Lecture 2

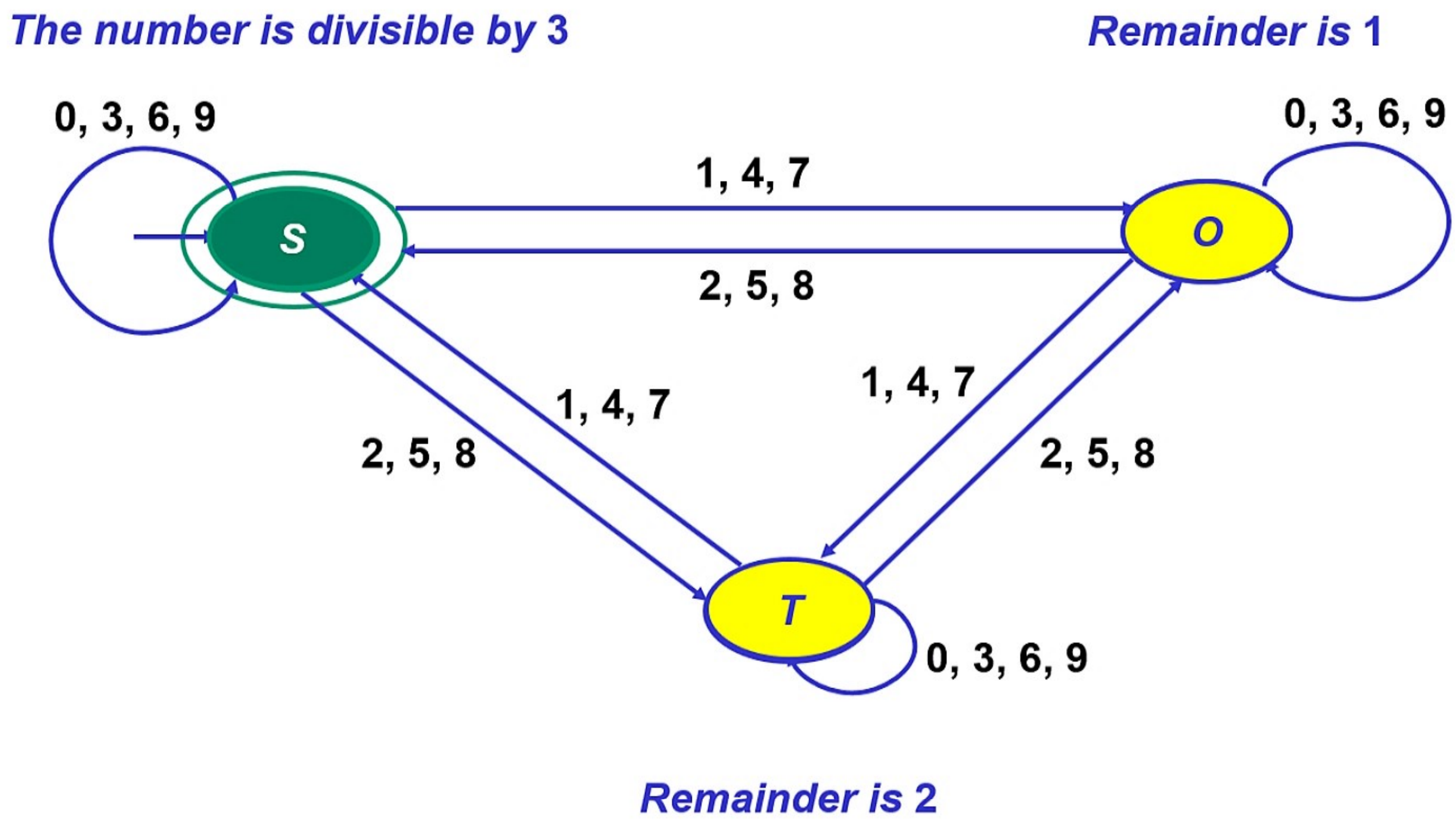
2 REGULAR LANGUAGES

2.1 FINITE AUTOMATA

2.1.1 Deterministic finite automaton (DFA)

2.1.2 Minimization of deterministic finite automaton

Deterministic Finite Automaton (DFA)



Lecture 2

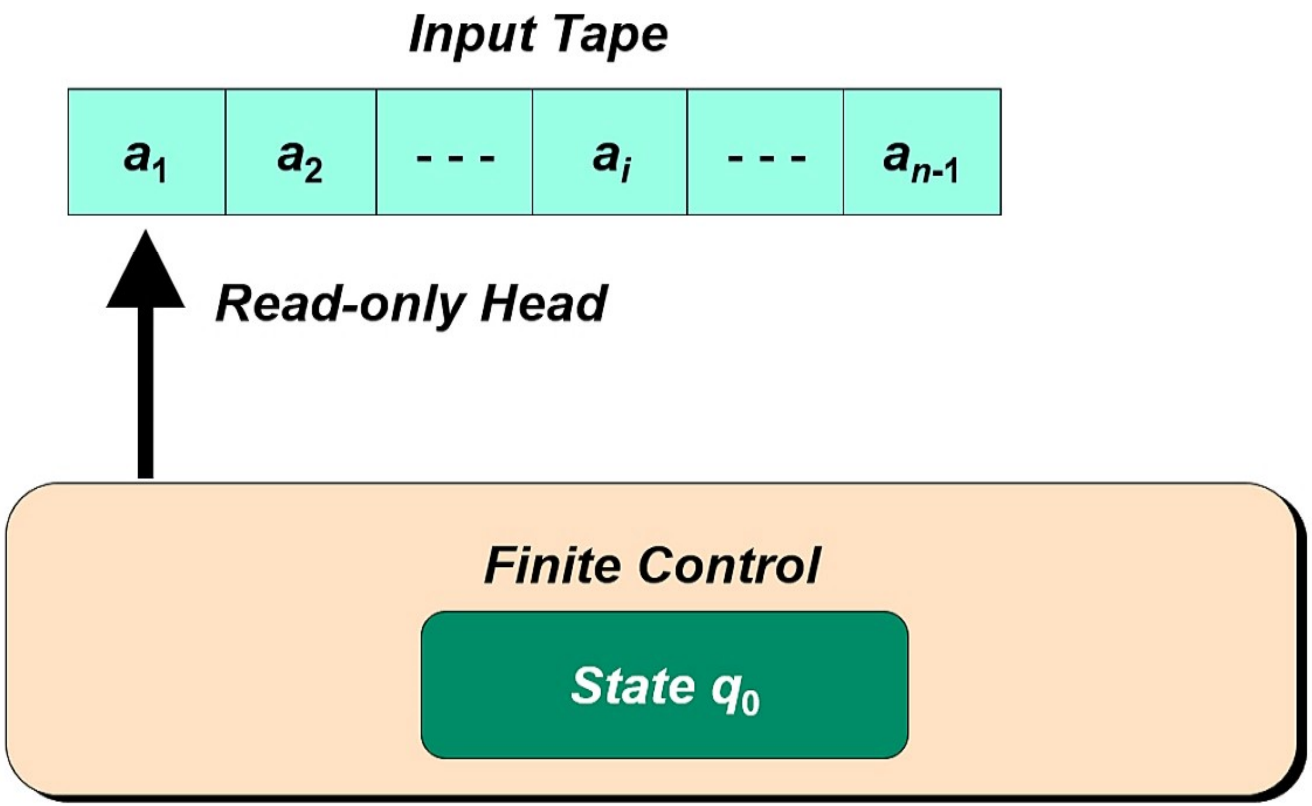
2 REGULAR LANGUAGES

2.1 FINITE AUTOMATA

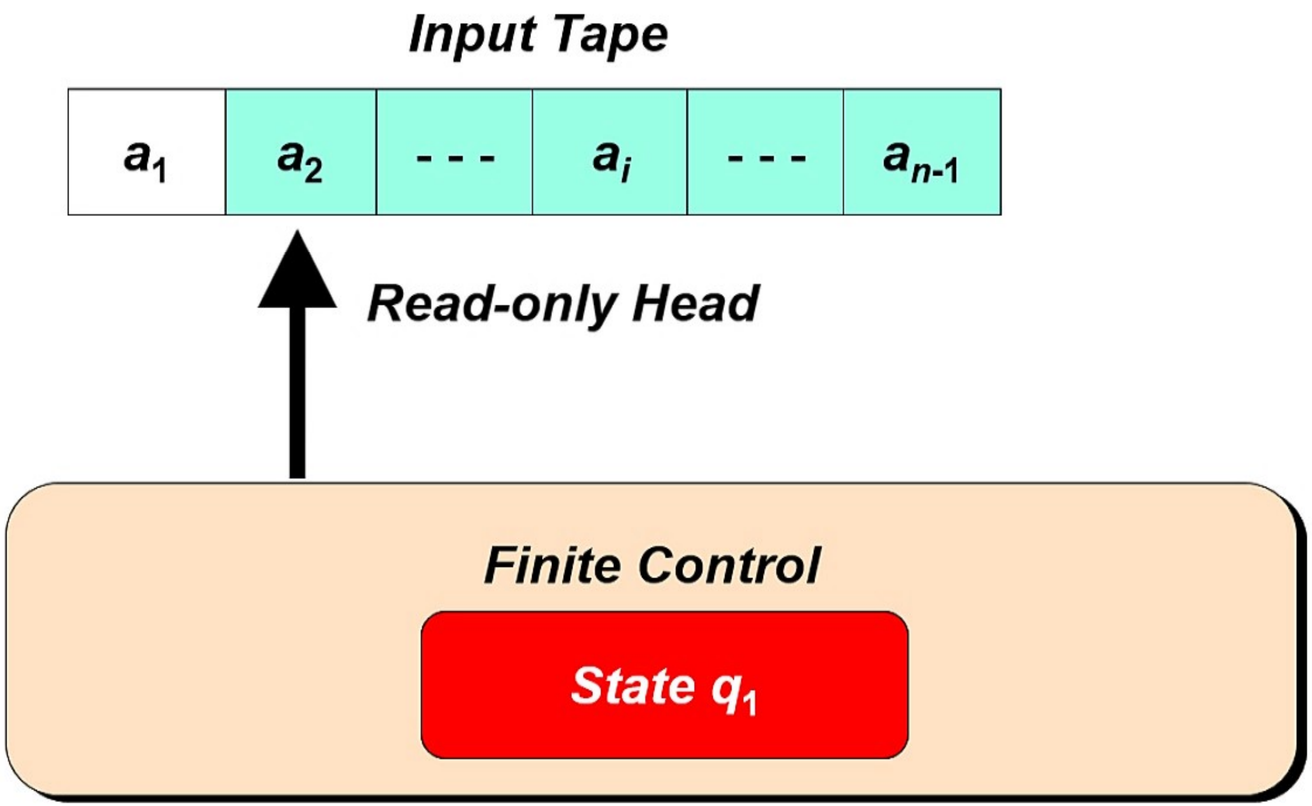
2.1.1 Deterministic finite automaton (DFA)

2.1.2 Minimization of deterministic finite automaton

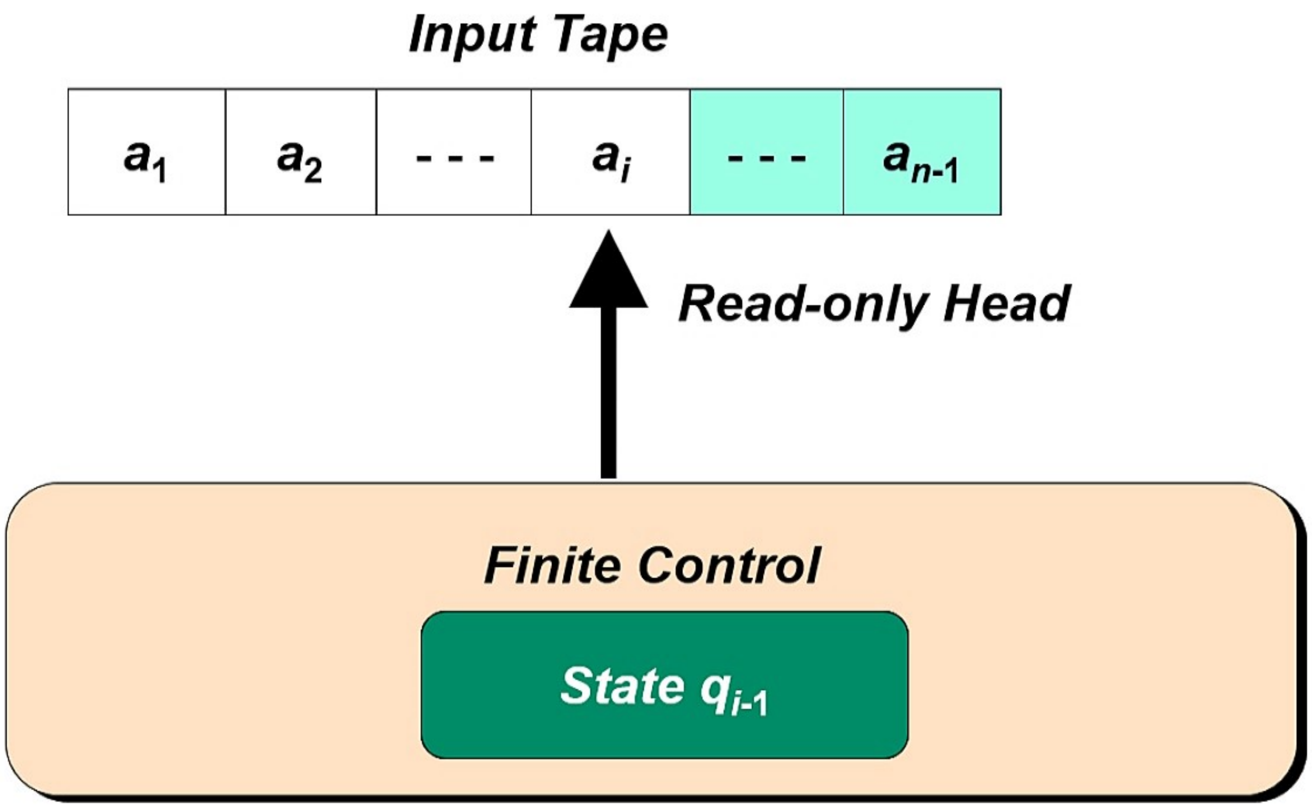
Deterministic Finite Automaton (DFA)



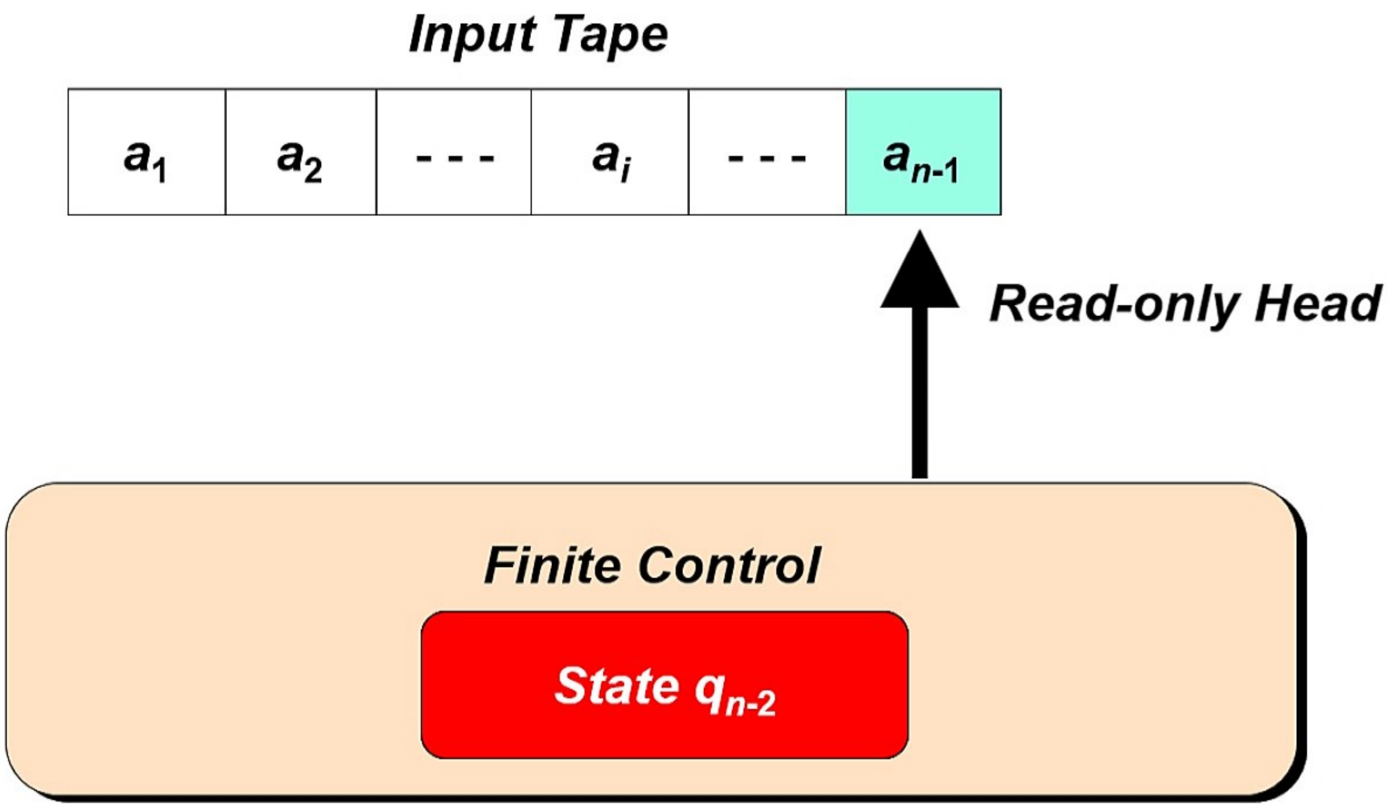
Deterministic Finite Automaton (DFA)



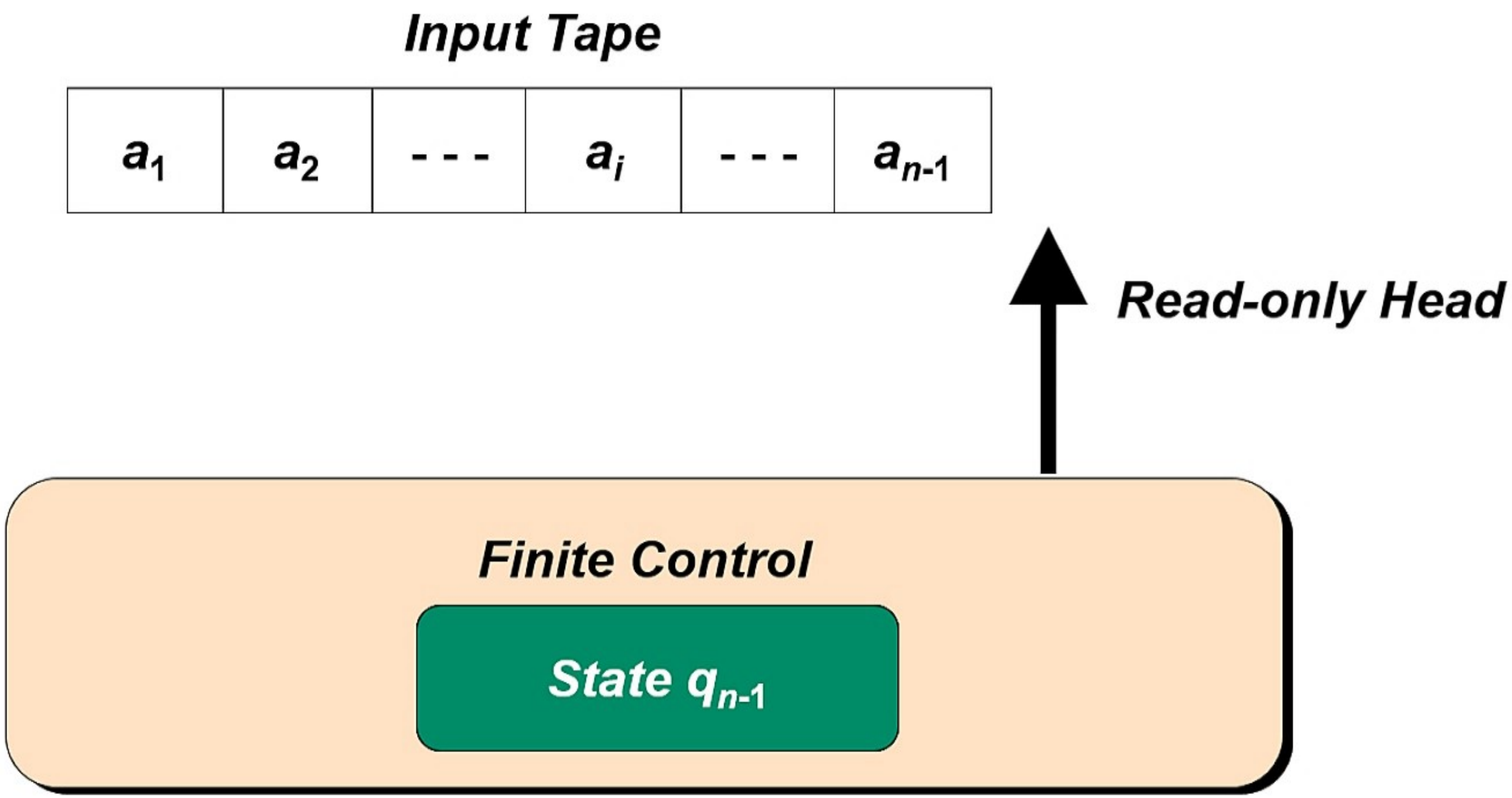
Deterministic Finite Automaton (DFA)



Deterministic Finite Automaton (DFA)



Deterministic Finite Automaton (DFA)



Deterministic Finite Automaton (DFA)

$$dfa = (Q, \Sigma, \delta, q_0, F)$$

Q	- a finite set of states
Σ	- a finite set of input symbols
δ	- a transition function $Q \times \Sigma \rightarrow Q$
$q_0 \in Q$	- an initial (start) state
$F \subseteq Q$	- a set of accepting states

$$NextState = \delta\,(CurrentState, InputSymbol)$$

Deterministic Finite Automaton (DFA)

$$\hat{\delta} : Q \times \Sigma^* \rightarrow Q$$

$$(1) \quad \hat{\delta}(q, \varepsilon) = q$$

$$(2) \quad \hat{\delta}(q, wa) = \delta(\hat{\delta}(q, w), a)$$

$$\hat{\delta}(q, a) = \delta(\hat{\delta}(q, \varepsilon), a) = \delta(q, a)$$

String and Language Acceptance (DFA)

$$DFA = (Q, \Sigma, \delta, q_0, F)$$

DFA **accepts the string** w **if** $\delta(q_0, w)=p$ **and** $p \in F$

DFA **accepts the language** $L(DFA)=\{w \mid \delta(q_0, w) \in F\} \subseteq \Sigma^*$

Example of a DFA

$\delta(ee, 1) = eo$ $\quad \delta(eo, 1) = ee$ $\quad \delta(oe, 1) = oo$ $\quad \delta(oo, 1) = oe$

$\delta(ee, 0) = oe$ $\quad \delta(eo, 0) = oo$ $\quad \delta(oe, 0) = ee$ $\quad \delta(oo, 0) = eo$

$$\delta(ee, \varepsilon) = ee$$

Example of a DFA

$\delta(ee, 1) = eo$ $\delta(eo, 1) = ee$ $\delta(oe, 1) = oo$ $\delta(oo, 1) = oe$

$\delta(ee, 0) = oe$ $\delta(eo, 0) = oo$ $\delta(oe, 0) = ee$ $\delta(oo, 0) = eo$

$$\delta(ee, \varepsilon) = ee$$

$$\delta(ee, 1) = \delta(\ \delta(ee, \varepsilon)\ , 1) =$$

Example of a DFA

$\delta(ee, 1) = eo$ $\delta(eo, 1) = ee$ $\delta(oe, 1) = oo$ $\delta(oo, 1) = oe$

$\delta(ee, 0) = oe$ $\delta(eo, 0) = oo$ $\delta(oe, 0) = ee$ $\delta(oo, 0) = eo$

$$\boxed{\delta(ee, \varepsilon) = ee}$$

$$\delta(ee, 1) = \delta(\ \delta(ee, \varepsilon)\ , 1) =$$

Example of a DFA

$\delta(ee, 1) = eo$ $\quad$ $\delta(eo, 1) = ee$ $\quad$ $\delta(oe, 1) = oo$ $\quad$ $\delta(oo, 1) = oe$

$\delta(ee, 0) = oe$ $\quad$ $\delta(eo, 0) = oo$ $\quad$ $\delta(oe, 0) = ee$ $\quad$ $\delta(oo, 0) = eo$

$$\delta(ee, \varepsilon) = ee$$

$$\delta(ee, 1) = \delta(\ \delta(ee, \varepsilon)\ , 1) = \delta(\ ee\ , 1) =$$

Example of a DFA

$\delta(ee, 1) = eo$ $\quad$ $\delta(eo, 1) = ee$ $\quad$ $\delta(oe, 1) = oo$ $\quad$ $\delta(oo, 1) = oe$

$\delta(ee, 0) = oe$ $\quad$ $\delta(eo, 0) = oo$ $\quad$ $\delta(oe, 0) = ee$ $\quad$ $\delta(oo, 0) = eo$

$$\delta(ee, \varepsilon) = ee$$

$$\delta(ee, 1) = \delta(\ \delta(ee, \varepsilon)\ , 1) = \delta(\ ee\ , 1) = eo$$

Example of a DFA

$\delta(ee, 1) = eo$ $\delta(eo, 1) = ee$ $\delta(oe, 1) = oo$ $\delta(oo, 1) = oe$

$\delta(ee, 0) = oe$ $\delta(eo, 0) = oo$ $\delta(oe, 0) = ee$ $\delta(oo, 0) = eo$

$$\delta(ee, \varepsilon) = ee$$

$$\delta(ee, 1) = \delta(\ \delta(ee, \varepsilon)\ , 1) = \delta(\ ee\ , 1) = eo$$

$$\delta(ee, 1101) =$$

$$\delta(ee, 1101)$$

Example of a DFA

$\delta(ee, 1) = eo$ $\quad \delta(eo, 1) = ee$ $\quad \delta(oe, 1) = oo$ $\quad \delta(oo, 1) = oe$

$\delta(ee, 0) = oe$ $\quad \delta(eo, 0) = oo$ $\quad \delta(oe, 0) = ee$ $\quad \delta(oo, 0) = eo$

$$\delta(ee, \varepsilon) = ee$$

$$\delta(ee, 1) = \delta(\ \delta(ee, \varepsilon)\ , 1) = \delta(\ ee\ , 1) = eo$$

$$\delta(ee, 1101) =$$

$$\delta(\ \ \delta(ee, 110)\ \ , 1)$$

Example of a DFA

$\delta(ee, 1) = eo$ $\delta(eo, 1) = ee$ $\delta(oe, 1) = oo$ $\delta(oo, 1) = oe$

$\delta(ee, 0) = oe$ $\delta(eo, 0) = oo$ $\delta(oe, 0) = ee$ $\delta(oo, 0) = eo$

$$\delta(ee, \varepsilon) = ee$$

$$\delta(ee, 1) = \delta(\ \delta(ee, \varepsilon)\ , 1) = \delta(\ ee\ , 1) = eo$$

$$\delta(ee, 1101) =$$

$$\delta(\delta(\delta(ee, 11), 0), 1)$$

Example of a DFA

$\delta(ee, 1) = eo$ $\quad$ $\delta(eo, 1) = ee$ $\quad$ $\delta(oe, 1) = oo$ $\quad$ $\delta(oo, 1) = oe$

$\delta(ee, 0) = oe$ $\quad$ $\delta(eo, 0) = oo$ $\quad$ $\delta(oe, 0) = ee$ $\quad$ $\delta(oo, 0) = eo$

$$\delta(ee, \varepsilon) = ee$$

$$\delta(ee, 1) = \delta(\ \delta(ee, \varepsilon)\ , 1) = \delta(\ ee\ , 1) = eo$$

$$\delta(ee, 1101) =$$

$$\delta(\delta(\delta(\delta(ee, 1), 1), 0), 1)$$

Example of a DFA

$\delta(ee, 1) = eo$ $\quad$ $\delta(eo, 1) = ee$ $\quad$ $\delta(oe, 1) = oo$ $\quad$ $\delta(oo, 1) = oe$

$\delta(ee, 0) = oe$ $\quad$ $\delta(eo, 0) = oo$ $\quad$ $\delta(oe, 0) = ee$ $\quad$ $\delta(oo, 0) = eo$

$$\delta(ee, \varepsilon) = ee$$

$$\delta(ee, 1) = \delta(\ \delta(ee, \varepsilon)\ , 1) = \delta(\ ee\ , 1) = eo$$

$$\delta(ee, 1101) =$$

$$\delta(\delta(\delta(eo, 1), 0), 1)$$

Example of a DFA

$\delta(ee, 1) = eo$ $\delta(eo, 1) = ee$ $\delta(oe, 1) = oo$ $\delta(oo, 1) = oe$

$\delta(ee, 0) = oe$ $\delta(eo, 0) = oo$ $\delta(oe, 0) = ee$ $\delta(oo, 0) = eo$

$$\delta(ee, \varepsilon) = ee$$

$$\delta(ee, 1) = \delta(\ \delta(ee, \varepsilon)\ , 1) = \delta(\ ee\ , 1) = eo$$

$$\delta(ee, 1101) =$$

$$\delta(\delta(ee, 0), 1)$$

Example of a DFA

$\delta(ee, 1) = eo$ $\delta(eo, 1) = ee$ $\delta(oe, 1) = oo$ $\delta(oo, 1) = oe$

$\delta(ee, 0) = oe$ $\delta(eo, 0) = oo$ $\delta(oe, 0) = ee$ $\delta(oo, 0) = eo$

$$\delta(ee, \varepsilon) = ee$$

$$\delta(ee, 1) = \delta(\ \delta(ee, \varepsilon)\ , 1) = \delta(\ ee\ , 1) = eo$$

$$\delta(ee, 1101) =$$

$$\delta(oe, 1)$$

Example of a DFA

$\delta(ee, 1) = eo$ $\delta(eo, 1) = ee$ $\delta(oe, 1) = oo$ $\delta(oo, 1) = oe$

$\delta(ee, 0) = oe$ $\delta(eo, 0) = oo$ $\delta(oe, 0) = ee$ $\delta(oo, 0) = eo$

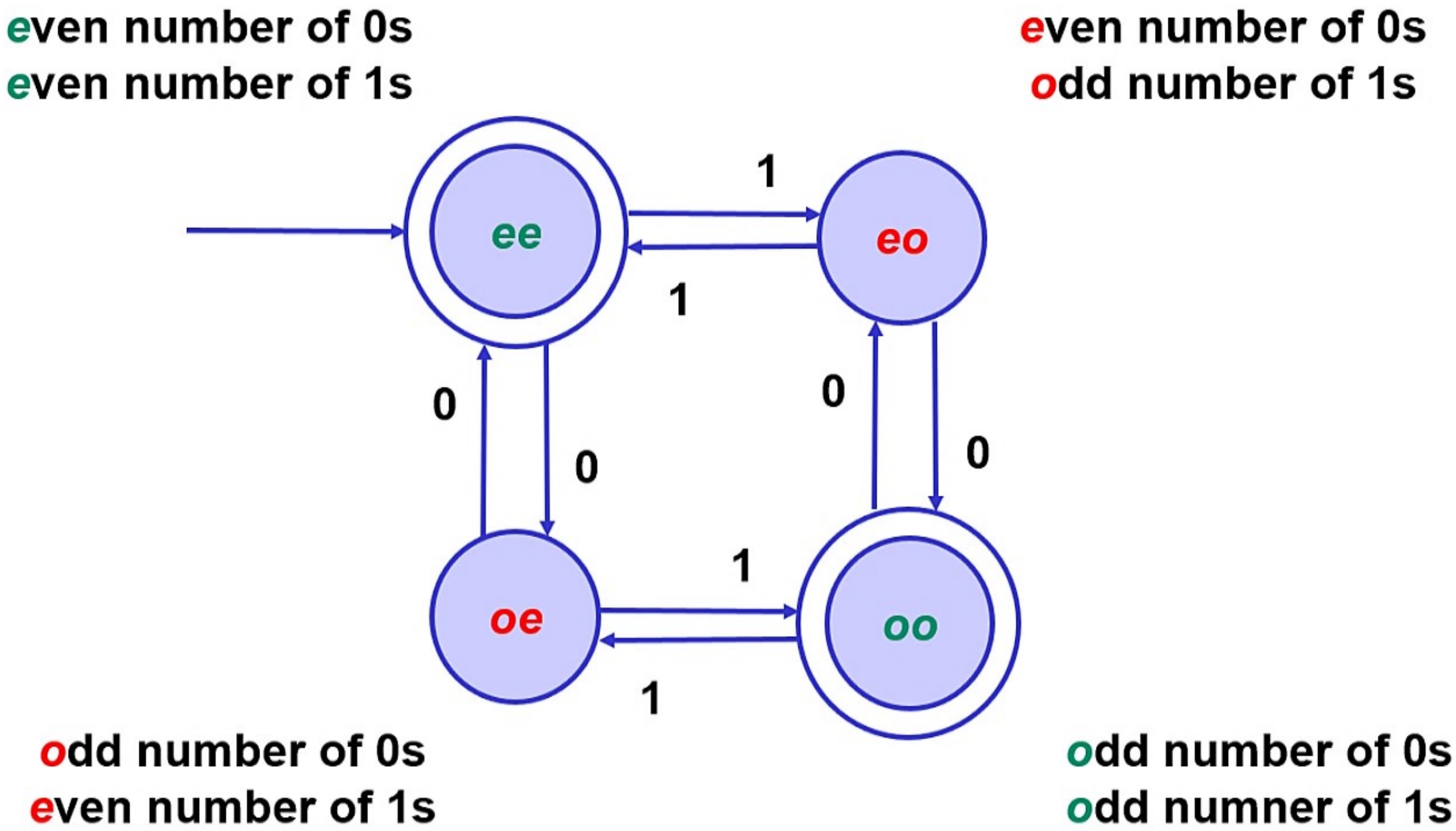
$$\delta(ee, \varepsilon) = ee$$

$$\delta(ee, 1) = \delta(\delta(ee, \varepsilon) , 1) = \delta(ee , 1) = eo$$

$$\delta(ee, 1101) =$$

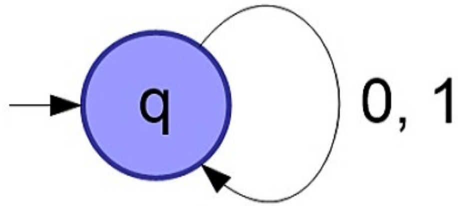
$$oo$$

Example of a DFA



Example of a DFA

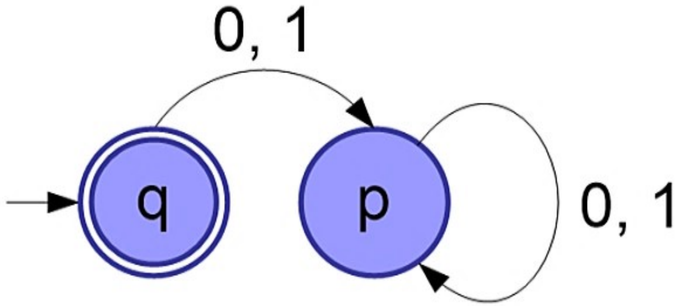
$M_1 = (\{q\}, \{0, 1\}, \{\delta(q, 0) = q, \delta(q, 1) = q\}, q, \{\ \})$



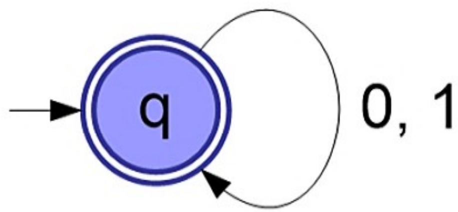
$L(M_1) = \{\ \}$

$M_2 = (\{q, p\}, \{0, 1\}, \{\delta(q, 0) = p, \delta(q, 1) = p, \delta(p, 0) = p, \delta(p, 1) = p\}, q, \{ q \})$

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

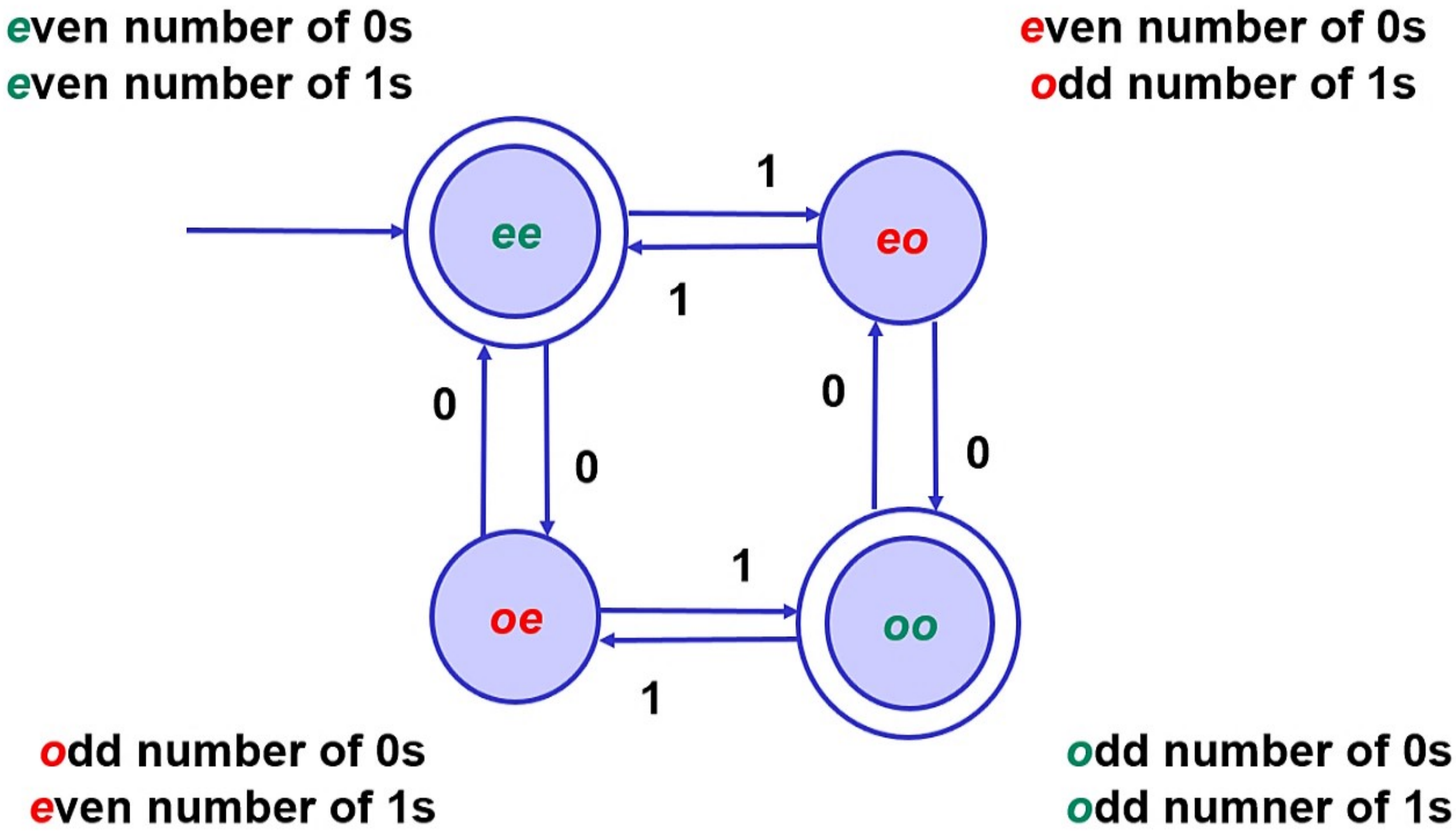


$M_3 = (\{q\}, \{0, 1\}, \{\delta(q, 0) = q, \delta(q, 1) = q\}, q, \{ q \})$



$L(M_3) = \Sigma^*$

Software Implementation of DFA



State Representation Method – EXPLICIT

$\delta(ee, 0) = oe$
$\delta(ee, 1) = eo$
$\delta(eo, 0) = oo$
$\delta(eo, 1) = ee$
$\delta(oe, 0) = ee$
$\delta(oe, 1) = oo$
$\delta(oo, 0) = eo$
$\delta(oo, 1) = oe$

```
Table [EE, 0] = OE;
Table [EE, 1] = EO;
Table [EE, ⊥] = 1;
Table [OE, 0] = EE;
Table [OE, 1] = OO;
Table [OE, ⊥] = 0;
Table [EO, 0] = OO;
Table [EO, 1] = EE;
Table [EO, ⊥] = 0;
Table [OO, 0] = EO;
Table [OO, 1] = OE;
Table [OO, ⊥] = 1;

State = EE;
Read(Symbol);

while (Symbol != ⊥)
{
    State = Table [State, Symbol];
    Read(Symbol);
}

Print(Table [State, ⊥], State);
```

State Representation Method – IMPLICIT

$\delta(ee, 1) = eo$ $\delta(eo, 1) = ee$ $\delta(oe, 1) = oo$ $\delta(oo, 1) = oe$

$\delta(ee, 0) = oe$ $\delta(eo, 0) = oo$ $\delta(oe, 0) = ee$ $\delta(oo, 0) = eo$

```
EE: Read(Symbol);

    if (Symbol == ⊥)
        Print("STRING IS ACCEPTABLE, EVEN:0, EVEN:1");
    if (Symbol == 0)
        goto OE;
    else
        goto EO;

OE: Read(Symbol);

    if (Symbol == ⊥)
        Print("STRING IS NOT ACCEPTABLE, ODD:0, EVEN:1");
    if (Symbol == 0)
        goto EE;
    else
        goto OO;
```

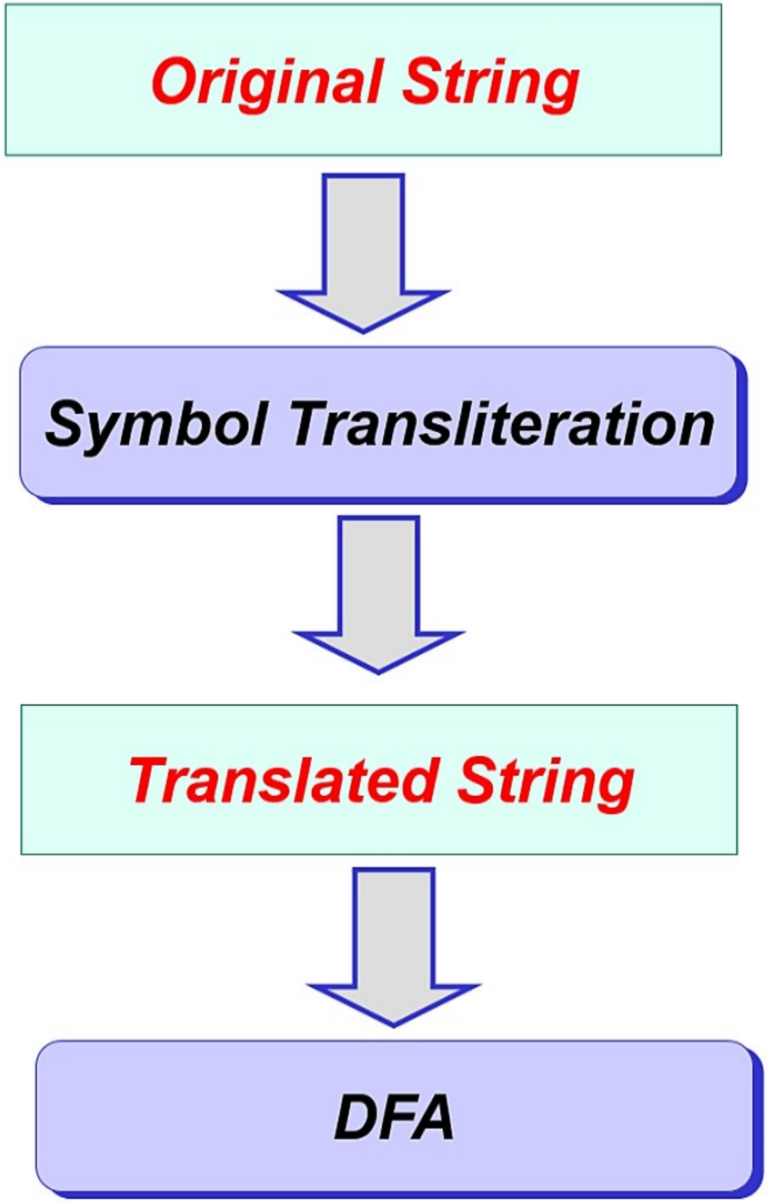
State Representation Method – IMPLICIT

$\delta(ee, 1) = eo$ $\delta(eo, 1) = ee$ $\delta(oe, 1) = oo$ $\delta(oo, 1) = oe$

$\delta(ee, 0) = oe$ $\delta(eo, 0) = oo$ $\delta(oe, 0) = ee$ $\delta(oo, 0) = eo$

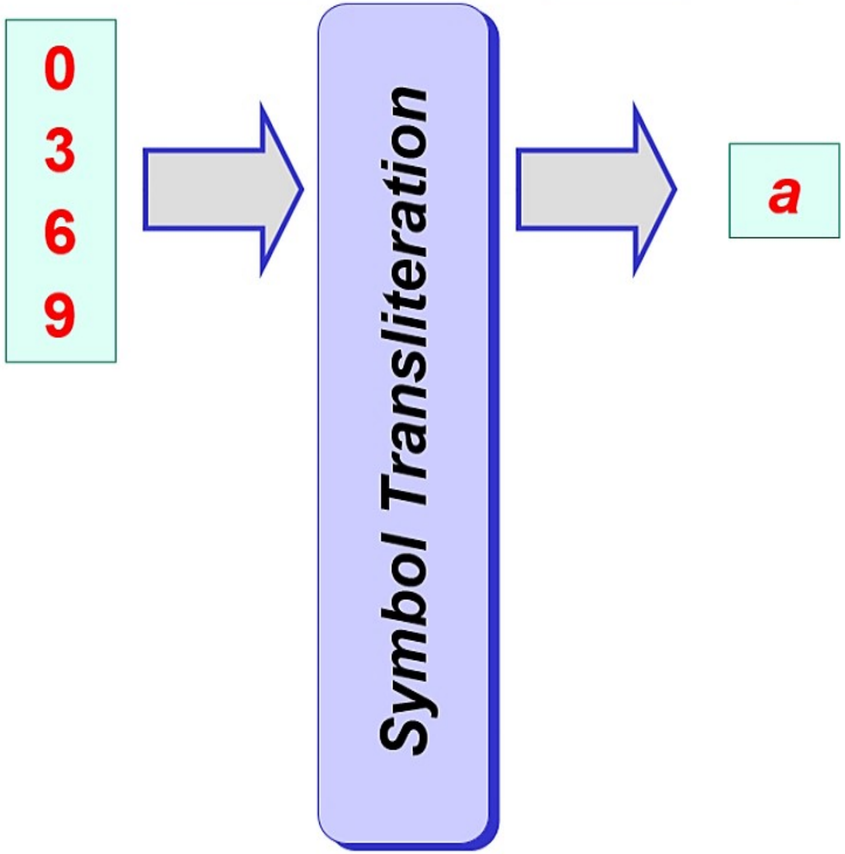
```
EO: Read(Symbol);

    if (Symbol == ⊥)
        Print("STRING IS NOT ACCEPTABLE, EVEN:0, ODD:1");
    if (Symbol == 0)
        goto OO;
    else
        goto EE;

OO: Read(Symbol);

    if (Symbol == ⊥)
        Print("STRING IS ACCEPTABLE, ODD:0, ODD:1");
    if (Symbol == 0)
        goto EO;
    else
        goto OE;
```

Symbol Transliteration



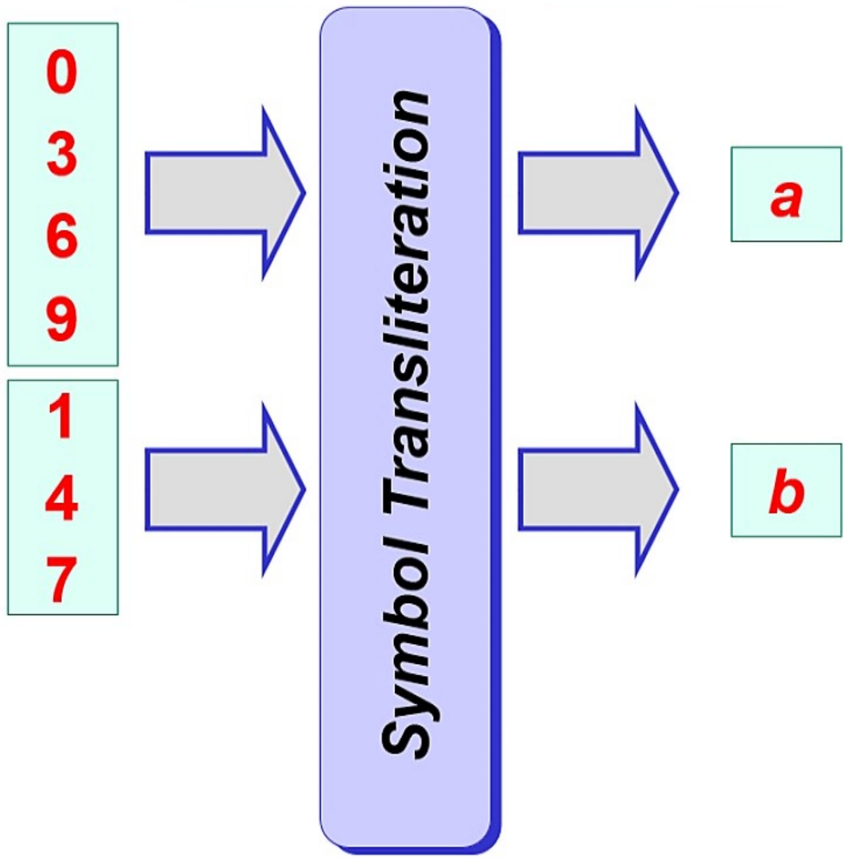
Example of Symbol Transliteration

	0	1	2	3	4	5	6	7	8	9	
S	*S*	*O*	*T*	*S*	*O*	*T*	*S*	*O*	*T*	*S*	1
O	*O*	*T*	*S*	*O*	*T*	*S*	*O*	*T*	*S*	*O*	0
T	*T*	*S*	*O*	*T*	*S*	*O*	*T*	*S*	*O*	*T*	0



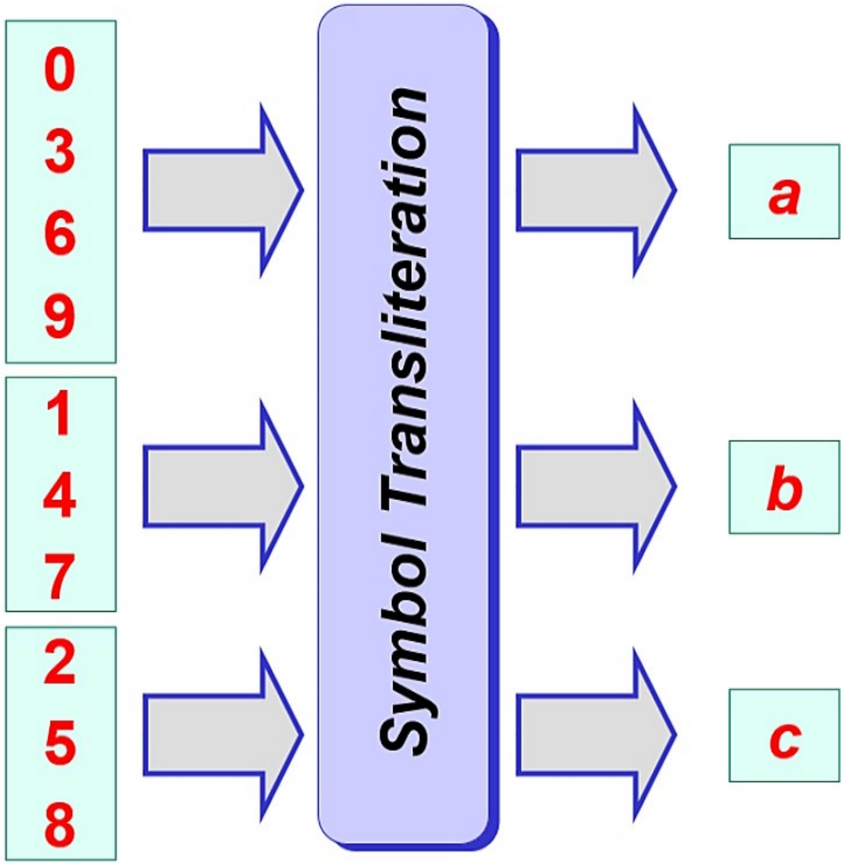
Example of Symbol Transliteration

	a	1	2	*a*	4	5	*a*	7	8	*a*	
S	*S*	*O*	*T*	*S*	*O*	*T*	*S*	*O*	*T*	*S*	1
O	*O*	*T*	*S*	*O*	*T*	*S*	*O*	*T*	*S*	*O*	0
T	*T*	*S*	*O*	*T*	*S*	*O*	*T*	*S*	*O*	*T*	0



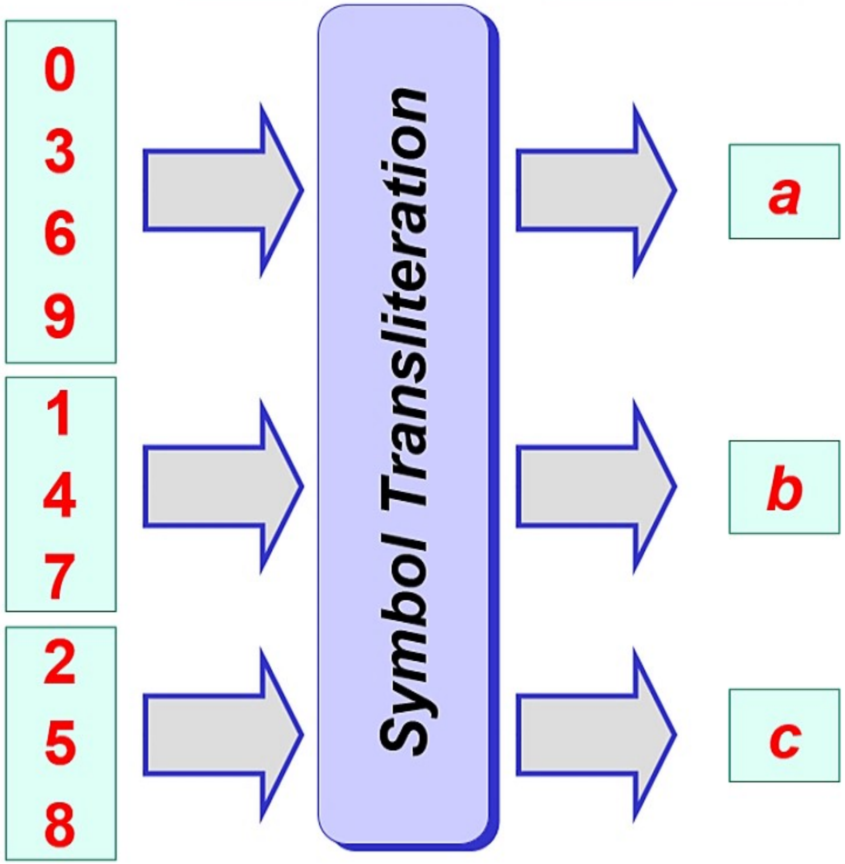
Example of Symbol Transliteration

	a	b	2	a	b	5	a	b	8	a	
S	*S*	*O*	*T*	*S*	*O*	*T*	*S*	*O*	*T*	*S*	1
O	*O*	*T*	*S*	*O*	*T*	*S*	*O*	*T*	*S*	*O*	0
T	*T*	*S*	*O*	*T*	*S*	*O*	*T*	*S*	*O*	*T*	0



Example of Symbol Transliteration

	a	b	c	a	b	c	a	b	c	a	
S	*S*	*O*	*T*	*S*	*O*	*T*	*S*	*O*	*T*	*S*	1
O	*O*	*T*	*S*	*O*	*T*	*S*	*O*	*T*	*S*	*O*	0
T	*T*	*S*	*O*	*T*	*S*	*O*	*T*	*S*	*O*	*T*	0



	a	b	c	
S	*S*	*O*	*T*	1
O	*O*	*T*	*S*	0
T	*T*	*S*	*O*	0

Lecture 2

2 REGULAR LANGUAGES

2.1 FINITE AUTOMATA

2.1.1 Deterministic finite automaton (DFA)

2.1.2 Minimization of deterministic finite automaton

Minimization of DFA

- **Efficient software implementation**

  - **Requires DFA with a minimum possible number of states**

    For a regular language ***L*** there exists a ***DFA M*** with less of equal number of states than any other ***DFA M'*** that accepts that same language ***L***.

- **Equivalence of states**

  State ***p*** of a ***DFA*** $M=(Q, \Sigma, \delta, q_0, F)$ is equivalent to the state ***p'*** of a ***DFA*** $M'=(Q', \Sigma, \delta', q_0', F')$ if and only if ***DFA M*** from the state ***p*** accepts the same set of strings as ***DFA M'*** accepts from the state ***p'***.

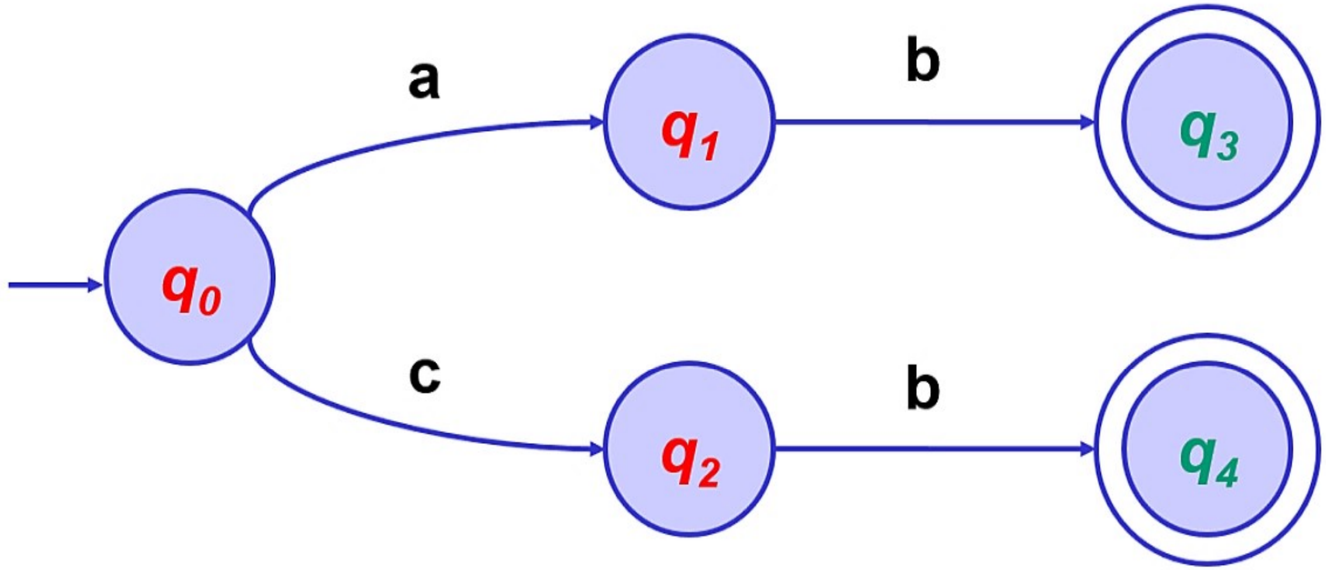
  For any string ***w*** in set Σ^* holds the following:

$$\delta(p, w) \in F \quad \wedge \quad \delta'(p', w) \in F'$$

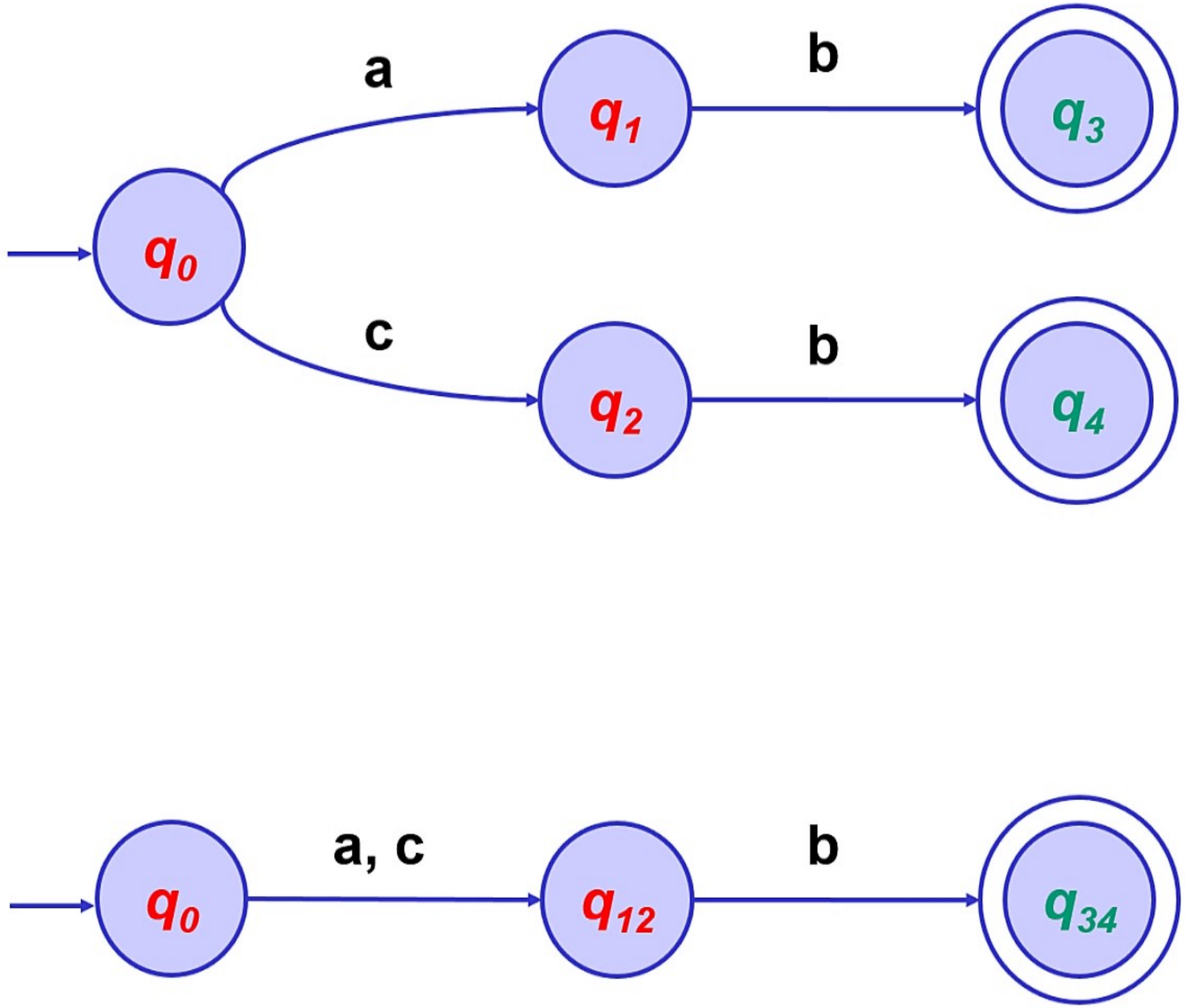
  or

$$\delta(p, w) \notin F \quad \wedge \quad \delta'(p', w) \notin F'$$

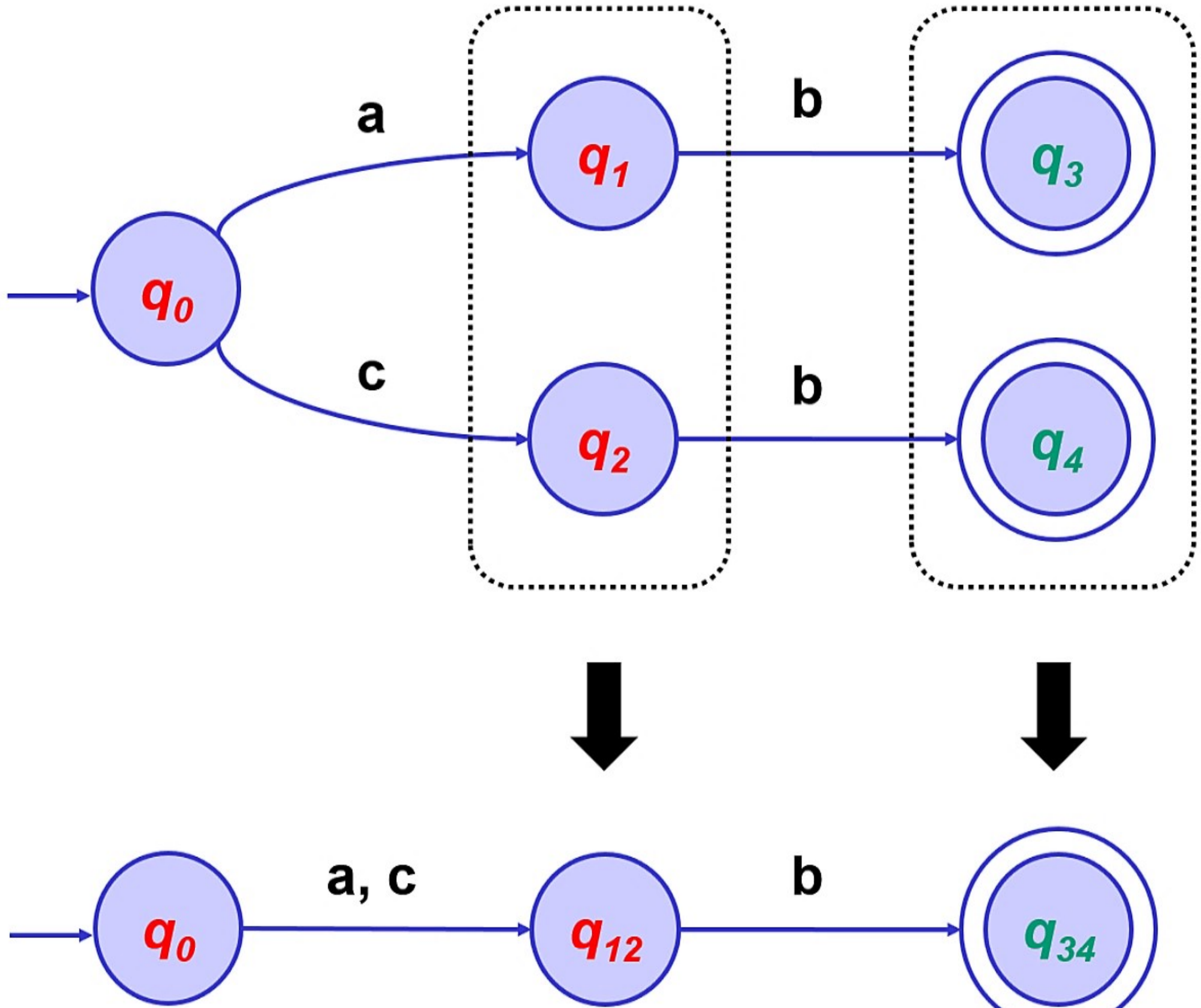
Minimization of DFA



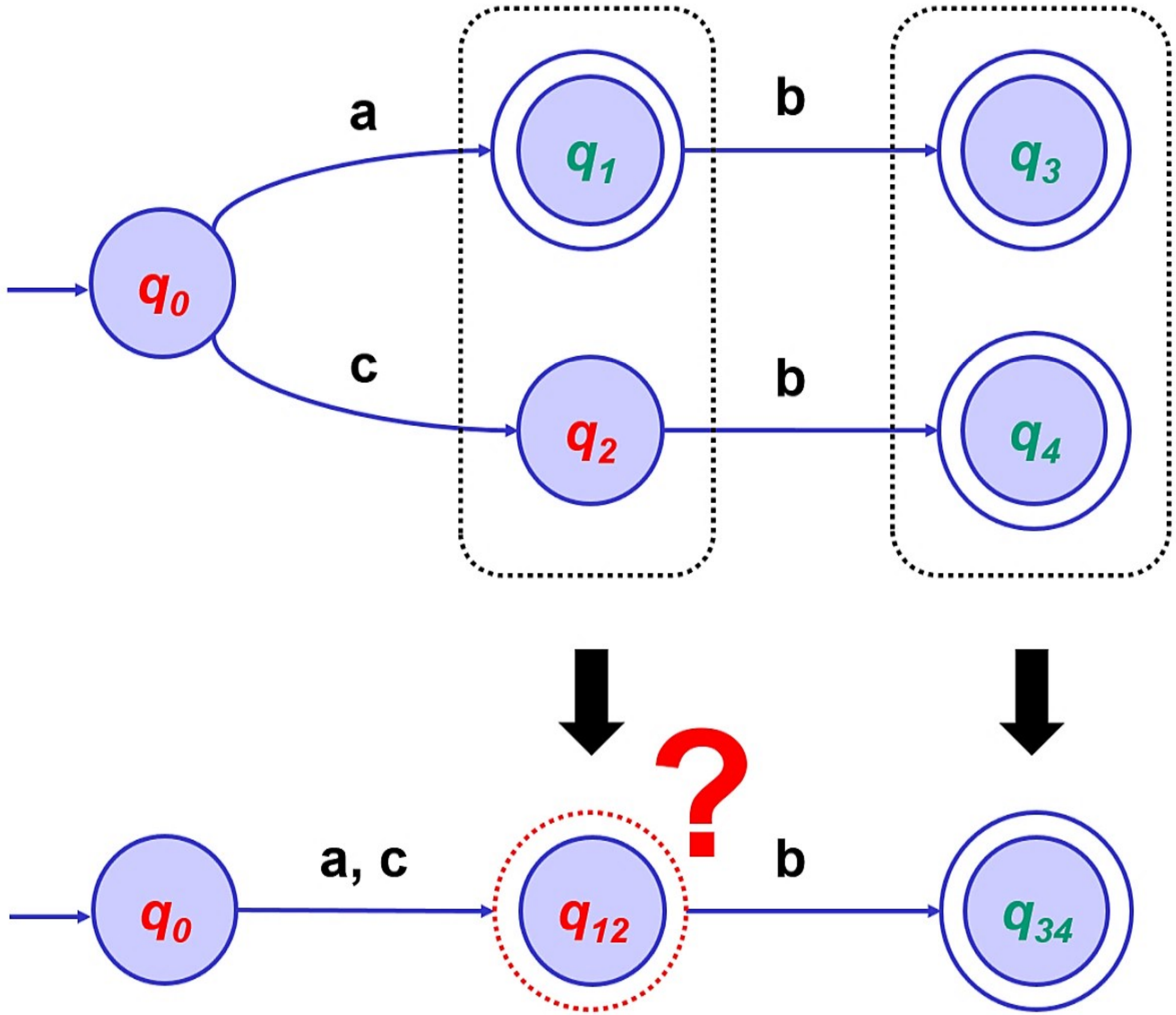
Minimization of DFA



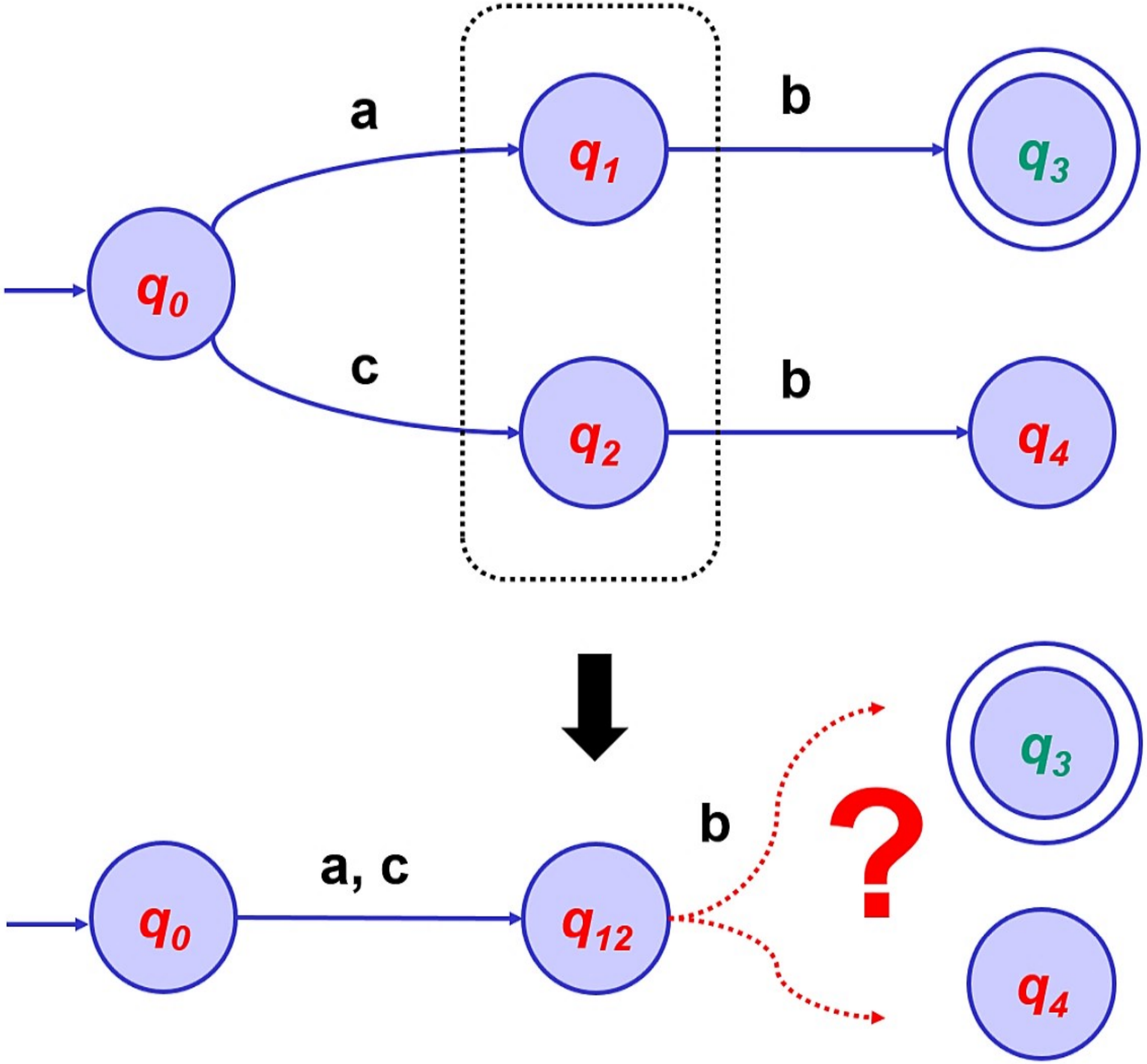
q_1 and q_2 are equivalent
q_3 and q_4 are equivalent



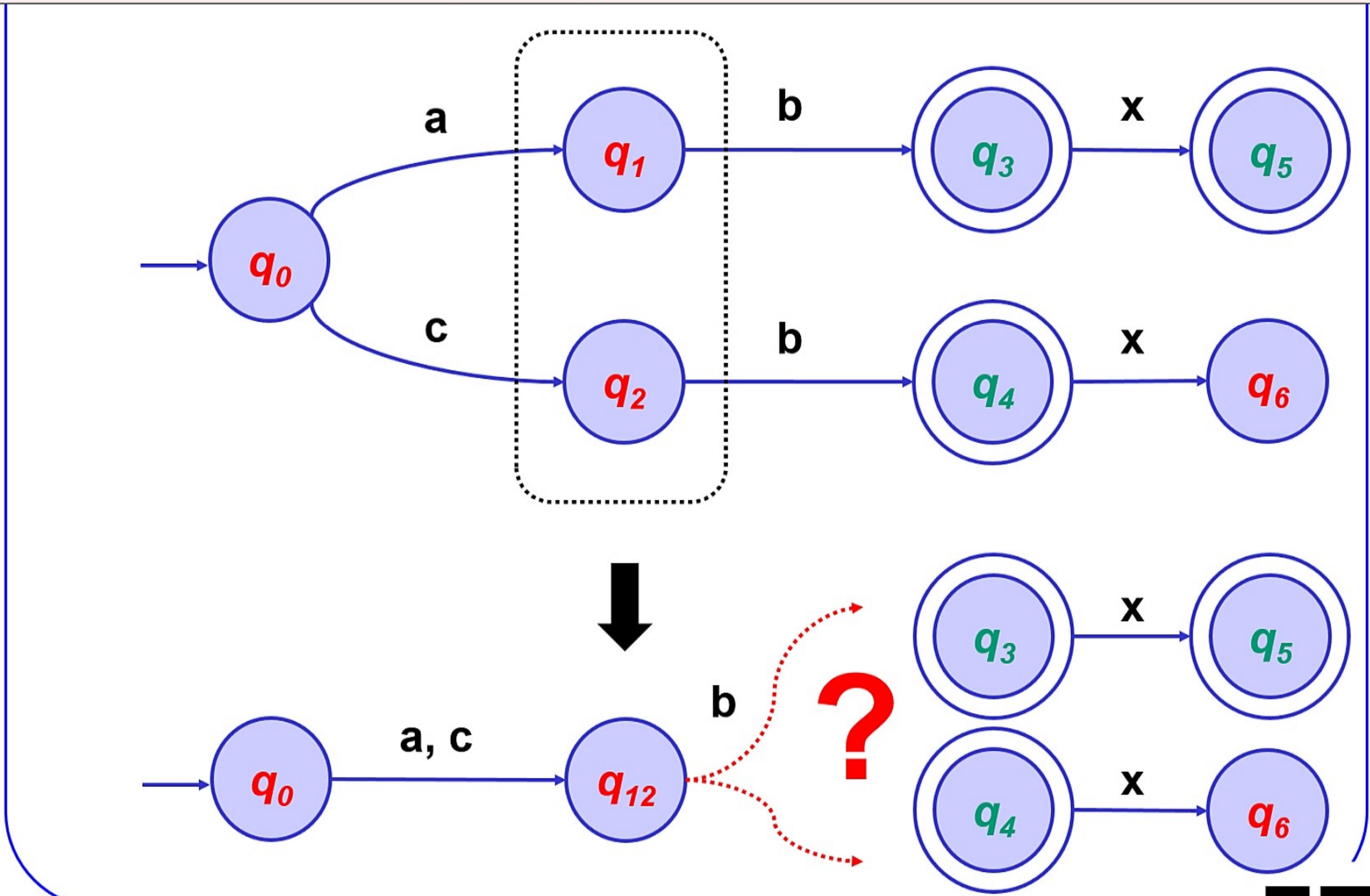
q_1 and q_2 are NOT equivalent
q_3 and q_4 are equivalent



q_3 and q_4 are NOT equivalent $\Rightarrow$ q_1 and q_2 are NOT equivalent



q_5 and q_6 are NOT equivalent $\Rightarrow$
q_3 and q_4 are NOT equivalent $\Rightarrow$
q_1 and q_2 are NOT equivalent



Minimization of DFA

- **Equivalence test for states *p* and *q***
  - **Testing of two conditions:**

Compatibility condition:

States ***p*** and ***q*** are either both accepting ($p \in F \wedge q \in F$) or both nonaccepting ($p \notin F \wedge q \notin F$).

Propagation condition:

For each input symbol ***a***, states $\delta(p, a)$ and $\delta(q, a)$ are equivalent.

Minimization of DFA – *table-filling algorithm*

	0	1	$\perp$
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

Minimization of DFA – *table-filling algorithm*

	0	1	$\perp$
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

p_2							
p_3							
p_4							
p_5							
p_6							
p_7							
p_8							
	p_1	p_2	p_3	p_4	p_5	p_6	p_7

1

Mark all pairs (p, q) for which holds that $p \in F$ and $q \notin F$

Minimization of DFA – *table-filling algorithm*

	0	1	⊥
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

p_2							
p_3	X	X					
p_4			X				
p_5			X				
p_6			X				
p_7			X				
p_8			X				
	p_1	p_2	p_3	p_4	p_5	p_6	p_7

1

Mark all pairs (p, q) for which holds that $p \in F$ and $q \notin F$

Minimization of DFA – *table-filling algorithm*

	0	1	$\perp$
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_2							
p_3	X	X					
p_4			X				
p_5			X				
p_6			X				
p_7			X				
p_8			X				

2

For any unmarked pair of states (p, q)

if the pair $(\delta(p,a), \delta(q,a))$ is already marked

Mark (p,q);

Recursively mark all the pairs in the list assigned to the marked pair

else

if $\delta(p,a) \;!= \delta(q,a)$

put the pair (p,q) into the list assigned to the pair $(\delta(p,a), \delta(q,a))$

Minimization of DFA – *table-filling algorithm*

	0	1	⊥
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_2							
p_3	X	X					
p_4			X				
p_5			X				
p_6			X				
p_7			X				
p_8			X				

2

For any unmarked pair of states (p, q)

if the pair ($\delta(p,a)$, $\delta(q,a)$) is already marked

Mark (p,q);

Recursively mark all the pairs in the list assigned to the marked pair

else

if $\delta(p,a) \; != \; \delta(q,a)$

put the pair (p,q) into the list assigned to the pair ($\delta(p,a)$, $\delta(q,a)$)

Minimization of DFA – *table-filling algorithm*

	0	1	⊥
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

p_2	X						
p_3	X	X					
p_4			X				
p_5			X				
p_6			X				
p_7			X				
p_8			X				
	p_1	p_2	p_3	p_4	p_5	p_6	p_7

2

For any unmarked pair of states (*p*, *q*)

if the pair ($\delta(p,a)$, $\delta(q,a)$) is already marked

Mark (*p*,*q*);

Recursively mark all the pairs in the list assigned to the marked pair

else

if $\delta(p,a)$! = $\delta(q,a)$

put the pair (*p*,*q*) into the list assigned to the pair ($\delta(p,a)$, $\delta(q,a)$)

Minimization of DFA – *table-filling algorithm*

	0	1	$\perp$
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_2	X						
p_3	X	X					
p_4	X		X				
p_5			X				
p_6			X				
p_7			X				
p_8			X				

Minimization of DFA – *table-filling algorithm*

	0	1	⊥
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_2	X						
p_3	X	X					
p_4	X		X				
p_5			X				
p_6			X				
p_7			X				
p_8			X				

Minimization of DFA – *table-filling algorithm*

	0	1	⊥
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_2	X						
p_3	X	X					
p_4	X		X				
p_5			X				
p_6			X				
p_7			X				
p_8			X				

List assigned to $(p_2, p_8) = \{(p_1, p_5)\}$

Minimization of DFA – *table-filling algorithm*

	0	1	⊥
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_2	X						
p_3	X	X					
p_4	X		X				
p_5			X				
p_6			X				
p_7			X				
p_8			X				

List assigned to $(p_2, p_8) = \{(p_1, p_5)\}$

Minimization of DFA – *table-filling algorithm*

	0	1	⊥
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_2	X						
p_3	X	X					
p_4	X		X				
p_5			X				
p_6	X		X				
p_7			X				
p_8	X		X				

List assigned to (p_2, p_8) = {(p_1, p_5)}

Minimization of DFA – *table-filling algorithm*

	0	1	$\perp$
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_2	X						
p_3	X	X					
p_4	X		X				
p_5			X				
p_6	X		X				
p_7			X				
p_8	X		X				

List assigned to (p_2, p_8) = {(p_1, p_5)}

Minimization of DFA – *table-filling algorithm*

	0	1	⊥
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_2	X						
p_3	X	X					
p_4	X		X				
p_5			X				
p_6	X		X				
p_7			X				
p_8	X		X				

List assigned to $(p_2, p_8) = \{(p_1, p_5)\}$

List assigned to $(p_2, p_7) = \{(p_1, p_7)\}$

List assigned to $(p_5, p_6) = \{(p_1, p_7)\}$

Minimization of DFA – *table-filling algorithm*

	0	1	⊥
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

p_2	X						
p_3	X	X					
p_4	X	X	X				
p_5		X	X				
p_6	X	X	X				
p_7		X	X				
p_8	X		X				
	p_1	p_2	p_3	p_4	p_5	p_6	p_7

List assigned to $(p_2, p_8) = \{(p_1, p_5)\}$

List assigned to $(p_2, p_7) = \{(p_1, p_7)\}$

List assigned to $(p_5, p_6) = \{(p_1, p_7)\}$

Minimization of DFA – *table-filling algorithm*

	0	1	⊥
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_2	X						
p_3	X	X					
p_4	X	X	X				
p_5		X	X				
p_6	X	X	X				
p_7		X	X				
p_8	X		X				

List assigned to $(p_2, p_8) = \{(p_1, p_5)\}$

List assigned to $(p_2, p_7) = \{(p_1, p_7)\}$

List assigned to $(p_5, p_6) = \{(p_1, p_7)\}$

Minimization of DFA – *table-filling algorithm*

	0	1	⊥
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_2	X						
p_3	X	X					
p_4	X	X	X				
p_5		X	X				
p_6	X	X	X				
p_7	X	X	X				
p_8	X		X				

List assigned to $(p_2, p_8) = \{(p_1, p_5)\}$

List assigned to $(p_2, p_7) = \{(p_1, p_7)\}$

List assigned to $(p_5, p_6) = \{(p_1, p_7)\}$

Minimization of DFA – *table-filling algorithm*

	0	1	⊥
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_2	X						
p_3	X	X					
p_4	X	X	X				
p_5		X	X	X			
p_6	X	X	X				
p_7	X	X	X	X			
p_8	X		X	X			

List assigned to $(p_2, p_8) = \{(p_1, p_5)\}$

List assigned to $(p_5, p_6) = \{(p_1, p_7)\}$

Minimization of DFA – *table-filling algorithm*

	0	1	$\perp$
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_2	X						
p_3	X	X					
p_4	X	X	X				
p_5		X	X	X			
p_6	X	X	X		X		
p_7	X	X	X	X			
p_8	X		X	X			

List assigned to $(p_2, p_8) = \{(p_1, p_5)\}$

List assigned to $(p_5, p_6) = \{(p_1, p_7)\}$

Minimization of DFA – *table-filling algorithm*

	0	1	⊥
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_2	X						
p_3	X	X					
p_4	X	X	X				
p_5		X	X	X			
p_6	X	X	X		X		
p_7	X	X	X	X			
p_8	X		X	X			

List assigned to $(p_2, p_8) = \{(p_1, p_5)\}$

List assigned to $(p_5, p_6) = \{(p_1, p_7)\}$

Minimization of DFA – *table-filling algorithm*

	0	1	⊥
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_2	X						
p_3	X	X					
p_4	X	X	X				
p_5		X	X	X			
p_6	X	X	X		X		
p_7	X	X	X	X	X	X	
p_8	X		X	X	X	X	X

List assigned to $(p_2, p_8) = \{(p_1, p_5)\}$

Minimization of DFA – *table-filling algorithm*

	0	1	⊥
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
p_2	X						
p_3	X	X					
p_4	X	X	X				
p_5		X	X	X			
p_6	X	X	X		X		
p_7	X	X	X	X	X	X	
p_8	X		X	X	X	X	X

List assigned to $(p_2, p_8) = \{(p_1, p_5)\}$

Equivalent states:

(p_1, p_5), (p_2, p_8), (p_4, p_6)

Minimization of DFA – *table-filling algorithm*

	0	1	$\perp$
p_1	p_2	p_6	0
p_2	p_7	p_3	0
p_3	p_1	p_3	1
p_4	p_3	p_7	0
p_5	p_8	p_6	0
p_6	p_3	p_7	0
p_7	p_7	p_5	0
p_8	p_7	p_3	0

	0	1	$\perp$
$[p_1, p_5]$	$[p_2, p_8]$	$[p_4, p_6]$	0
$[p_2, p_8]$	p_7	p_3	0
p_3	$[p_1, p_5]$	p_3	1
$[p_4, p_6]$	p_3	p_7	0
p_7	p_7	$[p_1, p_5]$	0

Unreachable States in DFA

	c	d	$\perp$
q_0	q_1	q_5	0
q_1	q_2	q_7	1
q_2	q_2	q_5	1
q_3	q_5	q_7	0
q_4	q_5	q_6	0
q_5	q_3	q_1	0
q_6	q_8	q_0	1
q_7	q_0	q_1	1
q_8	q_3	q_6	0

RS $\boxed{q_0}$

1) Put the initial state q_0 into the list of reachable states *RS*

2) Extend the list *RS* with the states $\{p \mid p=\delta(q_0, a)$, za sve $a \in \Sigma\}$

3) $q_i \in RS$, $RS = RS \cup \{p \mid p=\delta(q_i, a)$, for any $a \in \Sigma\}$

Unreachable States in DFA

	c	d	$\perp$
q_0	q_1	q_5	0
q_1	q_2	q_7	1
q_2	q_2	q_5	1
q_3	q_5	q_7	0
q_4	q_5	q_6	0
q_5	q_3	q_1	0
q_6	q_8	q_0	1
q_7	q_0	q_1	1
q_8	q_3	q_6	0

RS	q_0	q_1	q_5

1) Put the initial state q_0 into the list of reachable states *RS*

2) Extend the list *RS* with the states $\{p \mid p=\delta(q_0, a)$, za sve $a \in \Sigma\}$

3) $q_i \in RS$, $RS = RS \cup \{p \mid p=\delta(q_i, a)$, for any $a \in \Sigma\}$

Unreachable States in DFA

	c	d	$\perp$
q_0	q_1	q_5	0
q_1	q_2	q_7	1
q_2	q_2	q_5	1
q_3	q_5	q_7	0
q_4	q_5	q_6	0
q_5	q_3	q_1	0
q_6	q_8	q_0	1
q_7	q_0	q_1	1
q_8	q_3	q_6	0

RS	q_0	q_1	q_5	q_2	q_7	q_3

1) Put the initial state q_0 into the list of reachable states *RS*

2) Extend the list *RS* with the states $\{p \mid p=\delta(q_0, a)$, za sve $a\in\Sigma\}$

3) $q_i\in RS$, $RS = \mathrm{RS} \cup \{p \mid p=\delta(q_i, a)$, for any $a\in\Sigma\}$

Unreachable States in DFA

	c	d	$\perp$
q_0	q_1	q_5	0
q_1	q_2	q_7	1
q_2	q_2	q_5	1
q_3	q_5	q_7	0
q_4	q_5	q_6	0
q_5	q_3	q_1	0
q_6	q_8	q_0	1
q_7	q_0	q_1	1
q_8	q_3	q_6	0

RS	q_0	q_1	q_5	q_2	q_7	q_3

1) Put the initial state q_0 into the list of reachable states *RS*

2) Extend the list *RS* with the states $\{p \mid p=\delta(q_0, a)$, za sve $a\in\Sigma\}$

3) $q_i\in RS$, $RS = \mathrm{RS} \cup \{p \mid p=\delta(q_i, a)$, for any $a\in\Sigma\}$

Unreachable States in DFA

	c	d	$\perp$
q_0	q_1	q_5	0
q_1	q_2	q_7	1
q_2	q_2	q_5	1
q_3	q_5	q_7	0
q_4	q_5	q_6	0
q_5	q_3	q_1	0
q_6	q_8	q_0	1
q_7	q_0	q_1	1
q_8	q_3	q_6	0

RS					
q_0	q_1	q_5	q_2	q_7	q_3

	c	d	$\perp$
q_0	q_1	q_5	0
q_1	q_2	q_7	1
q_2	q_2	q_5	1
q_3	q_5	q_7	0
q_5	q_3	q_1	0
q_7	q_0	q_1	1

1) Put the initial state q_0 into the list of reachable states *RS*

2) Extend the list *RS* with the states $\{p \mid p=\delta(q_0, a)$, za sve $a \in \Sigma\}$

3) $q_i \in RS$, $RS = \mathrm{RS} \cup \{p \mid p=\delta(q_i, a)$, for any $a \in \Sigma\}$

Minimal DFA

	c	d	$\perp$
p_0	p_0	p_3	0
p_1	p_2	p_5	0
p_2	p_2	p_7	0
p_3	p_6	p_7	0
p_4	p_1	p_6	1
p_5	p_6	p_5	0
p_6	p_6	p_3	1
p_7	p_6	p_3	0

	c	d	$\perp$
A	A	B	0
B	C	B	0
C	C	B	1

	c	d
p_0, A	p_0, A	p_3, B
p_3, B	p_6, C	p_7, B
p_6, C	p_6, C	p_3, B
p_7, B	p_6, C	p_3, B