:

- $(q_s, q_a) : R_1 = 0^\star1,$

Example 2: divisible by 3

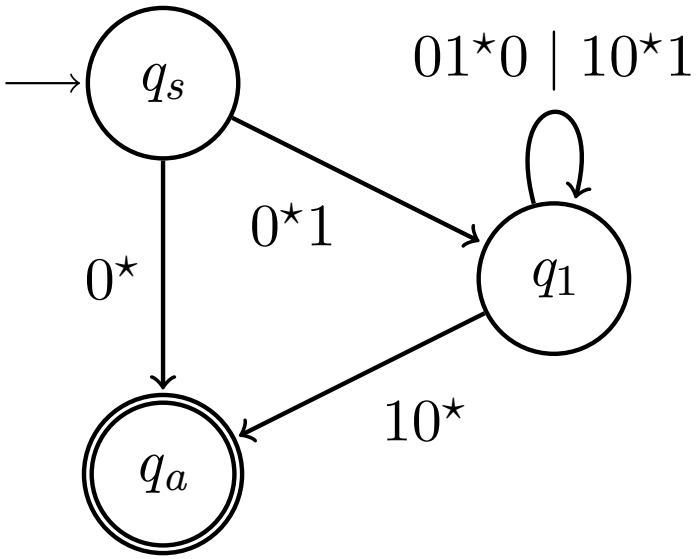


There is only q_1 left to remove.

All pairs of states with transitions which are not $\emptyset$ through q_1:

- $(q_s, q_a) : R_1 = 0^\star1, R_2 = 01^\star0 \mid 10^\star1,$

Example 2: divisible by 3

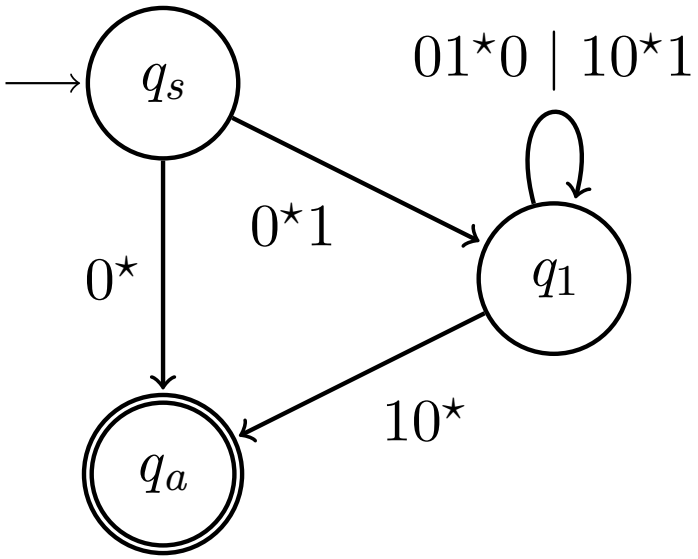


There is only q_1 left to remove.

All pairs of states with transitions which are not $\emptyset$ through q_1:

- $(q_s, q_a) : R_1 = 0^\star1, R_2 = 01^\star0 \mid 10^\star1, R_3 = 10^\star,$

Example 2: divisible by 3

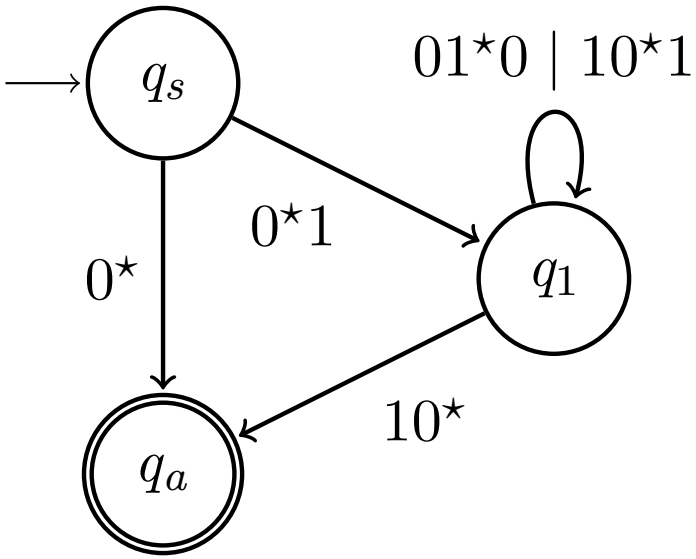


There is only q_1 left to remove.

All pairs of states with transitions which are not $\emptyset$ through q_1:

- $(q_s, q_a) : R_1 = 0^\star1, R_2 = 01^\star0 \mid 10^\star1, R_3 = 10^\star, R_4 = 0^\star$

Example 2: divisible by 3

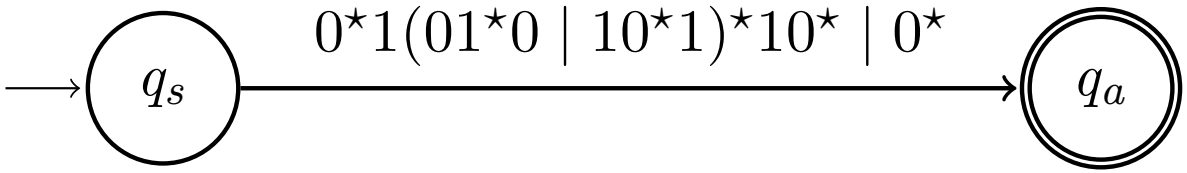


There is only q_1 left to remove.

All pairs of states with transitions which are not $\emptyset$ through q_1:

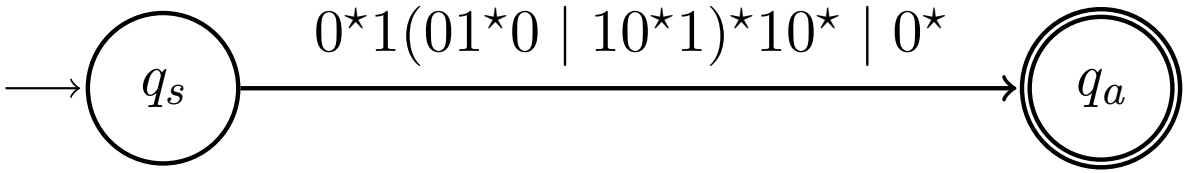
- $(q_s, q_a) : R_1 = 0^\star1, R_2 = 01^\star0 \mid 10^\star1, R_3 = 10^\star, R_4 = 0^\star$
  so $(R_1)(R_2)^\star(R_3) \mid (R_4) = 0^\star1(01^\star0 \mid 10^\star1)^\star10^\star \mid 0^\star$

Example 2: divisible by 3



So, $0^\star1(01^\star0 \mid 10^\star1)^\star10^\star \mid 0^\star$ is a RE which recognises binary strings which are divisible by 3.

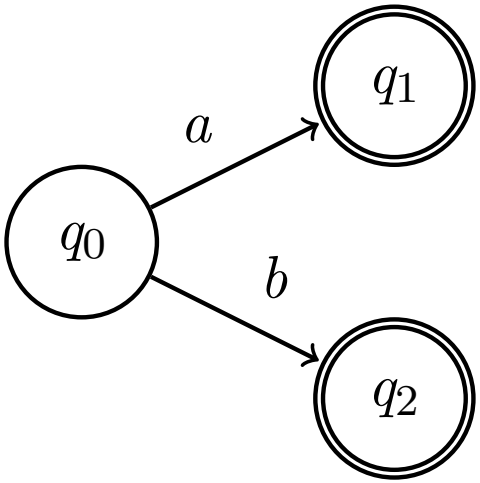
Example 2: divisible by 3



So, $0^\star 1(01^\star 0 \mid 10^\star 1)^\star 10^\star \mid 0^\star$ is a RE which recognises binary strings which are divisible by 3.

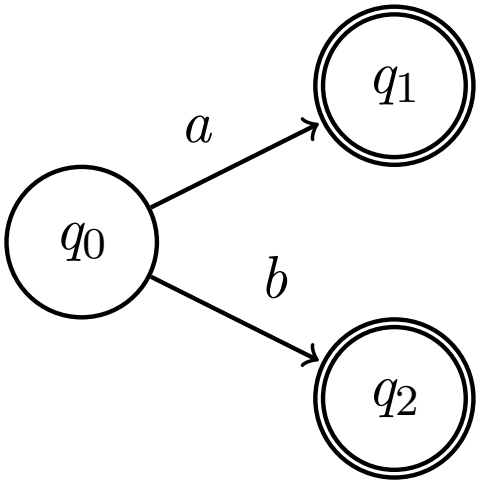
Eliminating states in a different order can result in a different, but equivalent RE.
e.g. eliminating q_2, q_1, then q_0 yields: $(1(01^\star 0)^\star 1|0)^\star$

Example 3: Multiple accept states

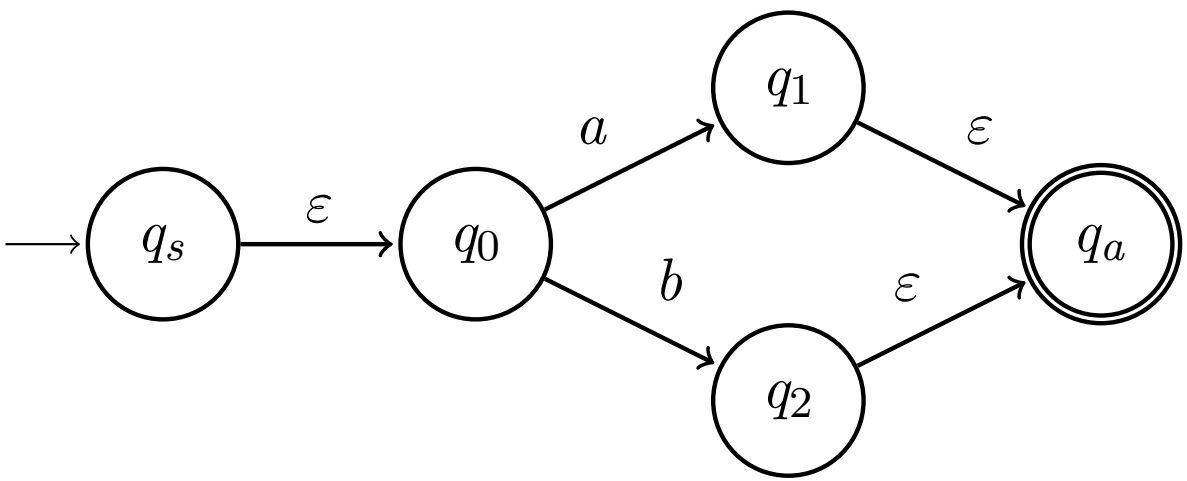


As GNFA:

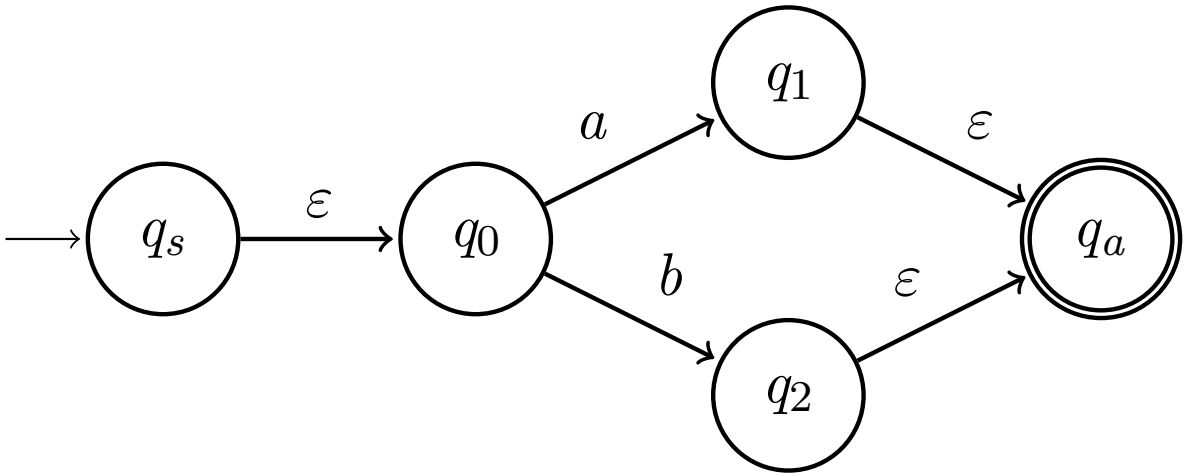
Example 3: Multiple accept states



As GNFA:

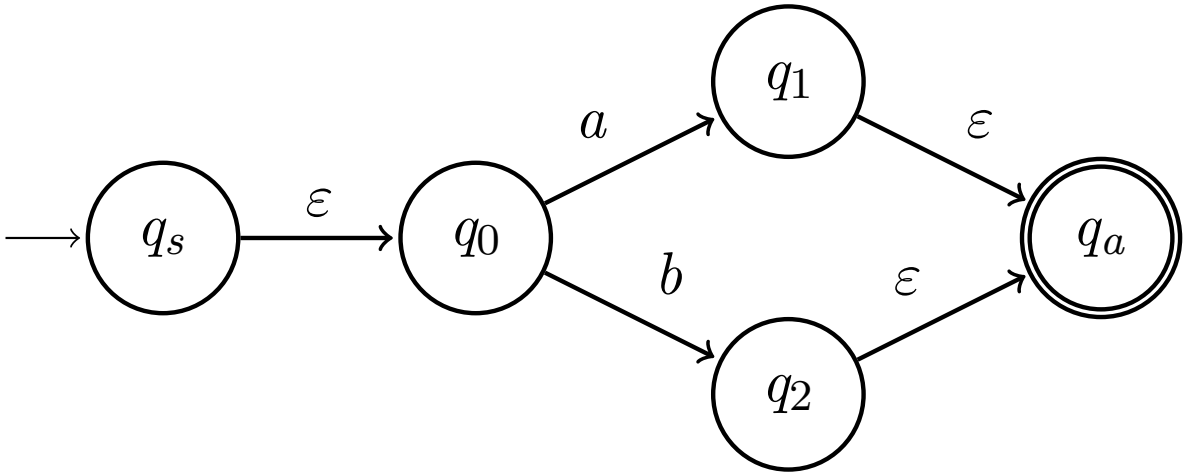


Example 3: Multiple accept states

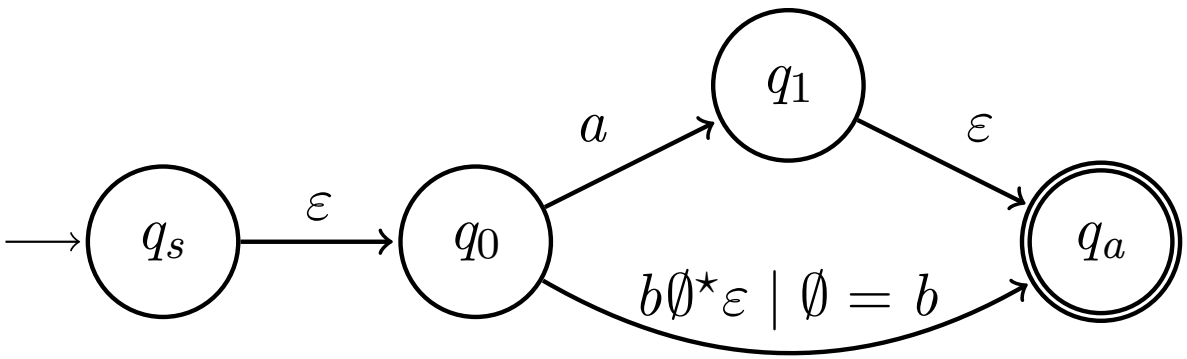


Eliminate q_2:

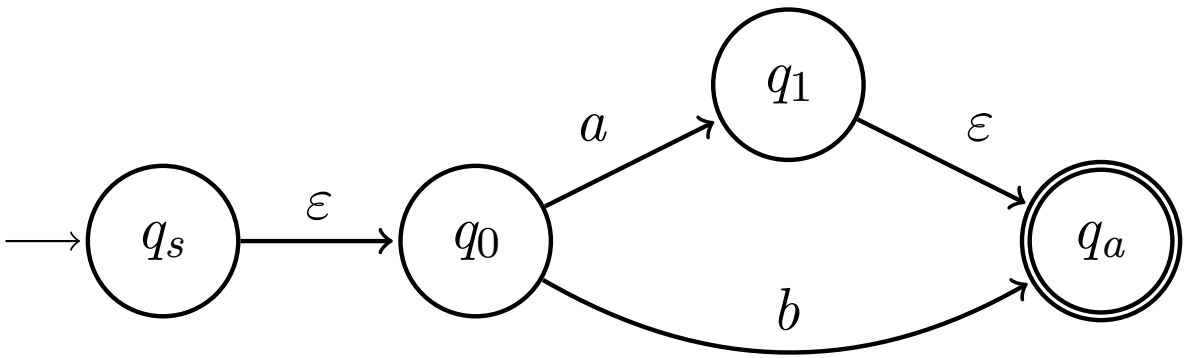
Example 3: Multiple accept states



Eliminate q_2:

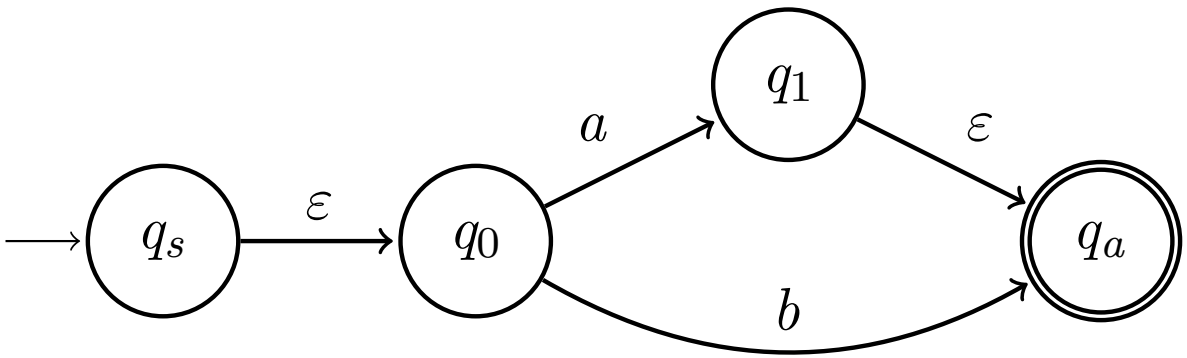


Example 3: Multiple accept states

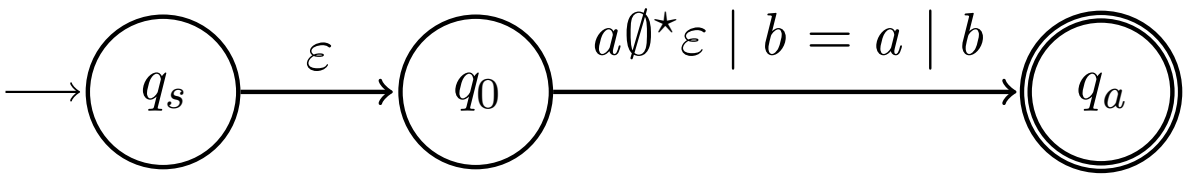


Eliminate q_1:

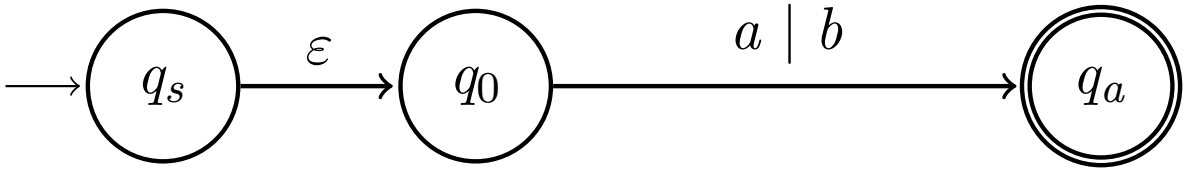
Example 3: Multiple accept states



Eliminate q_1:



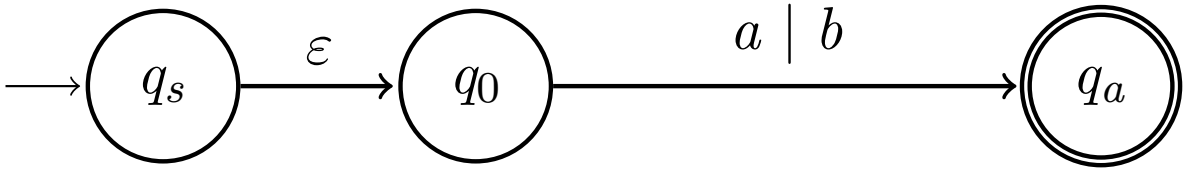
Example 3: Multiple accept states



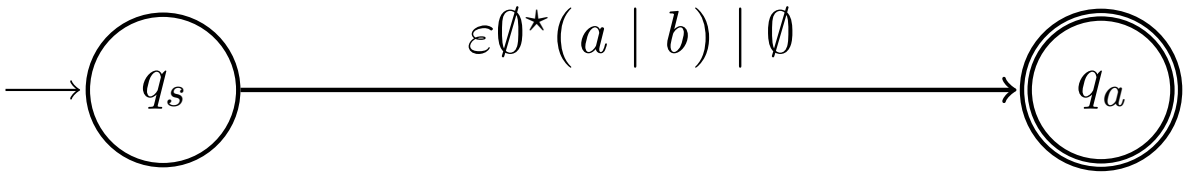
Eliminate q_0:

Simplify, to get the expected:

Example 3: Multiple accept states

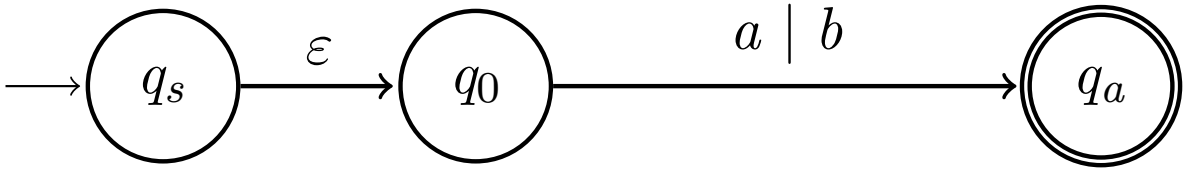


Eliminate q_0:

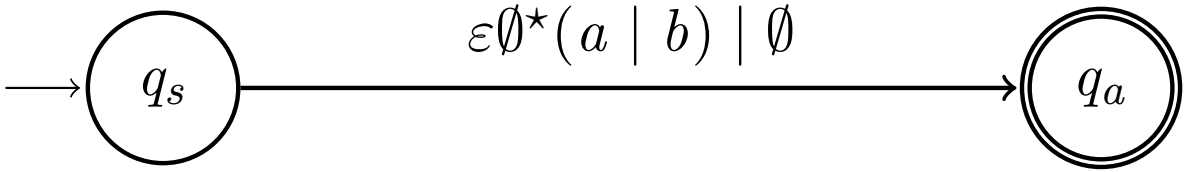


Simplify, to get the expected:

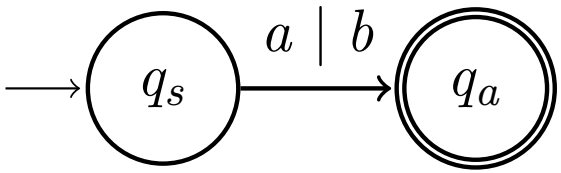
Example 3: Multiple accept states



Eliminate q_0:



Simplify, to get the expected:



Summary

Algorithm to convert an NFA to a RE:

1. Convert the NFA to a Generalised NFA (GNFA)
   - Only one start state, no incoming transitions
   - Only one accept state, no outgoing transitions
   - Transitions described by RE

Summary

Algorithm to convert an NFA to a RE:

1. Convert the NFA to a Generalised NFA (GNFA)
   - Only one start state, no incoming transitions
   - Only one accept state, no outgoing transitions
   - Transitions described by RE
2. Progressively *eliminate* all states between start and accept
   - For each pair (q_i, q_j) where $q_i, q_j \in Q \setminus \{q_{elim}\}$
   - Replace arrow $q_i \to q_j$ with $((R_1)(R_2)^\star(R_3) \mid (R_4))$, where
     - R_1 is the RE on transition $q_i \to q_{elim}$
     - R_2 is the RE on transition $q_{elim} \to q_{elim}$
     - R_3 is the RE on transition $q_{elim} \to q_j$
     - R_4 is the RE on transition $q_i \to q_j$

Summary

Algorithm to convert an NFA to a RE:

1. Convert the NFA to a Generalised NFA (GNFA)
   - Only one start state, no incoming transitions
   - Only one accept state, no outgoing transitions
   - Transitions described by RE
2. Progressively *eliminate* all states between start and accept
   - For each pair (q_i, q_j) where $q_i, q_j \in Q \setminus \{q_{elim}\}$
   - Replace arrow $q_i \to q_j$ with $((R_1)(R_2)^\star(R_3) \mid (R_4))$, where
     - R_1 is the RE on transition $q_i \to q_{elim}$
     - R_2 is the RE on transition $q_{elim} \to q_{elim}$
     - R_3 is the RE on transition $q_{elim} \to q_j$
     - R_4 is the RE on transition $q_i \to q_j$
3. The transition between the start and the accept state is now a regular expression describing L

Outline

- Regular Expressions
- Equivalence of FA and Regular Expressions
- **Proving if a language is, *or is not*, regular**

Proving if a language is regular *or not*

Recall the definition:
A language is *regular* if and only if there exists a finite automaton which recognises it.

Suppose a language is regular. How can we prove it?

Proving if a language is regular *or not*

Recall the definition:
A language is *regular* if and only if there exists a finite automaton which recognises it.

Suppose a language is regular. How can we prove it?

- Devise an NFA which recognises it, or
- Devise a DFA which recognises it, or
- Devise a RE which describes it

Proving if a language is regular *or not*

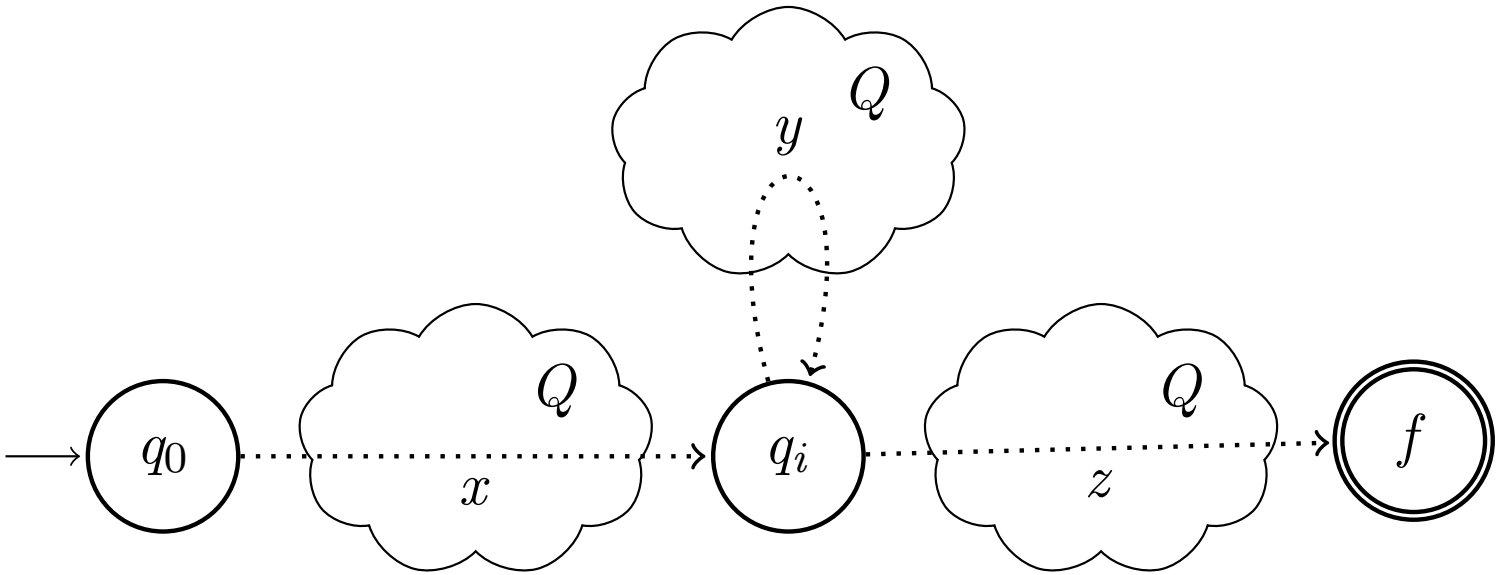
Recall the definition:
A language is *regular* if and only if there exists a finite automaton which recognises it.

Suppose a language is regular. How can we prove it?

- Devise an NFA which recognises it, or
- Devise a DFA which recognises it, or
- Devise a RE which describes it

What if the language was not regular? How can we *prove* that no suitable automata exists? We can use the *pumping lemma for regular langauges*

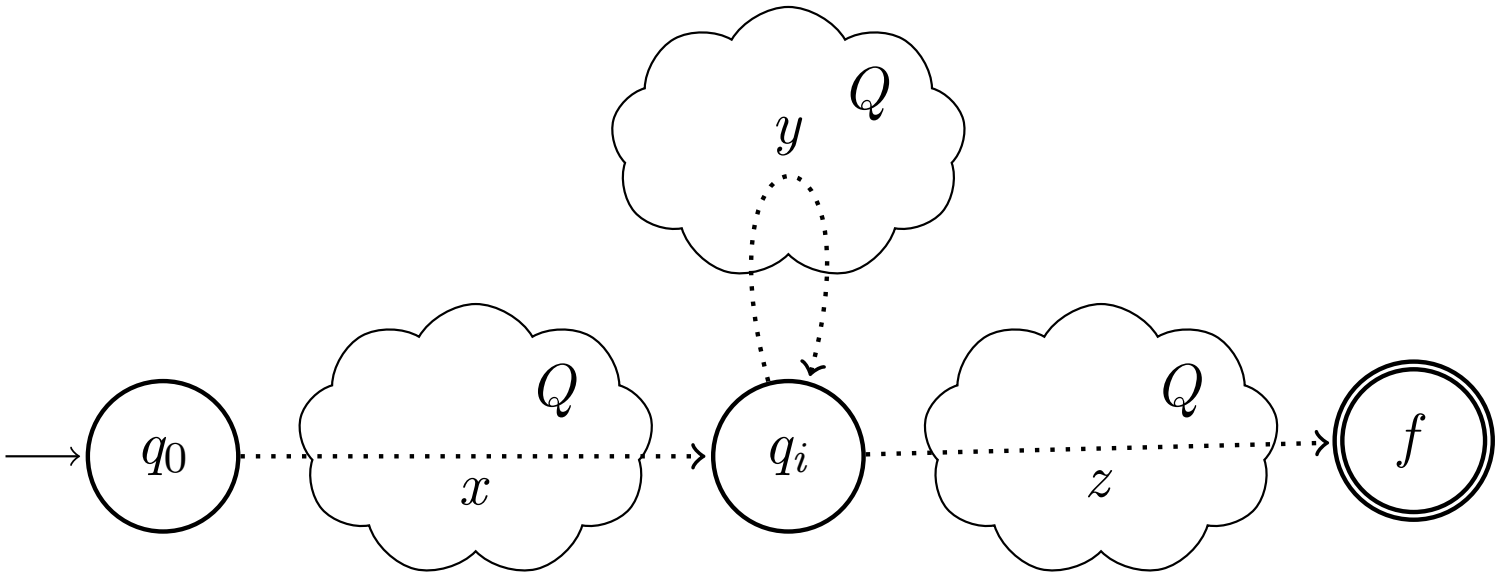
Pumping s using y



Suppose some string s exists, passing through M:

- From q_0 to q_i, using a string x, then
- from q_i back to q_i, using a string $y \neq \varepsilon$, then
- from q_i to some accept state $f \in F$, using a string z

Pumping s using y

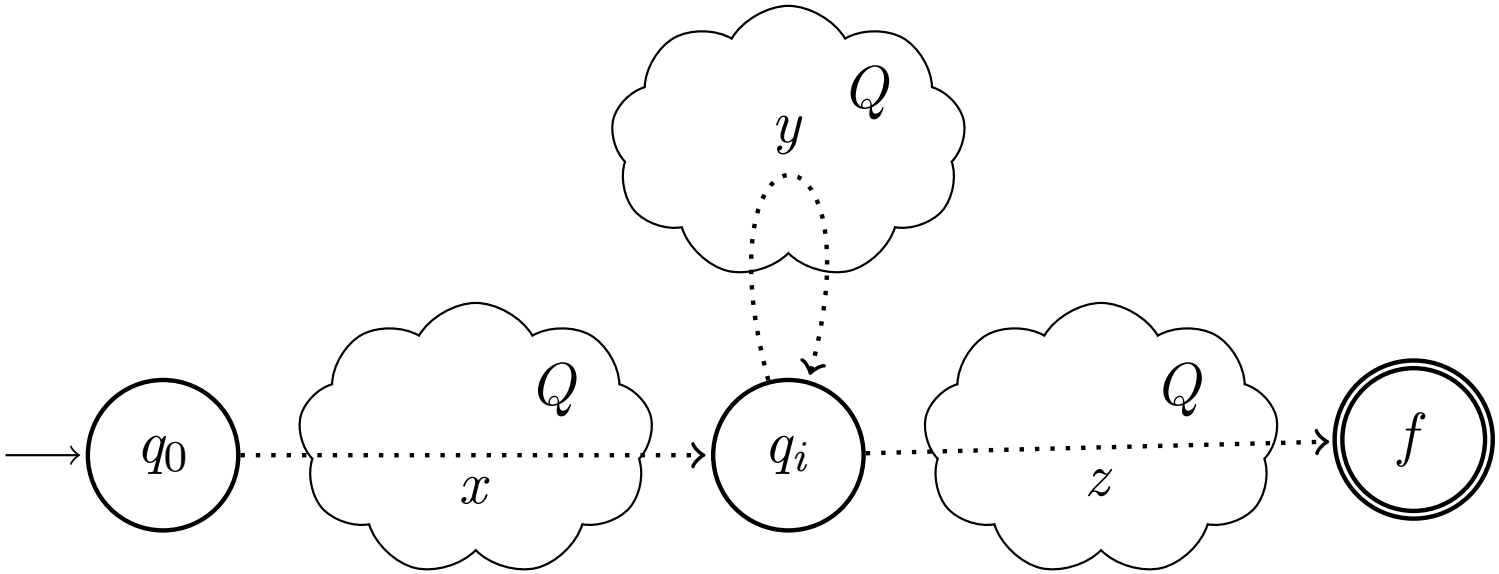


Suppose some string s exists, passing through M:

- From q_0 to q_i, using a string x, then
- from q_i back to q_i, using a string $y \neq \varepsilon$, then
- from q_i to some accept state $f \in F$, using a string z

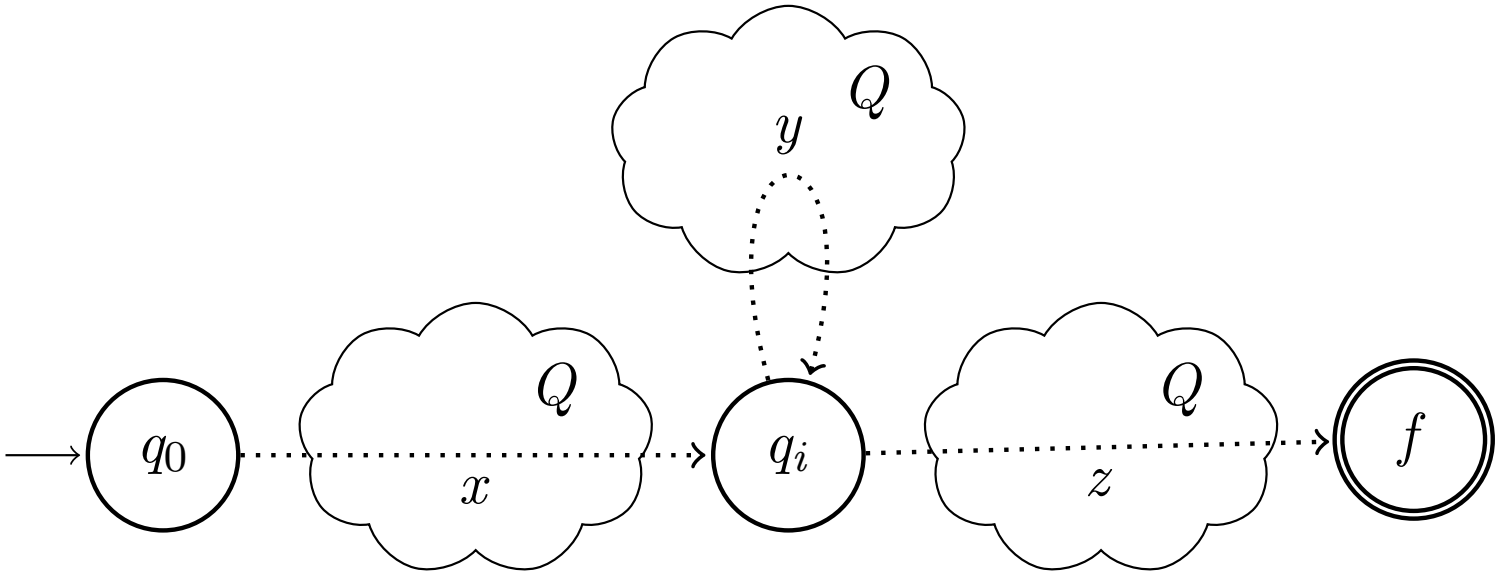
Then $s = xyz \in L$, and $xy^k z \in L$ for all $k \geq 0$
e.g. if $x = aa, y = b, z = c$, then $\{aac, aabc, aabbc, ...\} \subseteq L$

Long strings in finite automata



Suppose M is a DFA with n states, and $s \in L(M)$, but $|s| > n + 1$

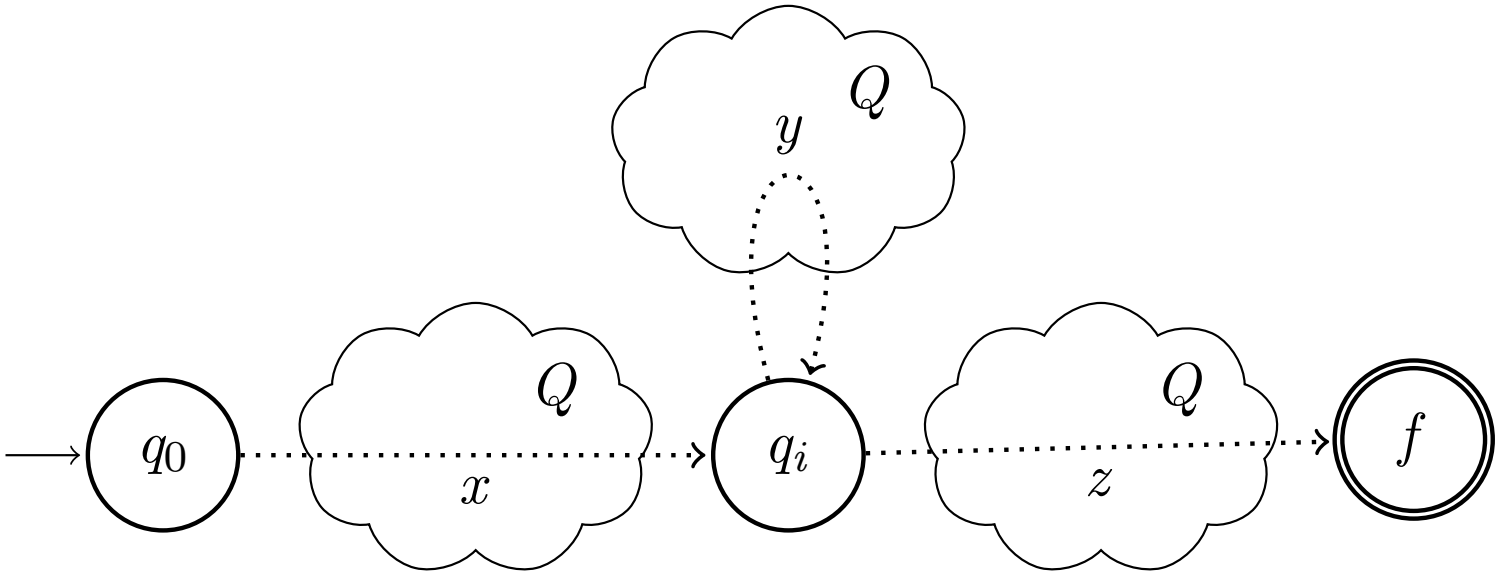
Long strings in finite automata



Suppose M is a DFA with n states, and $s \in L(M)$, but $|s| > n + 1$

Then there exists at least one state which was visited more than once (q_i), and a substring $y \neq \varepsilon$ corresponding to the path followed between the two of those visits.

Long strings in finite automata



Suppose M is a DFA with n states, and $s \in L(M)$, but $|s| > n + 1$

Then there exists at least one state which was visited more than once (q_i), and a substring $y \neq \varepsilon$ corresponding to the path followed between the two of those visits.

Hence we can *pump* s to find other strings in the language

Finite automata, infinite languages

All finite languages are regular. (Why?)

Finite automata, infinite languages

All finite languages are regular. (Why?)

Suppose L is an *infinite* regular language.

- L is regular, so a DFA M exists which recognises it.
- L is an infinite set over a finite alphabet, so it must include arbitrarily long words.
- Therefore words exist which can be *pumped*

Pumping lemma for regular languages

If L is a regular language, then there exists a number p (the *pumping length*) such that for any string $s \in L$ of length at least p, then s may be divided into three pieces, $s = xyz$, such that:

1. $xy^k z \in L$ for all $k \geq 0$ (i.e. we can use y to *pump* the string)
2. $|y| > 0$ (i.e. $y \neq \varepsilon$)
3. $|xy| \leq p$

If a language does not satisfy this lemma, then it cannot be regular

Using the pumping lemma

We can (only) use the pumping lemma to prove that a language is *not* regular

Proof by contradiction:

1. Assume that the langage is regular
2. Choose an appropriate string in the language
   - This is often the hardest part of the proof
   - We must find one which will allow us to:
3. Apply the lemma to find a contradiction
4. Thereby deduce that the assumption is false
   - i.e. The language cannot be regular

Example

Show that $L = \{a^n b^n | n \geq 0\}$ is not regular

Example

Show that $L = \{a^n b^n | n \geq 0\}$ is not regular

1. Assumption: L is regular.

Example

Show that $L = \{a^n b^n | n \geq 0\}$ is not regular

1. Assumption: L is regular.
2. Then there exists some $p \geq 0$ satisfying the pumping lemma.

Example

Show that $L = \{a^n b^n | n \geq 0\}$ is not regular

1. Assumption: L is regular.
2. Then there exists some $p \geq 0$ satisfying the pumping lemma.
3. Let $s = a^p b^p$ (note: the same p as above!)

Example

Show that $L = \{a^n b^n | n \geq 0\}$ is not regular

1. Assumption: L is regular.
2. Then there exists some $p \geq 0$ satisfying the pumping lemma.
3. Let $s = a^p b^p$ (note: the same p as above!)
4. $|s| \geq p$, therefore we can write $s = xyz$ such that
   - 4.1 $xy^k z \in L$ for all $k \geq 0$ (i.e. we can *pump* y)
   - 4.2 $|y| > 0$
   - 4.3 $|xy| \leq p$

Example

Show that $L = \{a^n b^n | n \geq 0\}$ is not regular

1. Assumption: L is regular.
2. Then there exists some $p \geq 0$ satisfying the pumping lemma.
3. Let $s = a^p b^p$ (note: the same p as above!)
4. $|s| \geq p$, therefore we can write $s = xyz$ such that
   - 4.1 $xy^k z \in L$ for all $k \geq 0$ (i.e. we can *pump* y)
   - 4.2 $|y| > 0$
   - 4.3 $|xy| \leq p$
5. Therefore $y = a^i$ for some $0 < i \leq p$ (at least one a, no b's)

Example

Show that $L = \{a^n b^n | n \geq 0\}$ is not regular

1. Assumption: L is regular.
2. Then there exists some $p \geq 0$ satisfying the pumping lemma.
3. Let $s = a^p b^p$ (note: the same p as above!)
4. $|s| \geq p$, therefore we can write $s = xyz$ such that
   - 4.1 $xy^k z \in L$ for all $k \geq 0$ (i.e. we can *pump* y)
   - 4.2 $|y| > 0$
   - 4.3 $|xy| \leq p$
5. Therefore $y = a^i$ for some $0 < i \leq p$ (at least one a, no b's)
6. Let $k = 2$. Then $xy^k z$ has more a's than b's, so $xy^k z \notin L$

Example

Show that $L = \{a^n b^n | n \geq 0\}$ is not regular

1. Assumption: L is regular.
2. Then there exists some $p \geq 0$ satisfying the pumping lemma.
3. Let $s = a^p b^p$ (note: the same p as above!)
4. $|s| \geq p$, therefore we can write $s = xyz$ such that
   - 4.1 $xy^k z \in L$ for all $k \geq 0$ (i.e. we can *pump* y)
   - 4.2 $|y| > 0$
   - 4.3 $|xy| \leq p$
5. Therefore $y = a^i$ for some $0 < i \leq p$ (at least one a, no b's)
6. Let $k = 2$. Then $xy^k z$ has more a's than b's, so $xy^k z \notin L$
7. Lines 6 and 4.1 form a contradiction, therefore the assumption is false (i.e. L is *not* regular.)

Regular Expressions

- What they are and how to use them
- Regular operations

Equivalence of FA and Regular Expressions

- Convert an NFA to a RE
- Convert a RE to an NFA

Proving if a language is, *or is not*, regular

- Prove regularity by finding a DFA, NFA, or RE
- Prove non-regularity by finding a contradiction in the Pumping Lemma