فصل سوم

زبان‌های منظم و گرامرهای منظم (۱)

Regular Languages and Regular Grammars (1)

کاظم فولادی

kazim@fouladi.ir

دانشکده‌ی مهندسی برق و کامپیوتر
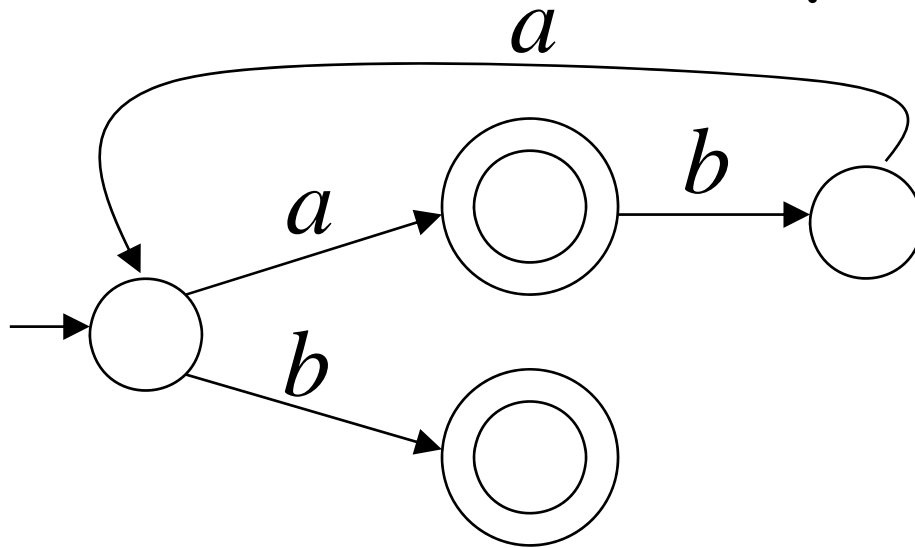
دانشگاه تهران

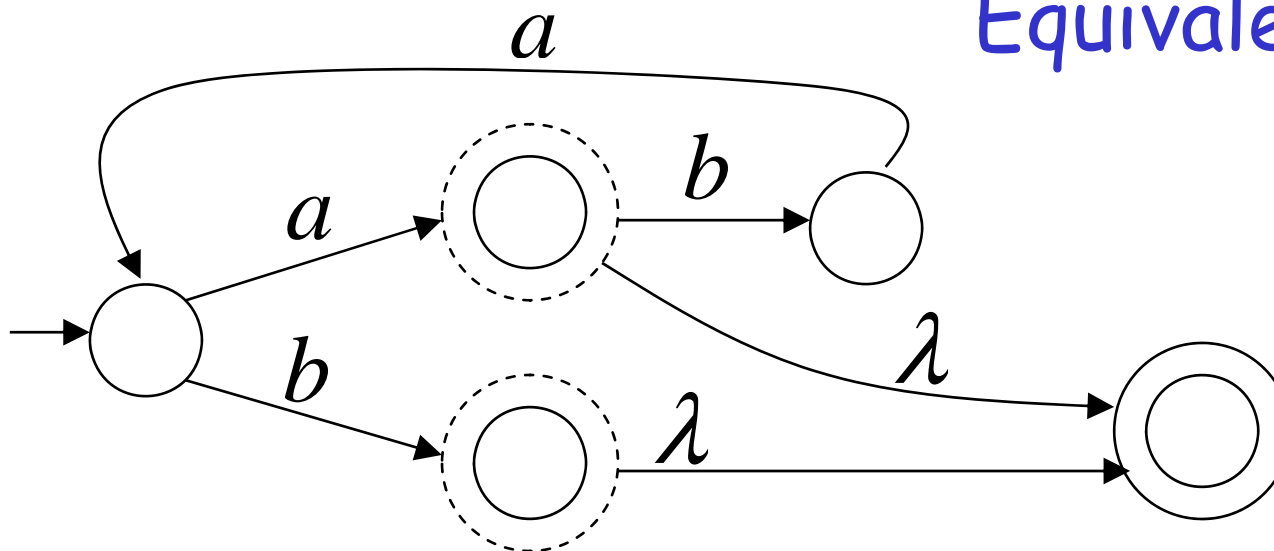# Single Final State
for NFAs and DFAs

Observation

Any Finite Automaton (NFA or DFA)

can be converted to an equivalent NFA
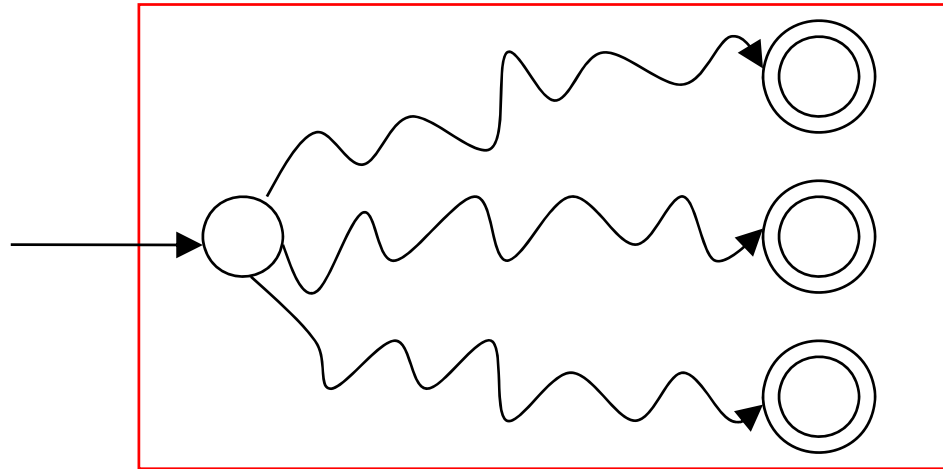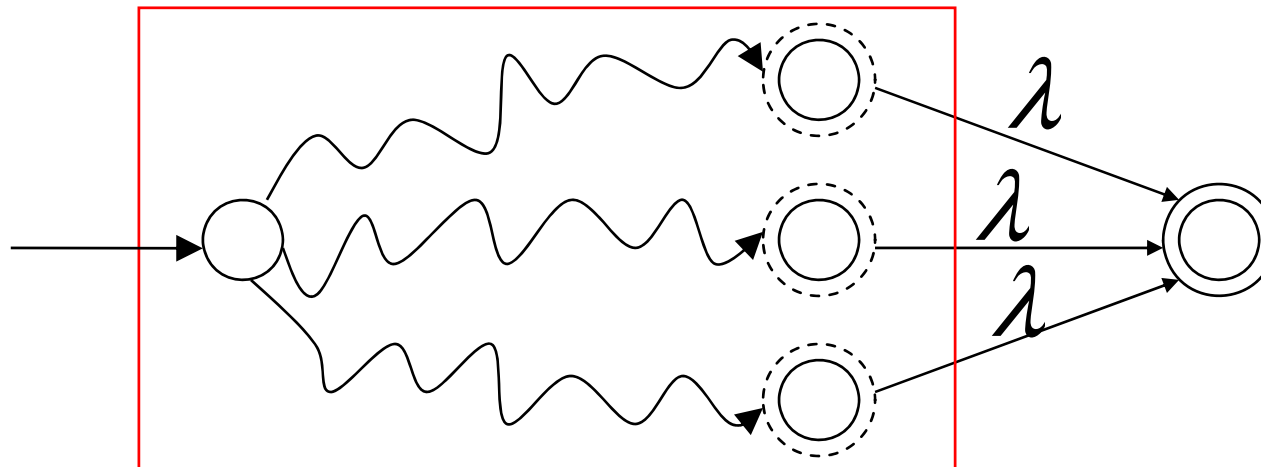
with a single final state

Example

NFA

Equivalent NFA
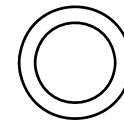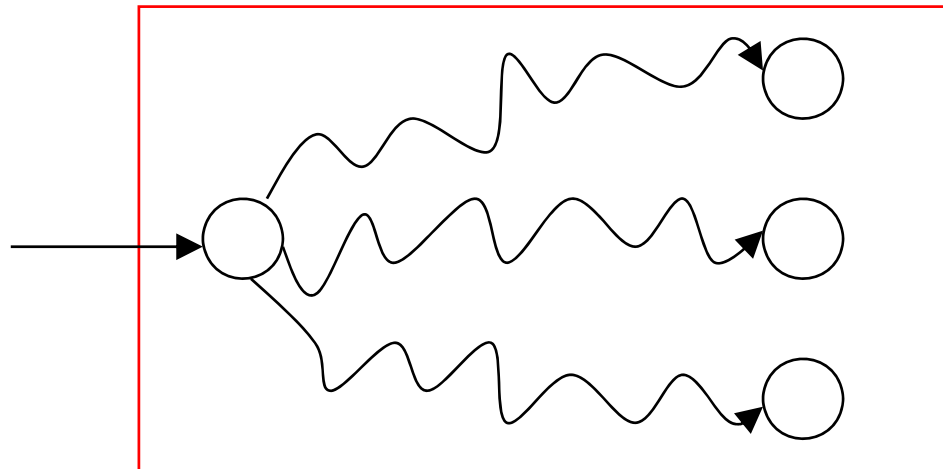
In General

NFA

Equivalent NFA

Single
final state

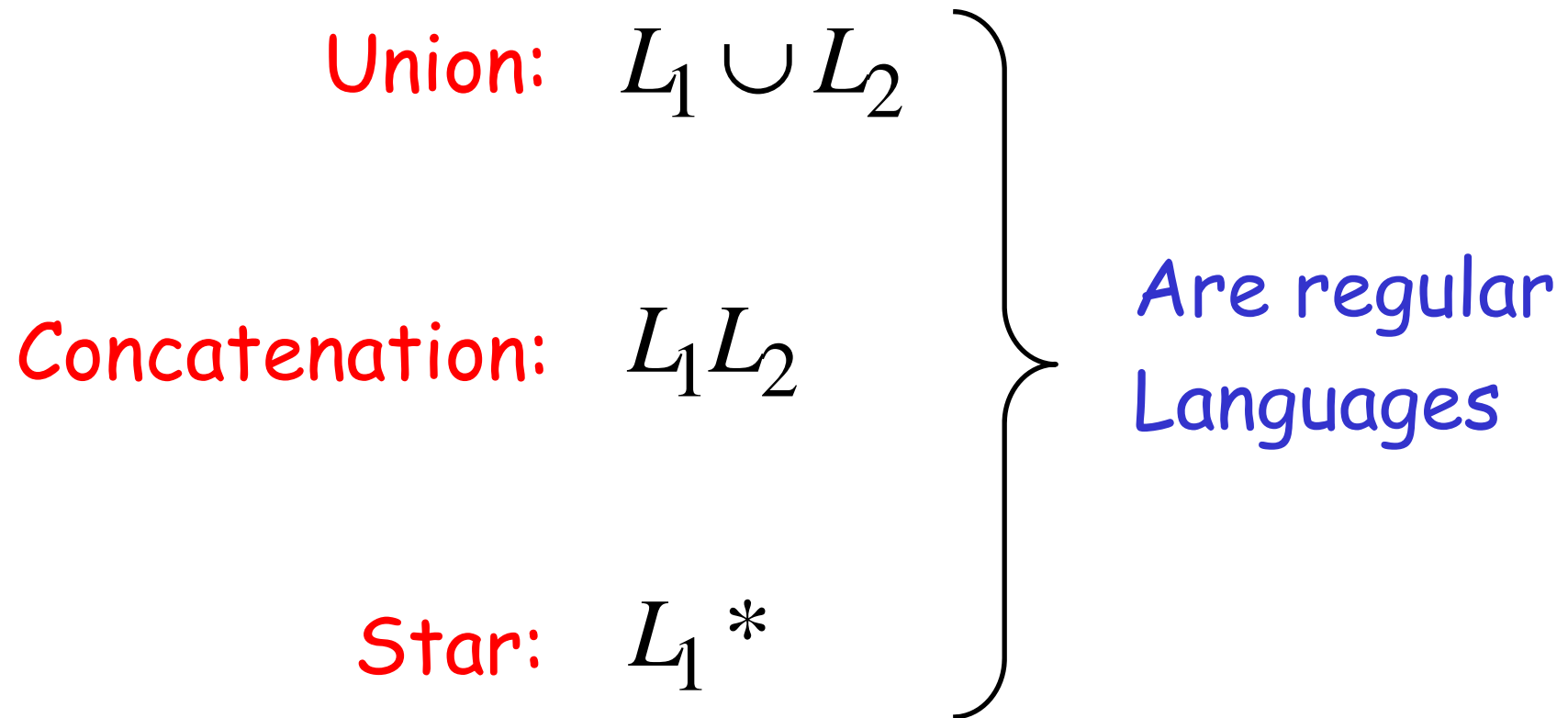# Extreme Case

NFA without final state

Add a final state
Without transitions

Some Properties of Regular Languages

Properties

For regular languages L_1 and L_2
we will prove that:

Union: $L_1 \cup L_2$

Concatenation: $L_1 L_2$

Star: $L_1{}^*$

Are regular
Languages

We Say:

Regular languages are **closed under**

Union: $L_1 \cup L_2$

Concatenation: $L_1 L_2$

Star: $L_1{}^*$

Regular language L_1 Regular language L_2
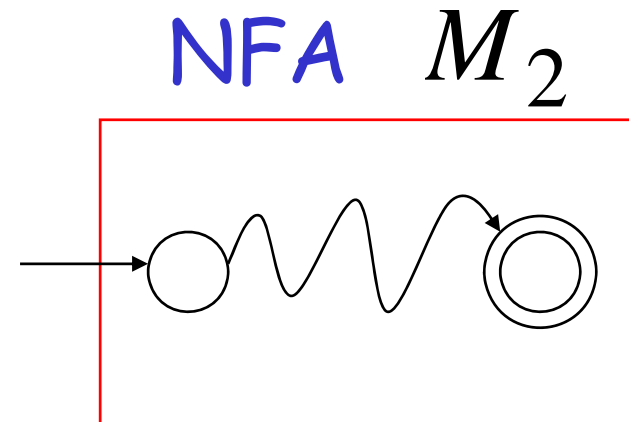
$$L(M_1) = L_1$$ $$L(M_2) = L_2$$

NFA M_1
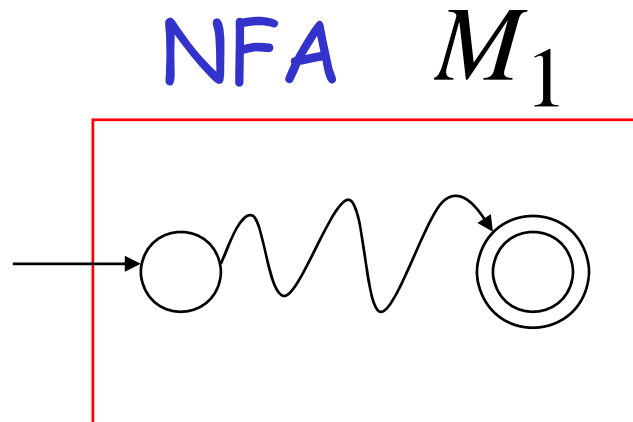
Single final state

NFA M_2

Single final state

Example

$$M_1$$

$$L_1 = \{a^n b\}$$

$$M_2$$
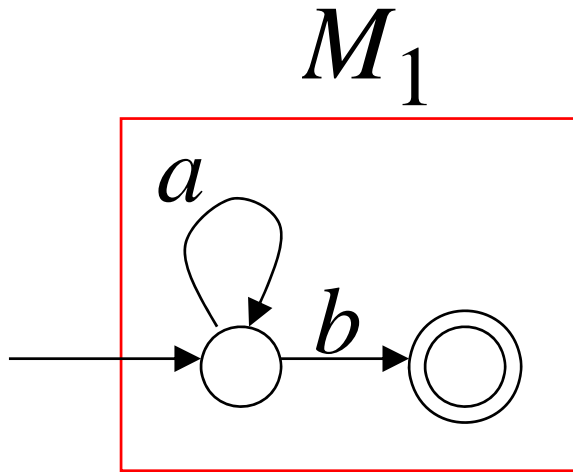
$$L_2 = \{ba\}$$

Union

NFA for $L_1 \cup L_2$

M_1

M_2

λ
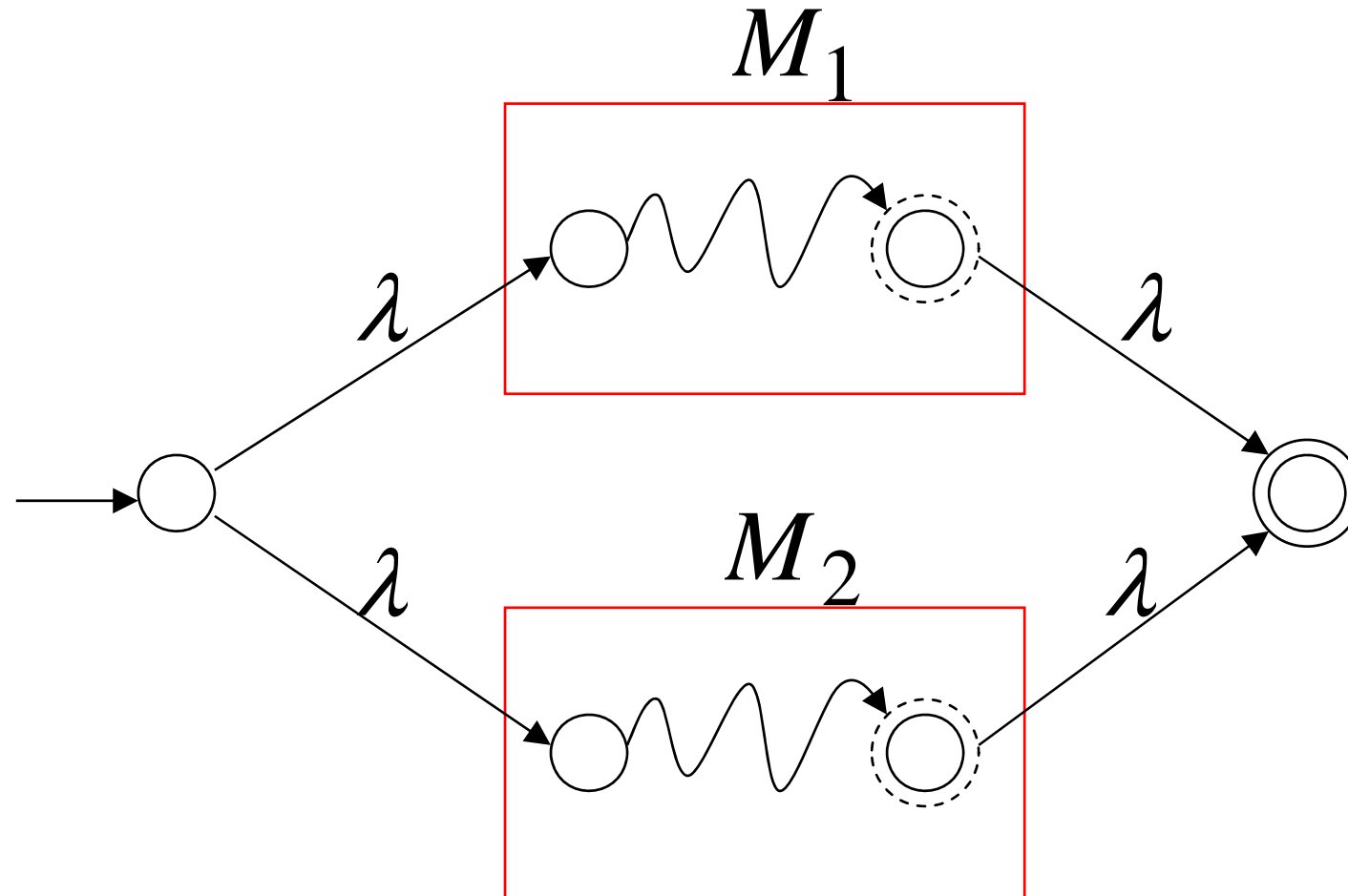
Example

NFA for $L_1 \cup L_2 = \{a^n b\} \cup \{ba\}$

$$L_1 = \{a^n b\}$$

$$L_2 = \{ba\}$$

Concatenation

NFA for $L_1 L_2$
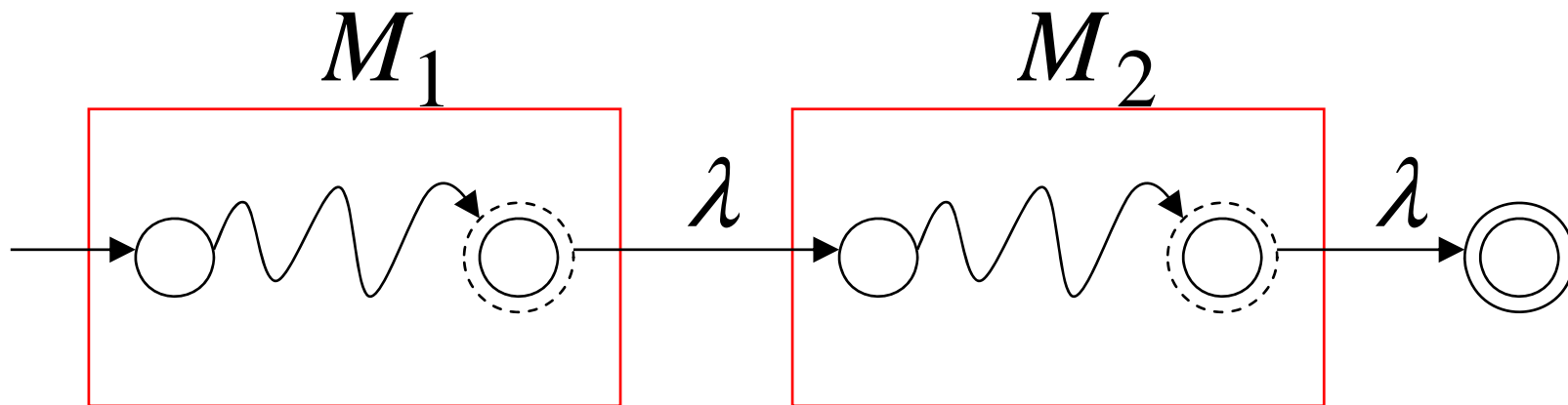
Example

NFA for $L_1 L_2 = \{a^n b\}\{ba\} = \{a^n bba\}$

$L_1 = \{a^n b\}$

$L_2 = \{ba\}$
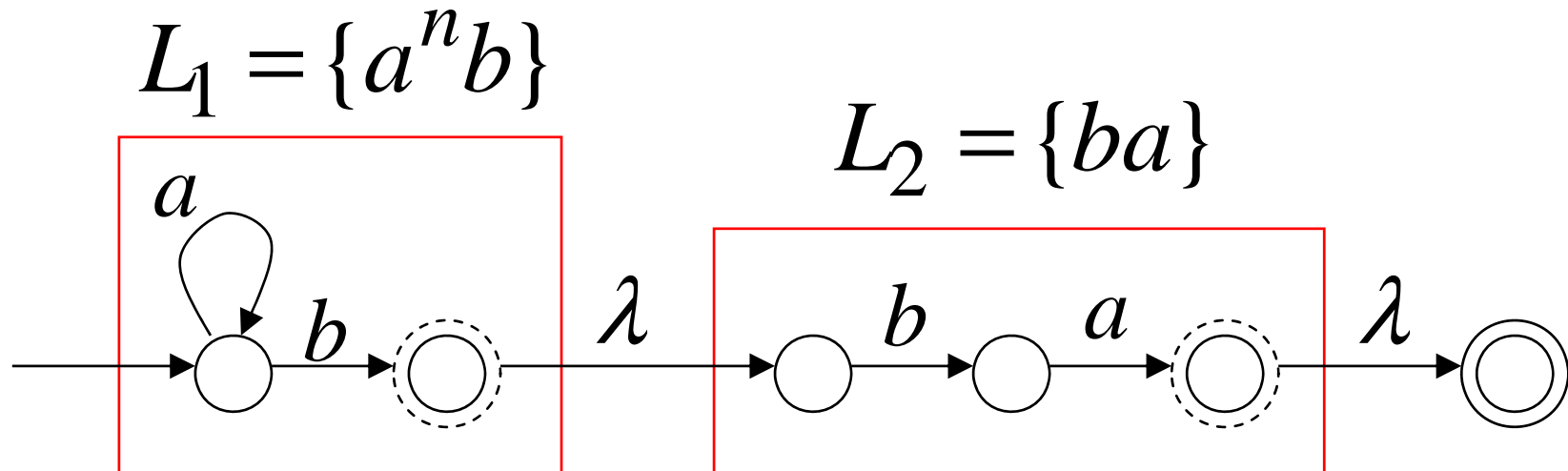
Star Operation

NFA for $L_1 *$

λ

M_1

$\lambda \in L_1 *$

λ

λ

λ
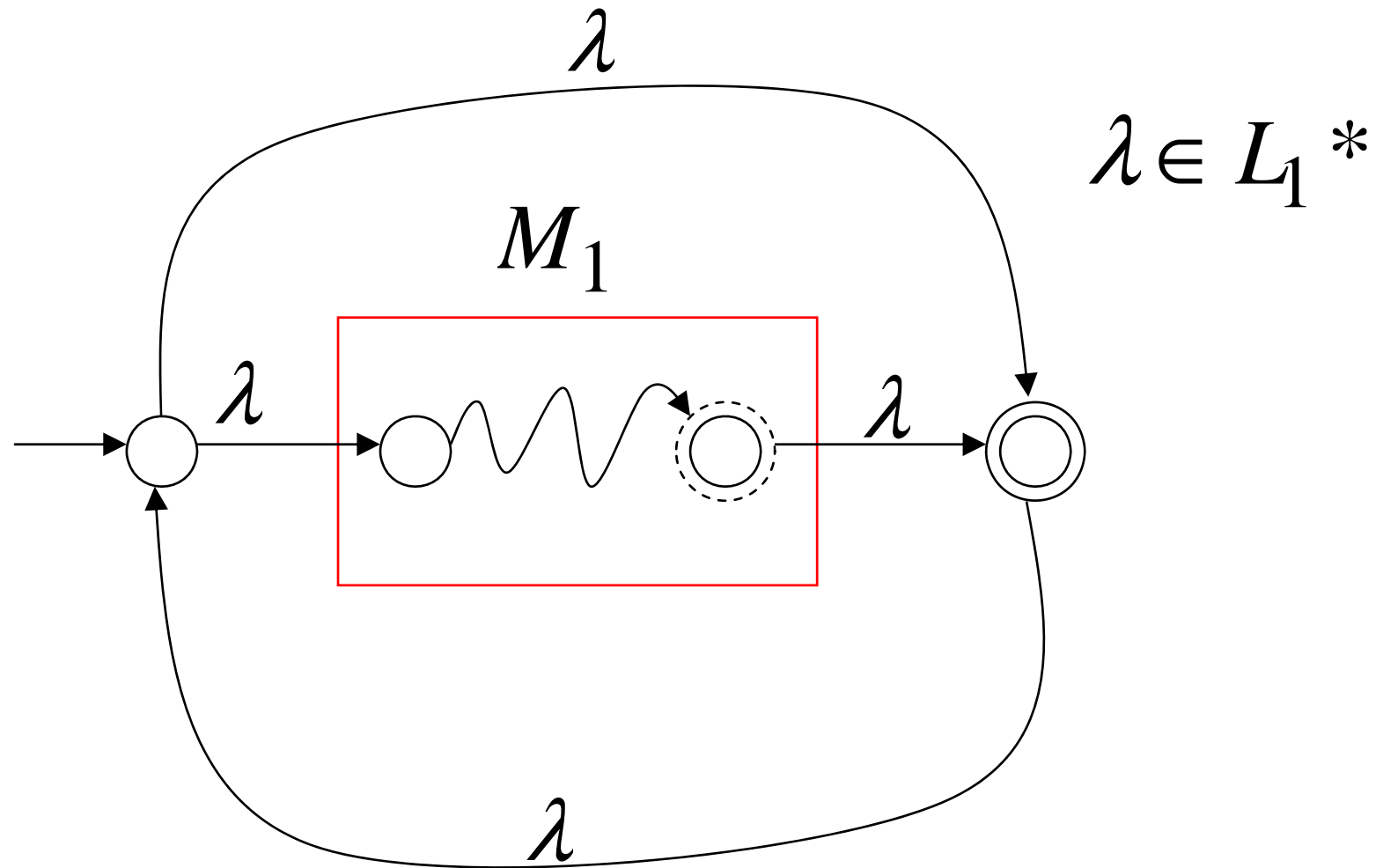
Example

NFA for $L_1* = \{a^n b\}*$

$$L_1 = \{a^n b\}$$
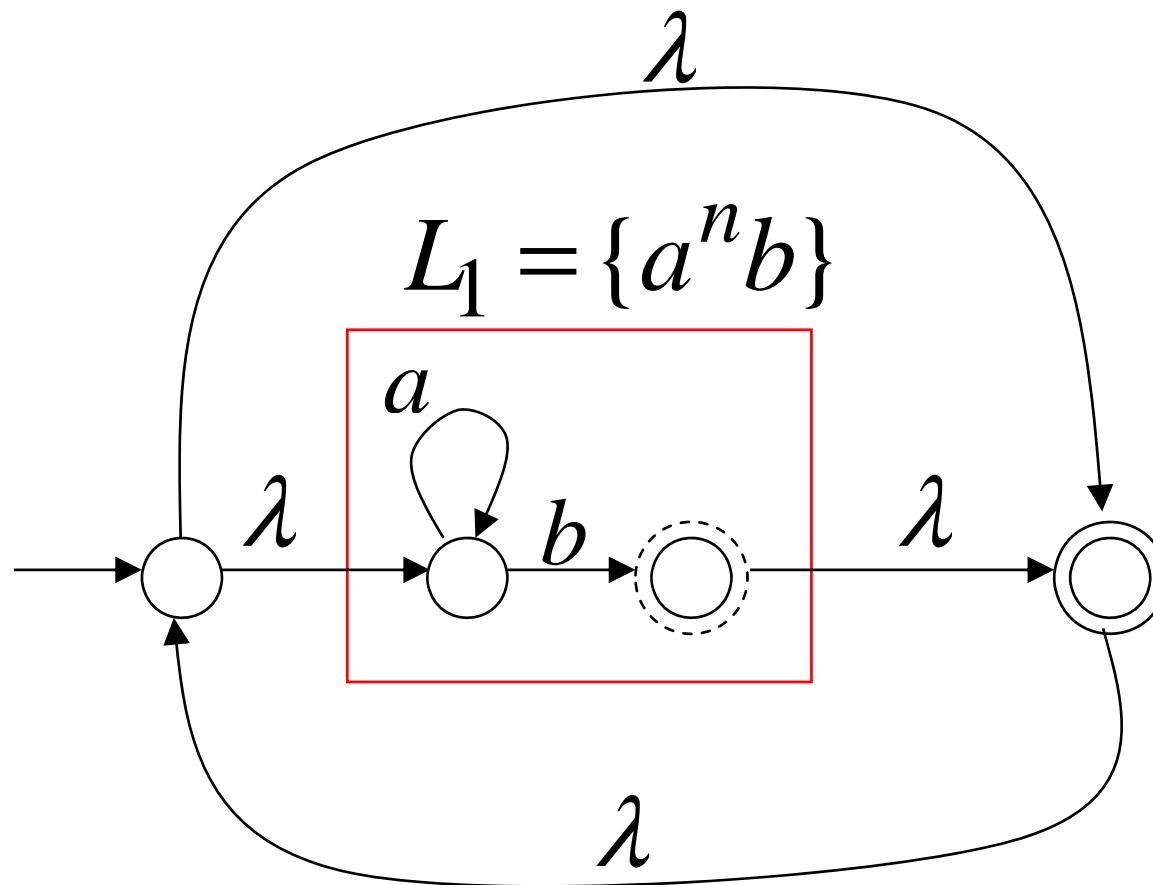
Regular Expressions

Regular Expressions

Regular expressions
describe regular languages

Example: $(a + b \cdot c)^*$

describes the language

$$\{a, bc\}^* = \{\lambda, a, bc, aa, abc, bca, ...\}$$

Recursive Definition

Primitive regular expressions: $\varnothing, \quad \lambda, \quad \alpha$

Given regular expressions r_1 and r_2

$$
\left.
\begin{aligned}
&r_1 + r_2 \\
&r_1 \cdot r_2 \\
&r_1{}^* \\
&(r_1)
\end{aligned}
\right\}
\text{Are regular expressions}
$$

Examples

A regular expression: $(a + b \cdot c)^* \cdot (c + \varnothing)$

Not a regular expression: $(a + b +)$

Languages of Regular Expressions

$L(r):$ language of regular expression r

Example

$$L((a + b \cdot c)*) = \{\lambda, a, bc, aa, abc, bca, ...\}$$

Definition

For primitive regular expressions:

$$L(\varnothing) = \varnothing$$

$$L(\lambda) = \{\lambda\}$$

$$L(a) = \{a\}$$

Definition (continued)

For regular expressions r_1 and r_2

$$L(r_1 + r_2) = L(r_1) \cup L(r_2)$$

$$L(r_1 \cdot r_2) = L(r_1)\, L(r_2)$$

$$L(r_1 *) = (L(r_1))*$$

$$L((r_1)) = L(r_1)$$

Example

Regular expression: $(a+b) \cdot a*$

$$L((a+b) \cdot a*) = L((a+b)) \, L(a*)$$
$$= L(a+b) \, L(a*)$$
$$= (L(a) \cup L(b)) \, (L(a))*$$
$$= (\{a\} \cup \{b\}) \, (\{a\})*$$
$$= \{a,b\} \{\lambda, a, aa, aaa, ...\}$$
$$= \{a, aa, aaa, ..., b, ba, baa, ...\}$$

Example

Regular expression $r = (a+b)^*(a+bb)$

$$L(r) = \{a, bb, aa, abb, ba, bbb, ...\}$$

Example

Regular expression $r = (aa)*(bb)*b$

$$L(r) = \{a^{2n}b^{2m}b : \quad n, m \geq 0\}$$

Example

Regular expression $r = (0+1)*00(0+1)*$

$L(r)$ = { all strings with at least two consecutive 0 }

Example

Regular expression $r = (1 + 01) * (0 + \lambda)$

$L(r)$ = { all strings without
          two consecutive 0 }

Equivalent Regular Expressions

Definition:

Regular expressions r_1 and r_2

are **equivalent** if $L(r_1) = L(r_2)$

Example

L = { all strings without
two consecutive 0 }

$$r_1 = (1+01)*(0+\lambda)$$

$$r_2 = (1*011*)*(0+\lambda)+1*(0+\lambda)$$

$$L(r_1) = L(r_2) = L \implies r_1 \text{ and } r_2$$

are equivalent
regular expr.

Regular Expressions
and
Regular Languages

Theorem

$$\left\{ \begin{array}{l} \text{Languages} \\ \text{Generated by} \\ \text{Regular Expressions} \end{array} \right\} = \left\{ \begin{array}{l} \text{Regular} \\ \text{Languages} \end{array} \right\}$$

Theorem - Part 1

$$\left\{ \begin{array}{l} \text{Languages} \\ \text{Generated by} \\ \text{Regular Expressions} \end{array} \right\} \subseteq \left\{ \begin{array}{l} \text{Regular} \\ \text{Languages} \end{array} \right\}$$

1. For any regular expression r
   the language $L(r)$ is regular

Theorem - Part 2

$$\left\{\begin{array}{l}\text{Languages} \\ \text{Generated by} \\ \text{Regular Expressions}\end{array}\right\} \supseteq \left\{\begin{array}{l}\text{Regular} \\ \text{Languages}\end{array}\right\}$$

2. For any regular language L there is
   a regular expression r with $L(r) = L$
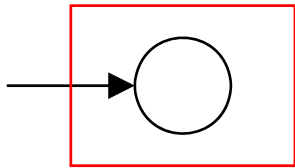
Proof - Part 1

1. For any regular expression r
   the language $L(r)$ is regular
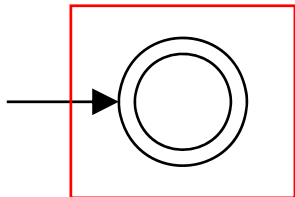
   Proof by induction on the size of r
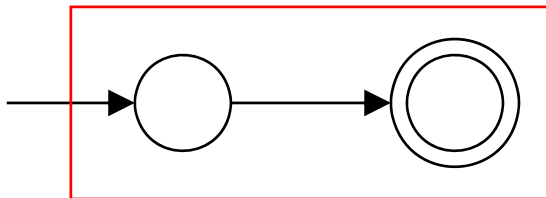
Induction Basis

Primitive Regular Expressions: $\varnothing, \quad \lambda, \quad \alpha$

NFAs

$$L(M_1) = \varnothing = L(\varnothing)$$

$$L(M_2) = \{\lambda\} = L(\lambda)$$

$$L(M_3) = \{a\} = L(a)$$

regular languages

Inductive Hypothesis

Assume
for regular expressions r_1 and r_2
that
$L(r_1)$ and $L(r_2)$ are regular languages

Inductive Step

We will prove:

$$L(r_1 + r_2)$$

$$L(r_1 \cdot r_2)$$

$$L(r_1 *)$$

$$L((r_1))$$

Are regular Languages

By definition of regular expressions:

$$L(r_1 + r_2) = L(r_1) \cup L(r_2)$$

$$L(r_1 \cdot r_2) = L(r_1)\, L(r_2)$$

$$L(r_1 *) = (L(r_1)) *$$

$$L((r_1)) = L(r_1)$$

By inductive hypothesis we know:

$L(r_1)$ and $L(r_2)$ are regular languages

We also know:

Regular languages are closed under

union	$L(r_1) \cup L(r_2)$
concatenation	$L(r_1)\,L(r_2)$
star	$(L(r_1))*$

Therefore:

$$L(r_1 + r_2) = L(r_1) \cup L(r_2)$$

$$L(r_1 \cdot r_2) = L(r_1)\, L(r_2)$$

$$L(r_1 *) = (L(r_1)) *$$

Are regular languages

And trivially:

$$L((r_1)) \quad \text{is a regular language}$$

Proof – Part 2

2. For any regular language L there is
a regular expression r with $L(r) = L$

Proof by construction of regular expression
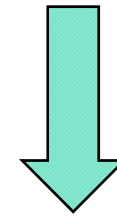
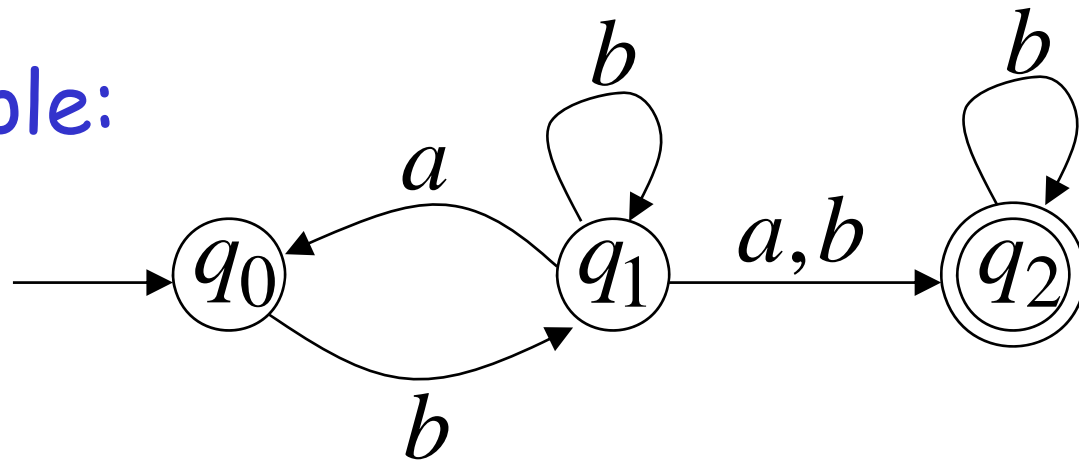Since L is regular take the
NFA M that accepts it

$$L(M) = L$$
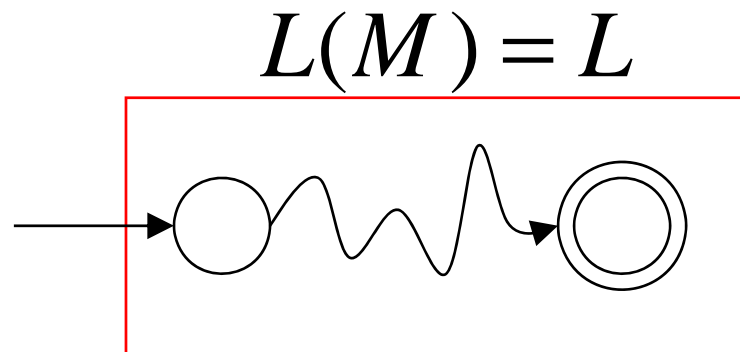
Single final state

From M construct the equivalent
Generalized Transition Graph
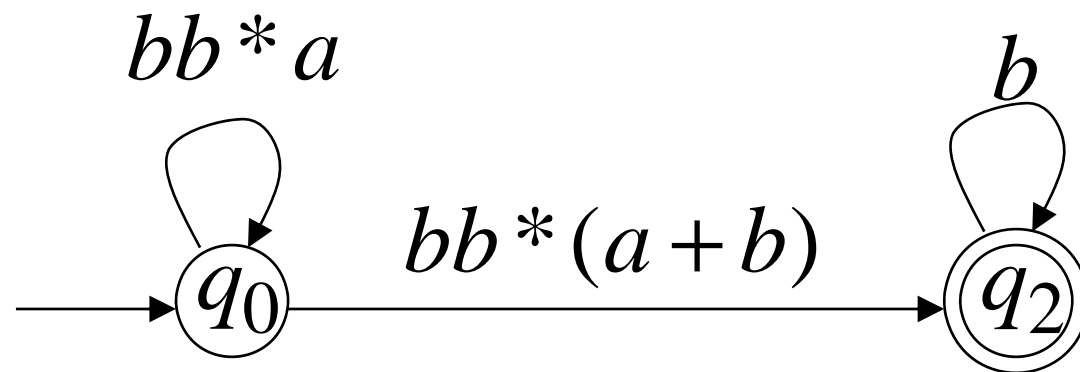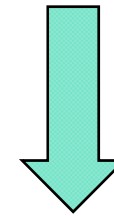   transition labels
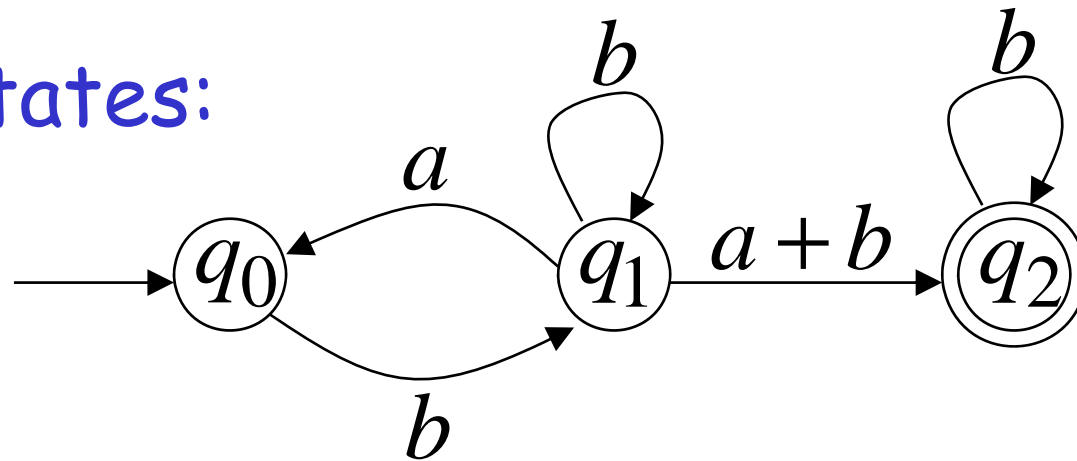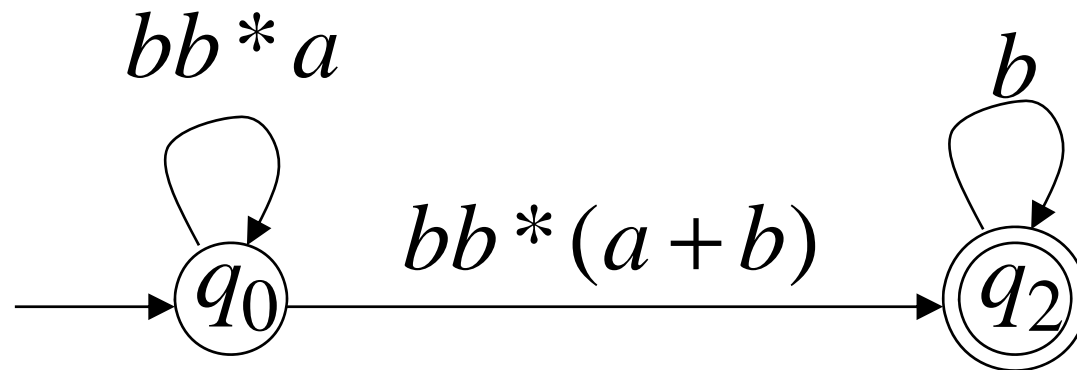   are regular expressions
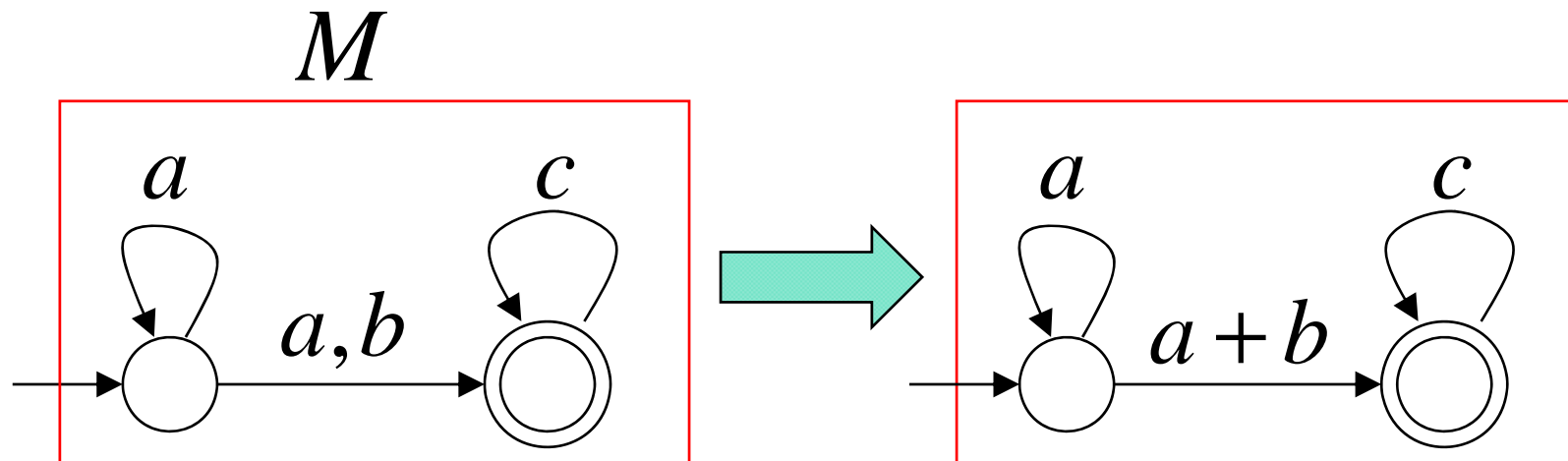
Example:

Another Example:
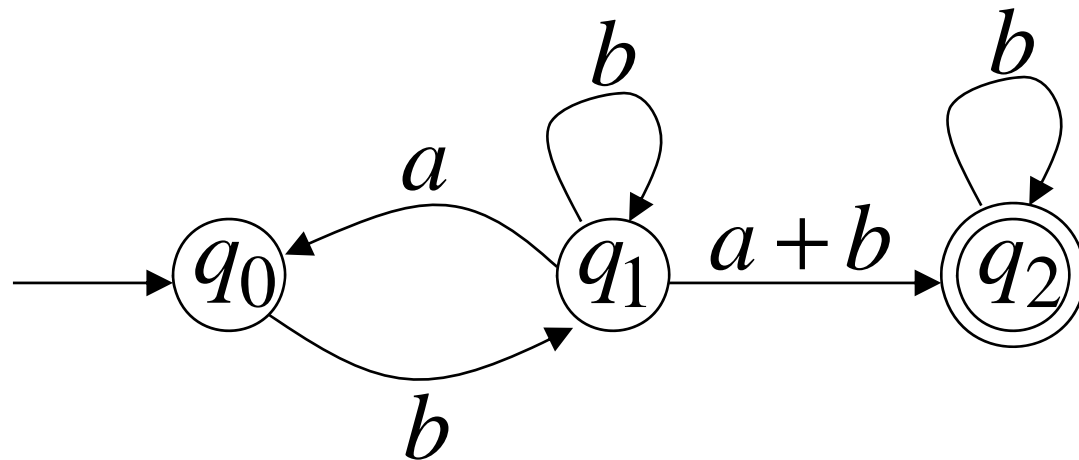
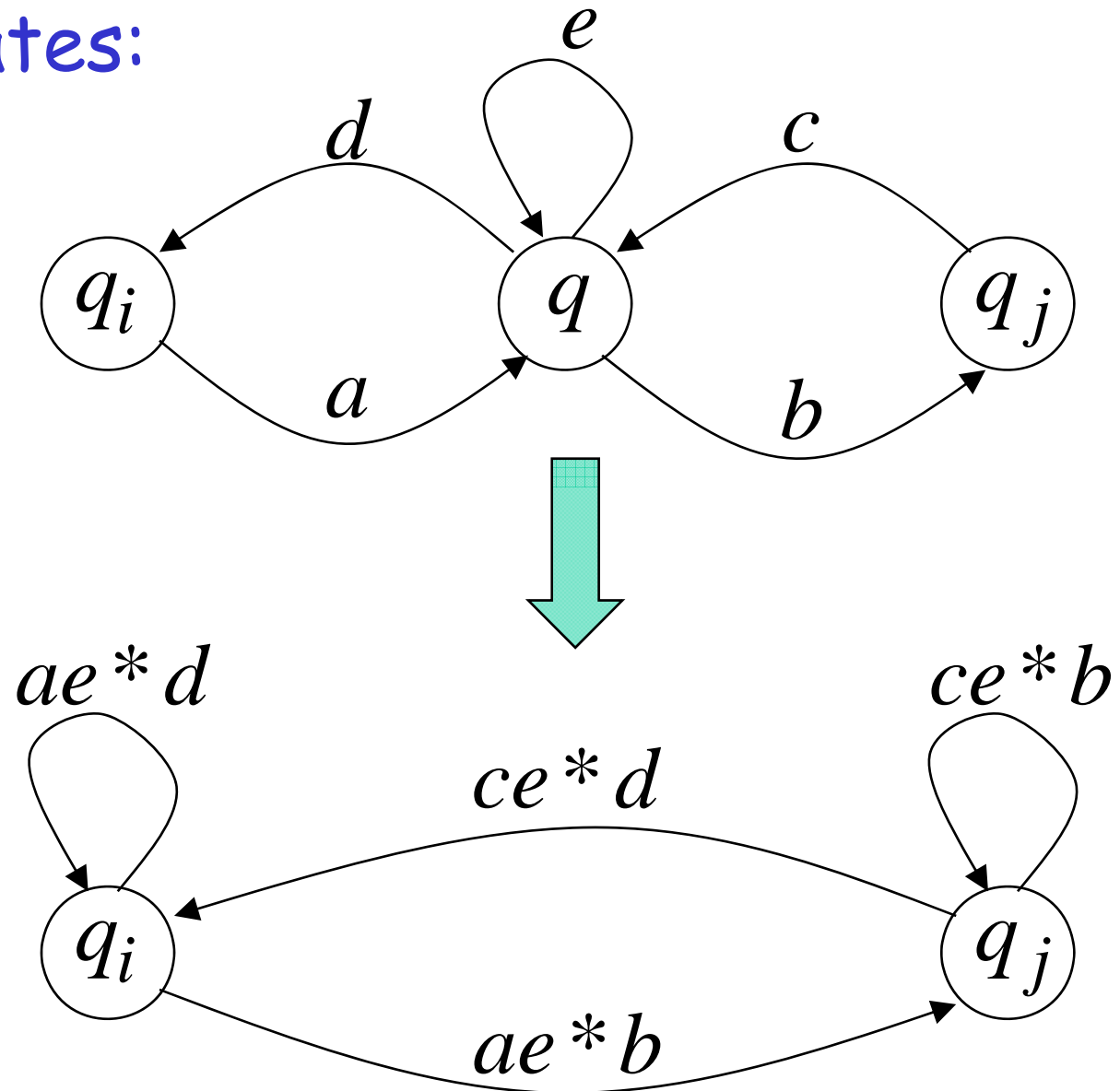# Reducing the states:

Resulting Regular Expression:
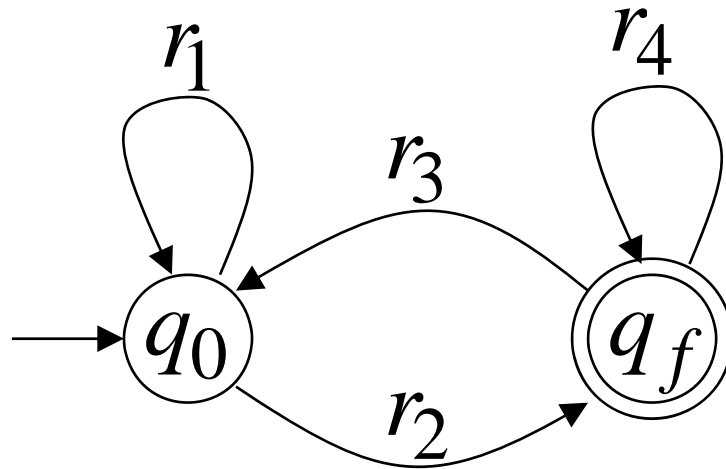

$$r = (bb*a)*bb*(a+b)b*$$

$$L(r) = L(M) = L$$

In General

Removing states:

Obtaining the final regular expression:

$$r = r_1 * r_2(r_4 + r_3 r_1 * r_2) *$$

$$L(r) = L(M) = L$$