

# Regular Expressions

# Regular Expressions

Regular expressions  
describe regular languages

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

describes the language

$$\{a, bc\}^* = \{\varepsilon, a, bc, aa, abc, bca, \dots\}$$

# Recursive Definition

Primitive regular expressions:  $\emptyset$ ,  $\varepsilon$ ,  $a$

Given regular expressions  $r_1$  and  $r_2$

$$r_1 + r_2$$

$$r_1 \cdot r_2$$

$$r_1^*$$

$$(r_1)$$

Are regular expressions

# Examples

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

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

# Languages of Regular Expressions

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

Example

$$L((a + b \cdot c)^*) = \{\varepsilon, a, bc, aa, abc, bca, \dots\}$$

# Definition

For primitive regular expressions:

$$L(\emptyset) = \emptyset$$

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

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

# 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:  $(a + b) \cdot a^*$

$$\begin{aligned} L((a + b) \cdot a^*) &= L((a + b)) L(a^*) \\ &= L(a + b) L(a^*) \\ &= (L(a) \cup L(b)) (L(a))^* \\ &= (\{a\} \cup \{b\}) (\{a\})^* \\ &= \{a, b\} \{\varepsilon, a, aa, aaa, \dots\} \\ &= \{a, aa, aaa, \dots, b, ba, baa, \dots\} \end{aligned}$$



# Example

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

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

# Example

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

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

# Example

Regular expression

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

$$L(r) = \{ \text{all strings containing substring } 00 \}$$

# Example

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

$L(r) = \{ \text{all strings without substring } 00 \}$

# Equivalent Regular Expressions

Definition:

Regular expressions  $r_1$  and  $r_2$


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

# Example

$L = \{ \text{all strings without substring } 00 \}$

$$r_1 = (1 + 01)^* (0 + \varepsilon)$$

$$r_2 = (1^* 0 1 1^*)^* (0 + \varepsilon) + 1^* (0 + \varepsilon)$$

$L(r_1) = L(r_2) = L$    $r_1$  and  $r_2$   
are equivalent  
regular expressions

# Regular Expressions and Regular Languages

# Theorem

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



## Proof:

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

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

# Proof - Part 1

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

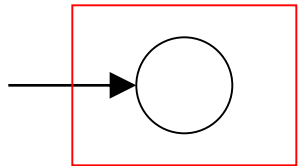
For any regular expression  $r$   
the language  $L(r)$  is regular

Proof by induction on the size of  $r$

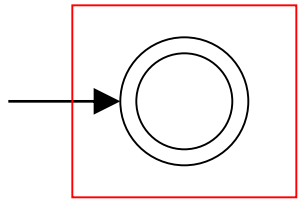
# Induction Basis

Primitive Regular Expressions:  $\emptyset$ ,  $\varepsilon$ ,  $a$

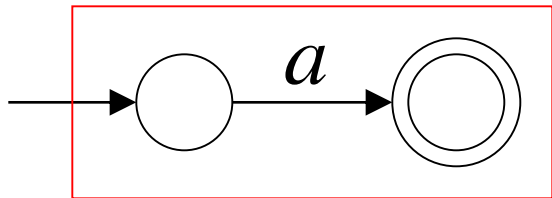
Corresponding  
NFAs



$$L(M_1) = \emptyset = L(\emptyset)$$



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



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

regular  
languages

# Inductive Hypothesis

Suppose

that for regular expressions  $r_1$  and  $r_2$ ,  
 $L(r_1)$  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

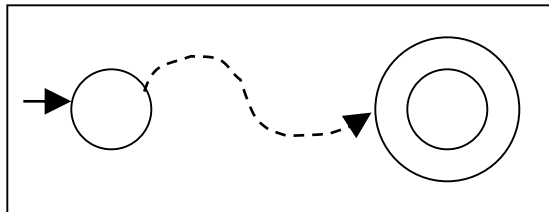
$L((r_1)) = L(r_1)$  is trivially a regular language  
(by induction hypothesis)



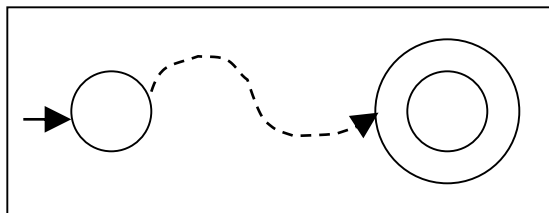
Using the regular closure of operations,  
we can construct recursively the NFA  $M$   
that accepts  $L(M) = L(r)$

Example:  $r = r_1 + r_2$

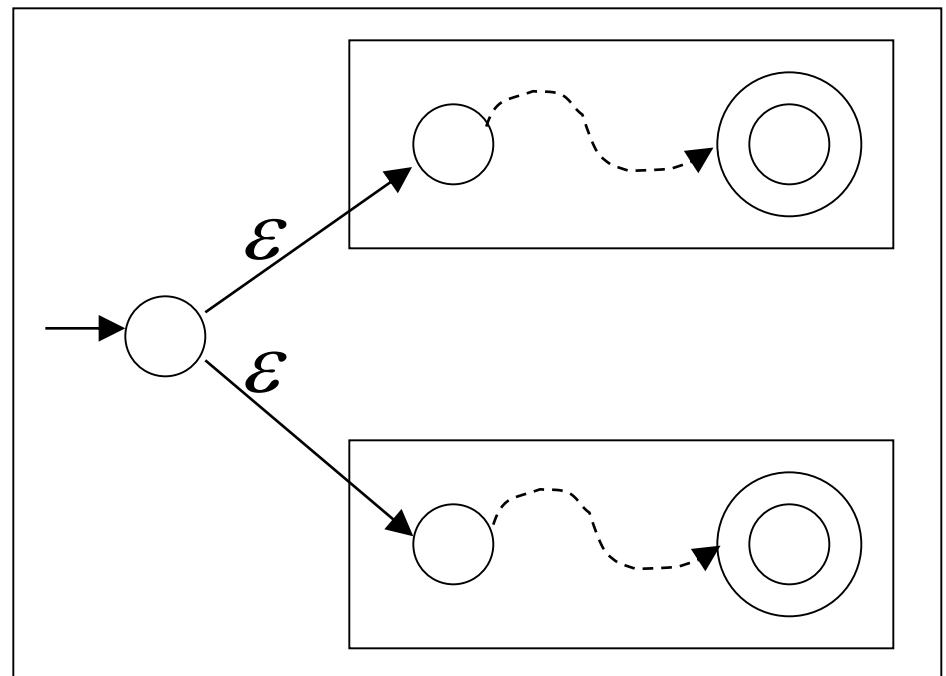
$$L(M_1) = L(r_1)$$



$$L(M_2) = L(r_2)$$



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



## Proof - Part 2

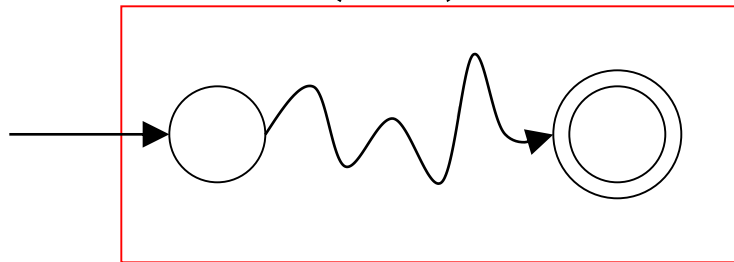
$$\left\{ \begin{array}{l} \text{Languages} \\ \text{Generated by} \\ \text{Regular Expressions} \end{array} \right\} \supseteq \left\{ \begin{array}{l} \text{Regular} \\ \text{Languages} \end{array} \right\}$$

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

We will convert an NFA that accepts  $L$   
to a regular expression

Since  $L$  is regular, there is a NFA  $M$  that accepts it

$$L(M) = L$$



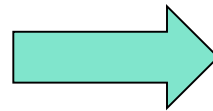
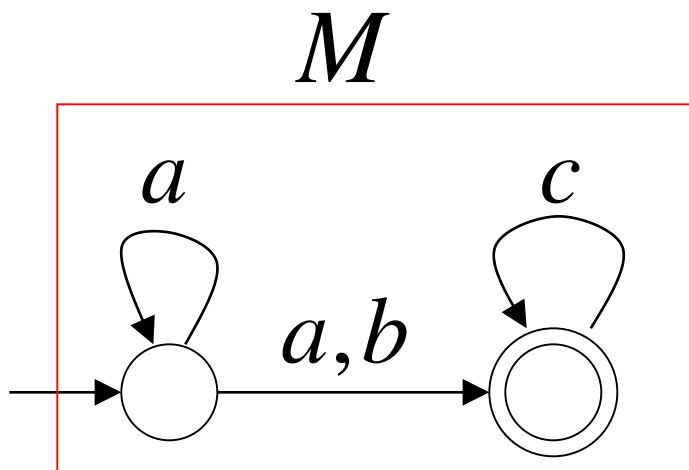
Take it with a single accept state

From  $M$  construct the equivalent

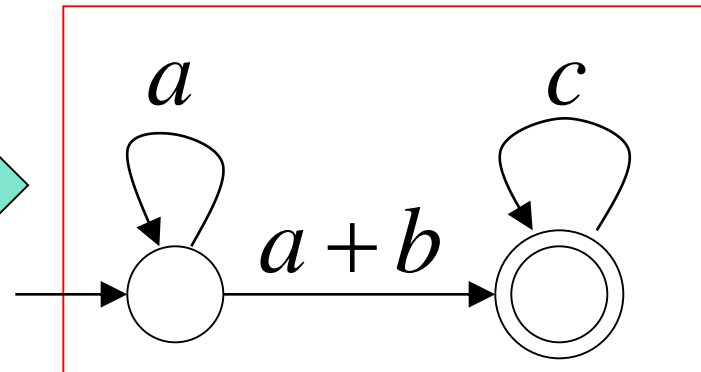
**Generalized Transition Graph**

in which transition labels are regular expressions

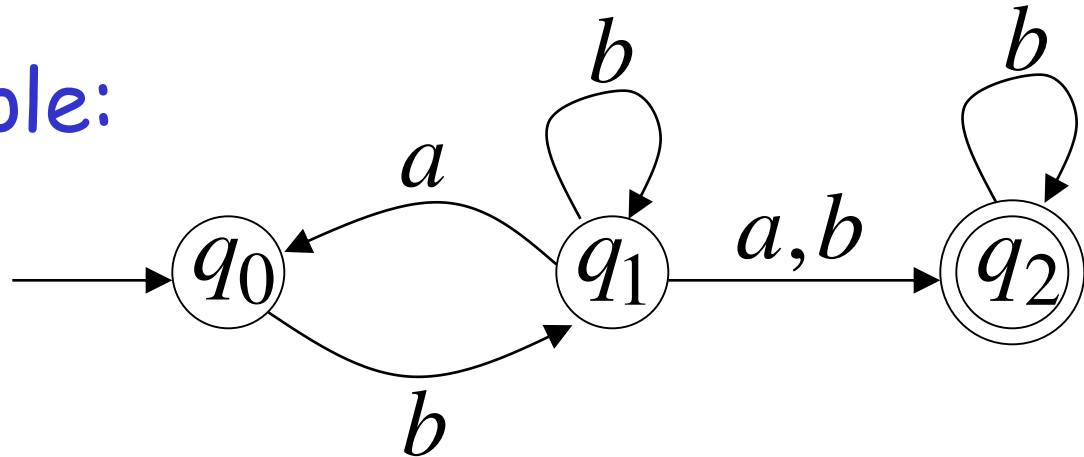
Example:



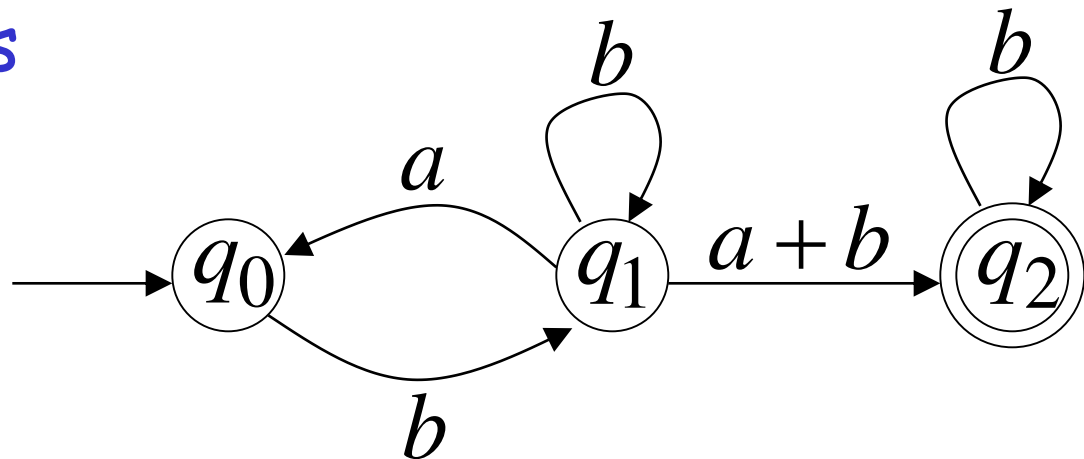
Corresponding  
Generalized transition graph



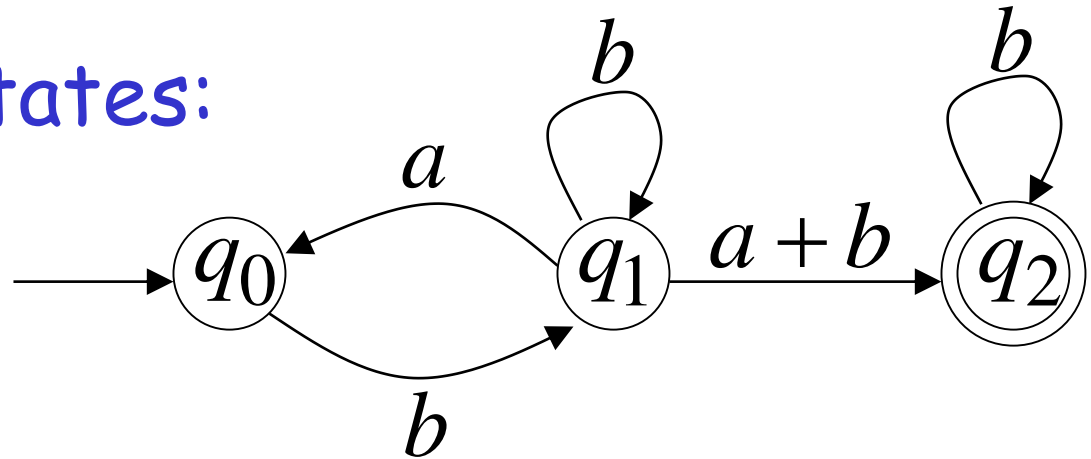
## Another Example:



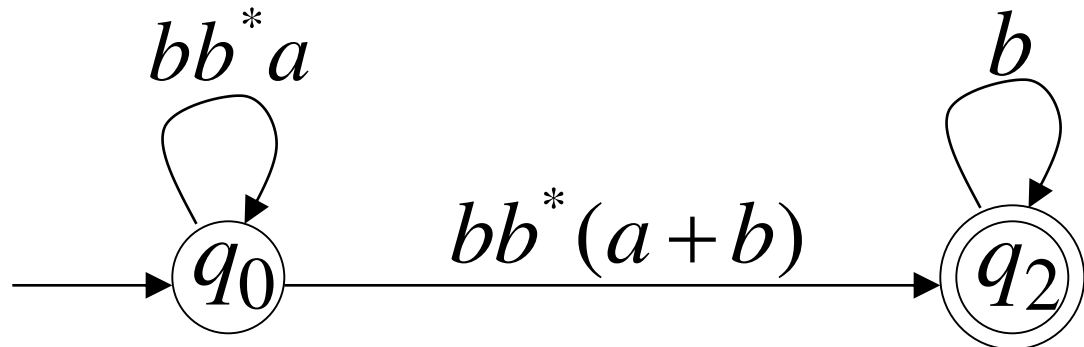
Transition labels  
are regular  
expressions



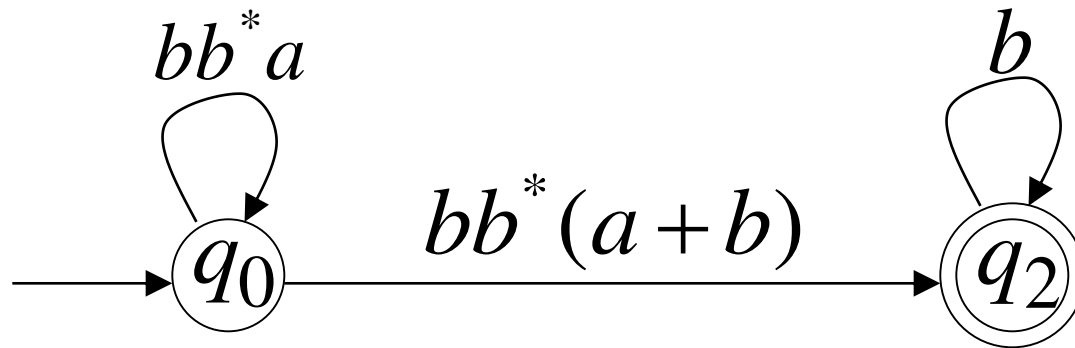
Reducing the states:



Transition labels  
are regular  
expressions



## Resulting Regular Expression:

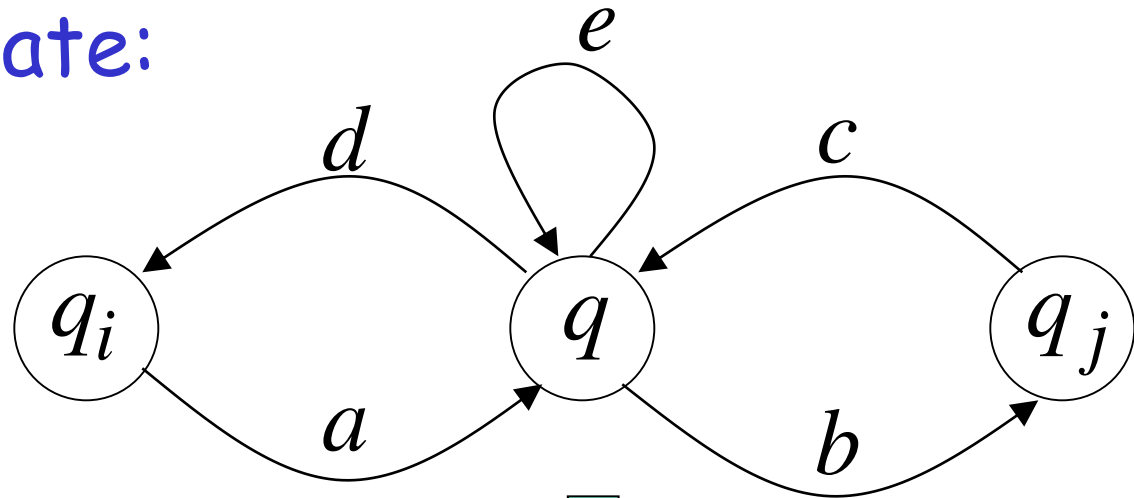


$$r = (bb^*a)^*bb^*(a+b)b^*$$

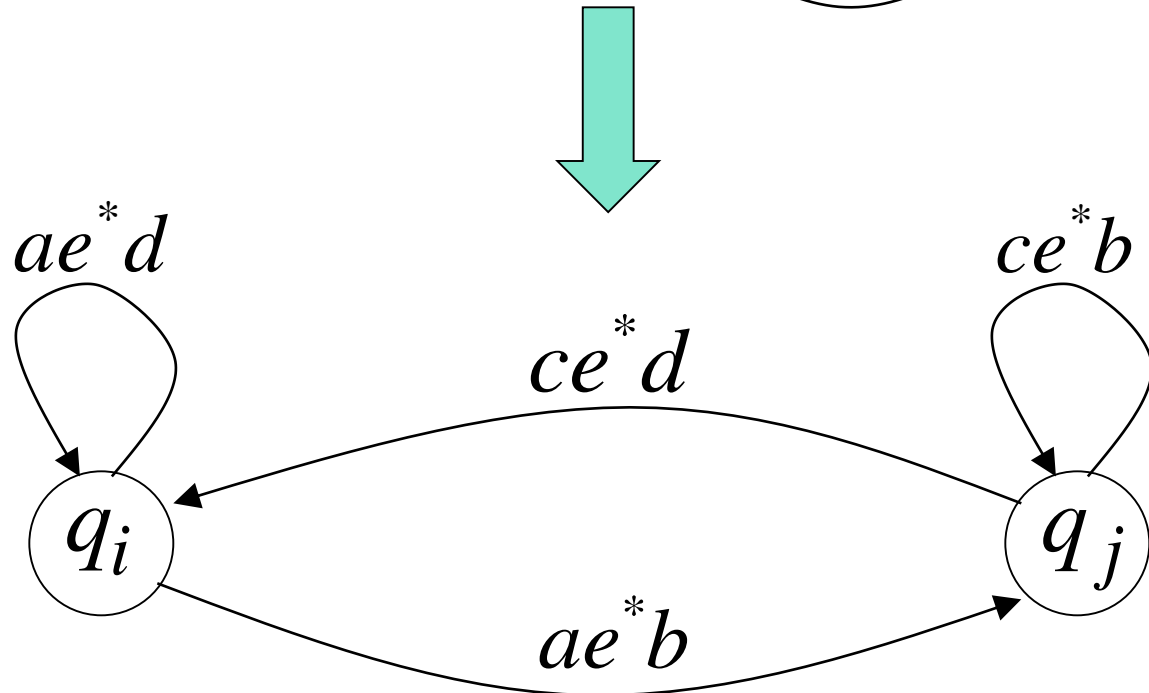
$$L(r) = L(M) = L$$

# In General

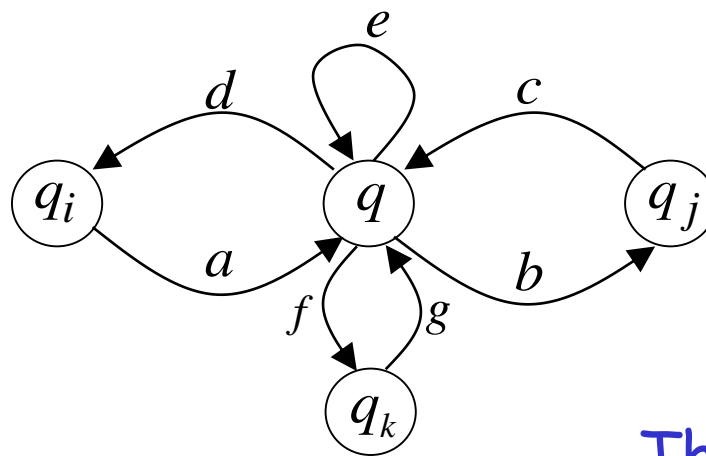
Removing a state:



2-neighbors

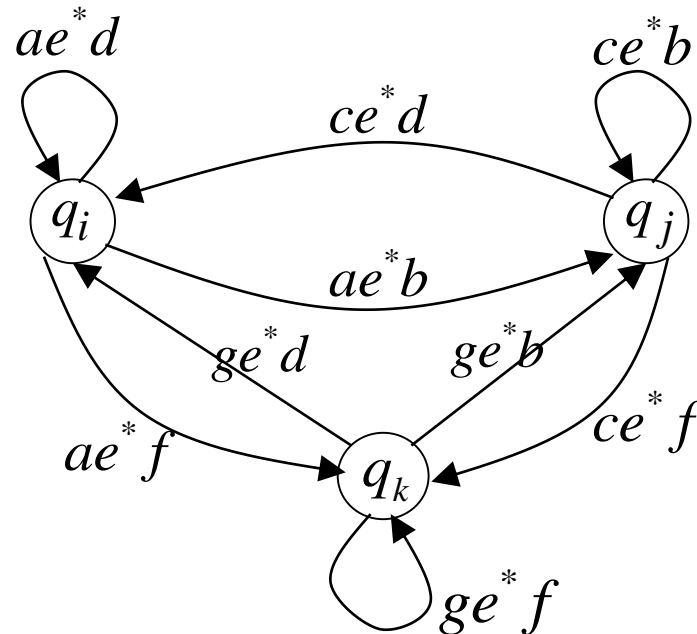
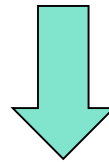




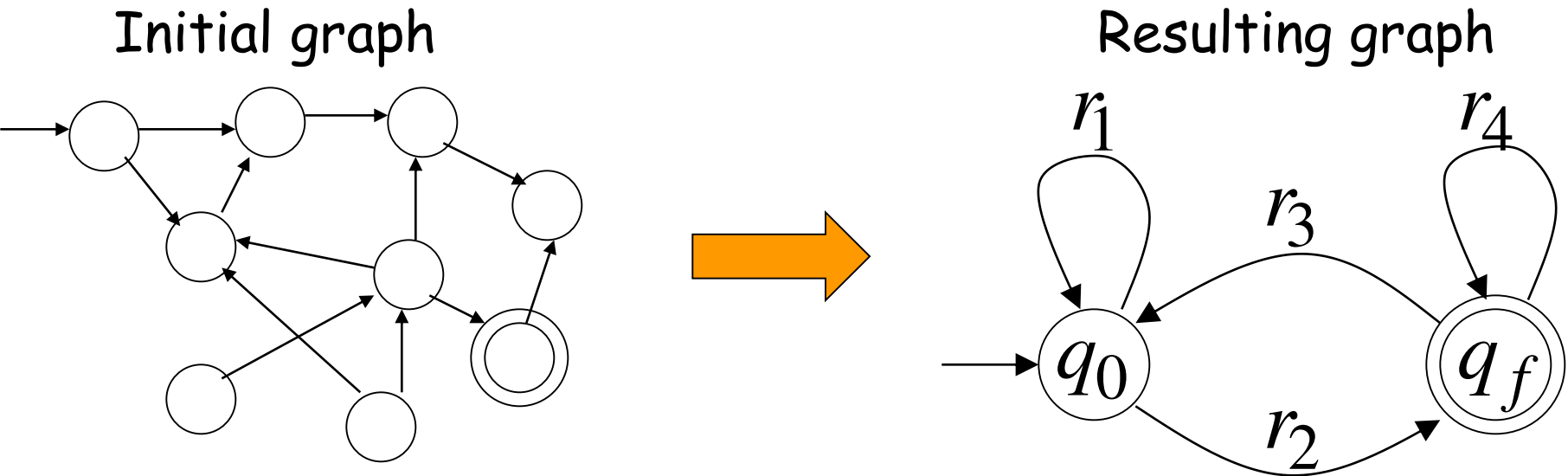


3-neighbors

This can be generalized  
to arbitrary number  
of neighbors to  $q$



By repeating the process until two states are left, the resulting graph is

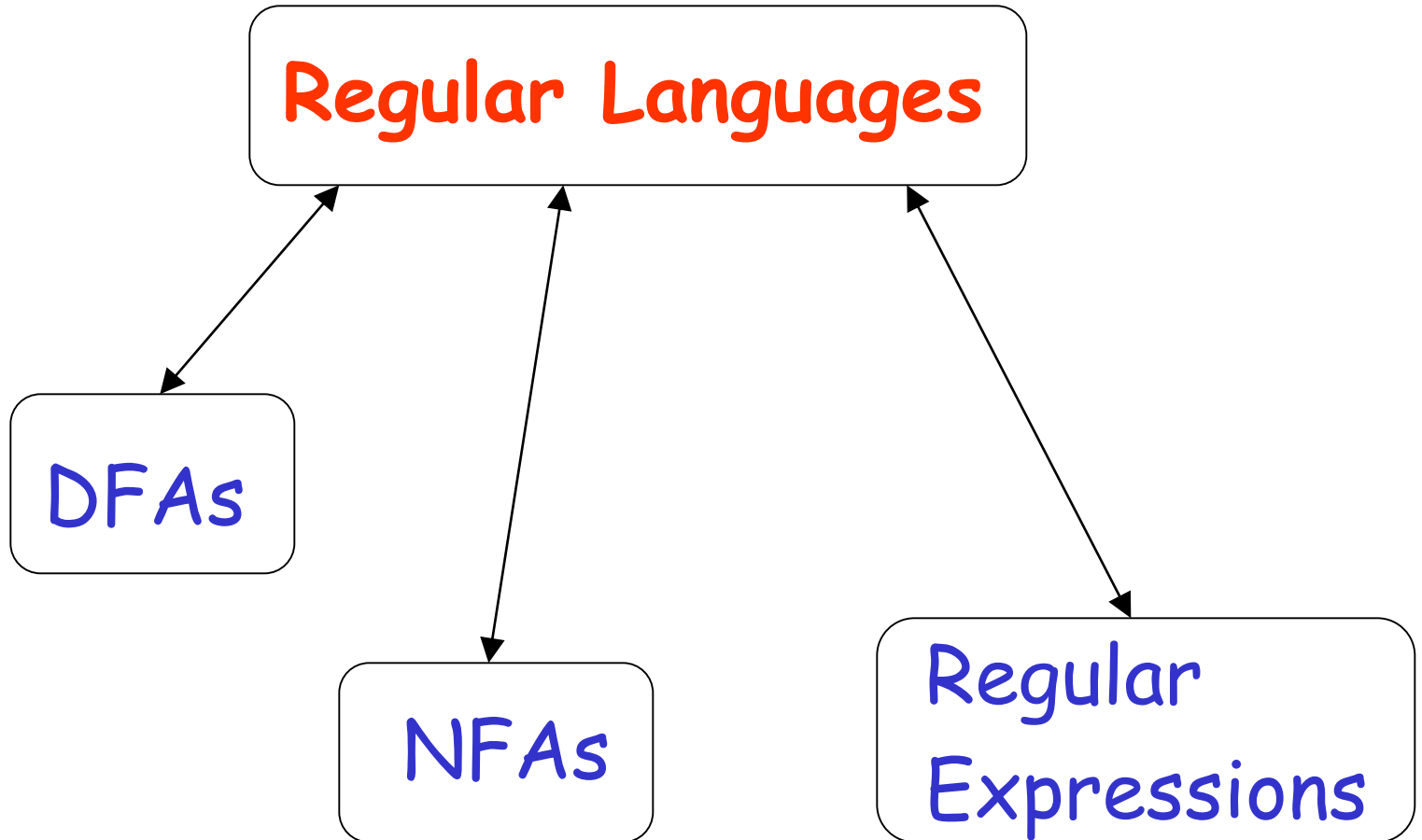


The resulting regular expression:

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

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

# Standard Representations of Regular Languages



When we say: We are given  
a Regular Language  $L$

We mean: Language  $L$  is in a standard  
representation

(DFA, NFA, or Regular Expression)