Properties of Regular Languages

For regular languages L_1 and L_2
we will prove that:

$$\left.\begin{array}{rl} \text{Union:} & L_1 \cup L_2 \\ \text{Concatenation:} & L_1 L_2 \\ \text{Star:} & L_1{}^* \\ \text{Reversal:} & L_1{}^R \\ \text{Complement:} & \overline{L_1} \\ \text{Intersection:} & L_1 \cap L_2 \end{array}\right\} \text{Are regular Languages}$$

We say: Regular languages are **closed under**

Union: $L_1 \cup L_2$

Concatenation: $L_1 L_2$

Star: $L_1{}^*$

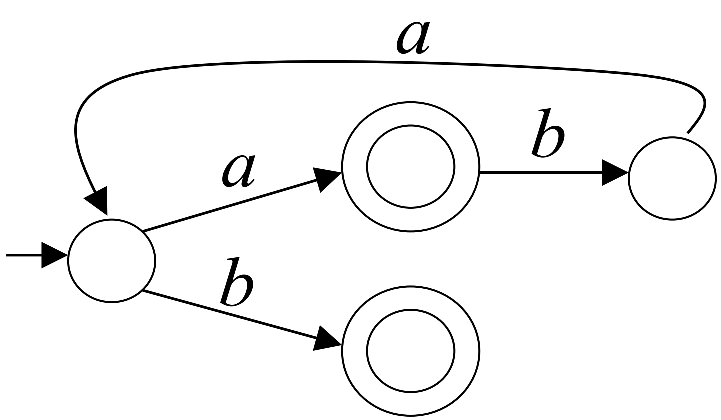
Reversal: $L_1{}^R$

Complement: $\overline{L_1}$

Intersection: $L_1 \cap L_2$

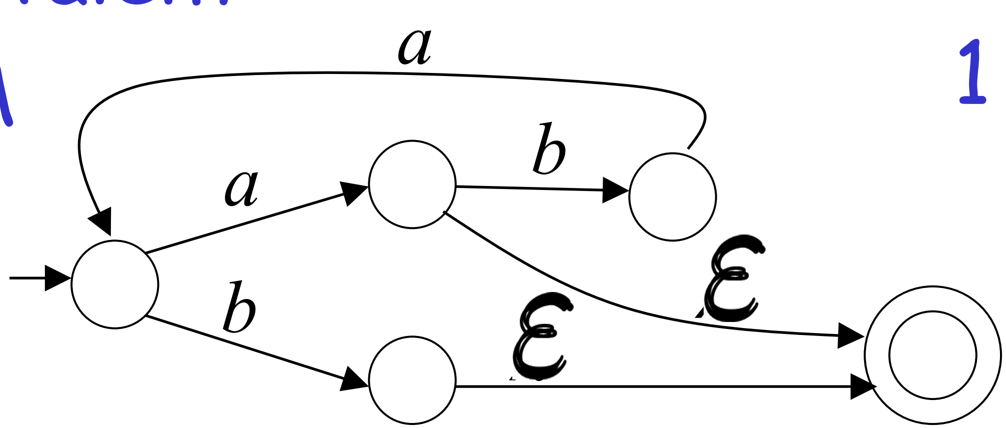
A useful transformation: use one accept state

NFA



2 accept states

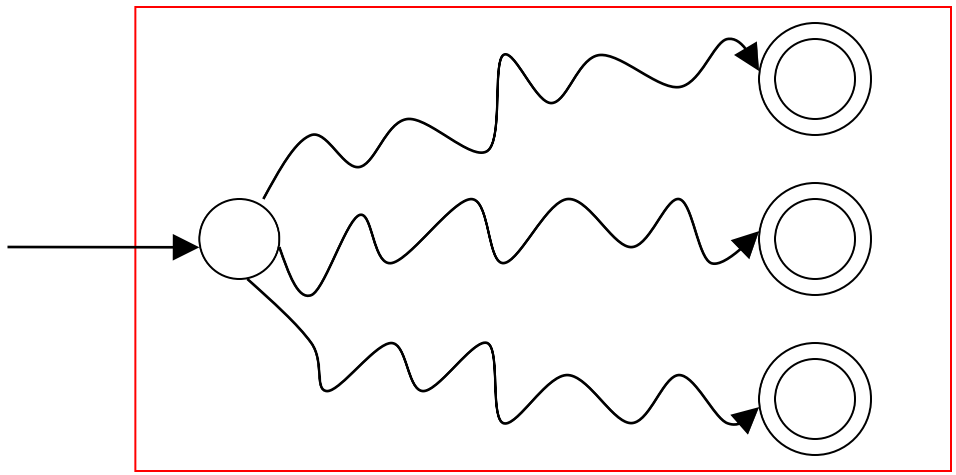
Equivalent
NFA



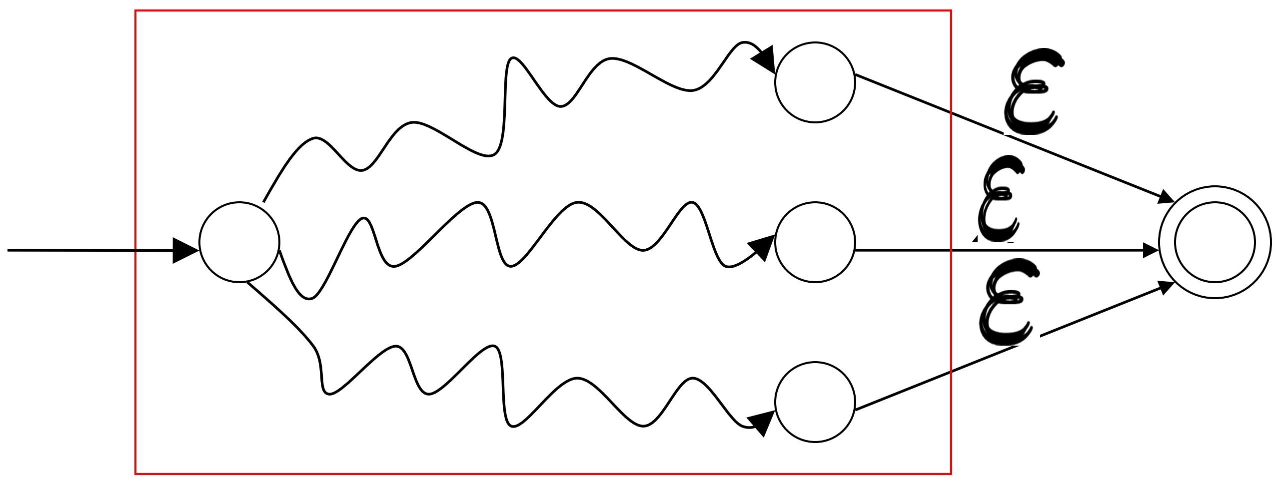
1 accept state

In General

NFA



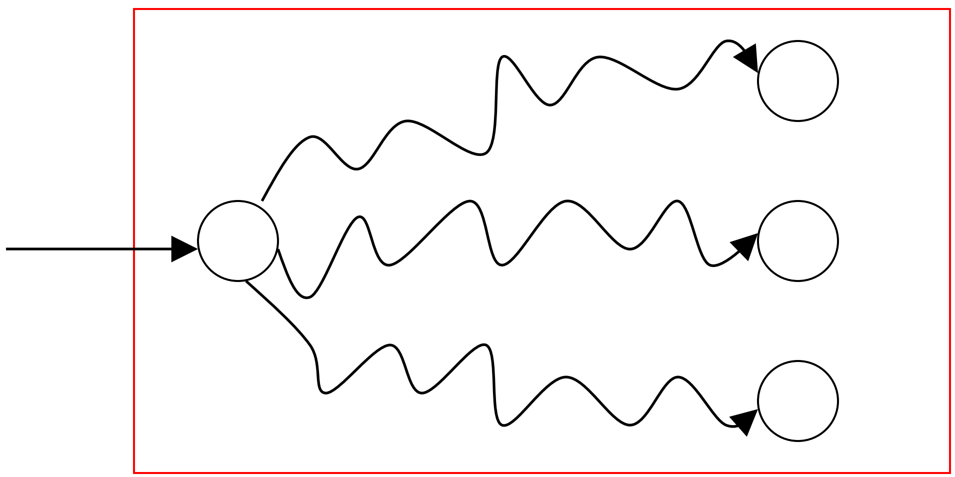
Equivalent NFA

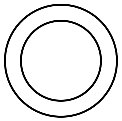


Single accepting state

Extreme case

NFA without accepting state





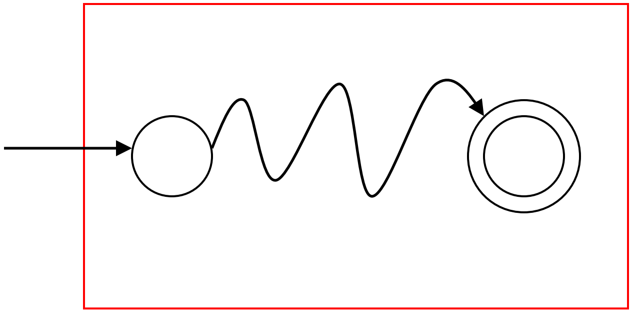
Add an accepting state without transitions

Take two languages

Regular language L_1

$$L(M_1) = L_1$$

NFA M_1

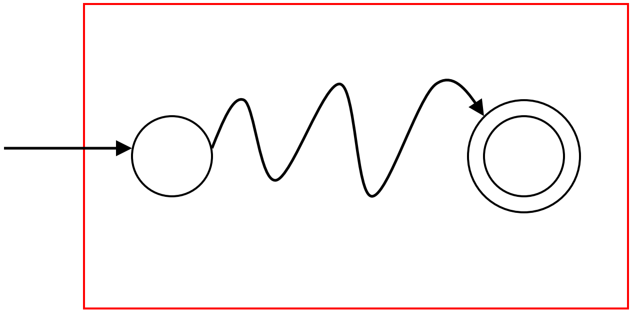


Single accepting state

Regular language L_2

$$L(M_2) = L_2$$

NFA M_2



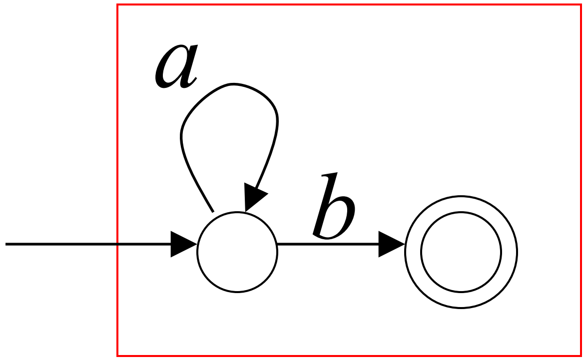
Single accepting state

Example

$n \geq 0$

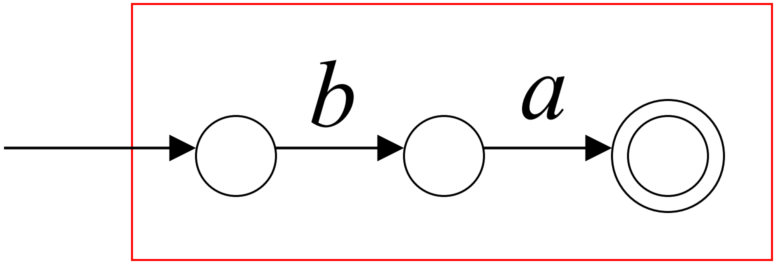
$$L_1 = \{a^n b\}$$

M_1



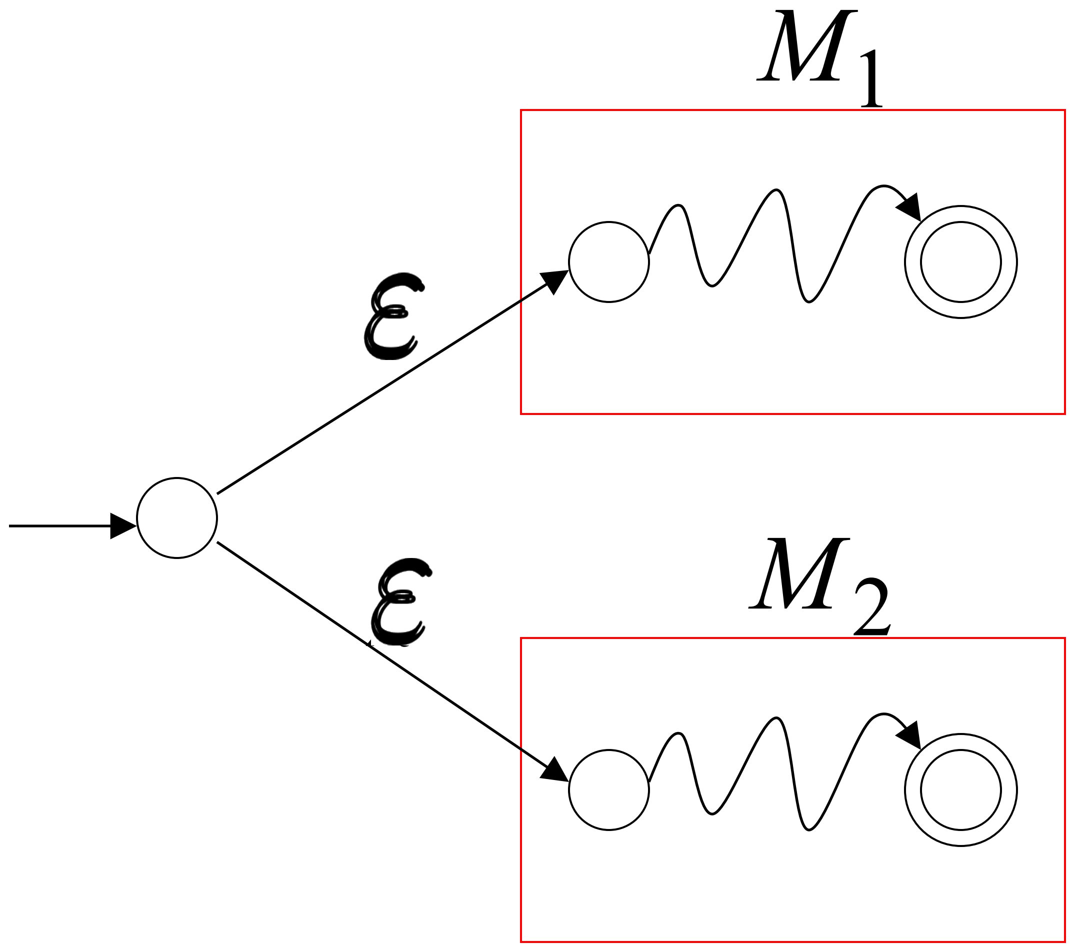
$$L_2 = \{ba\}$$

M_2



Union

NFA for $L_1 \cup L_2$

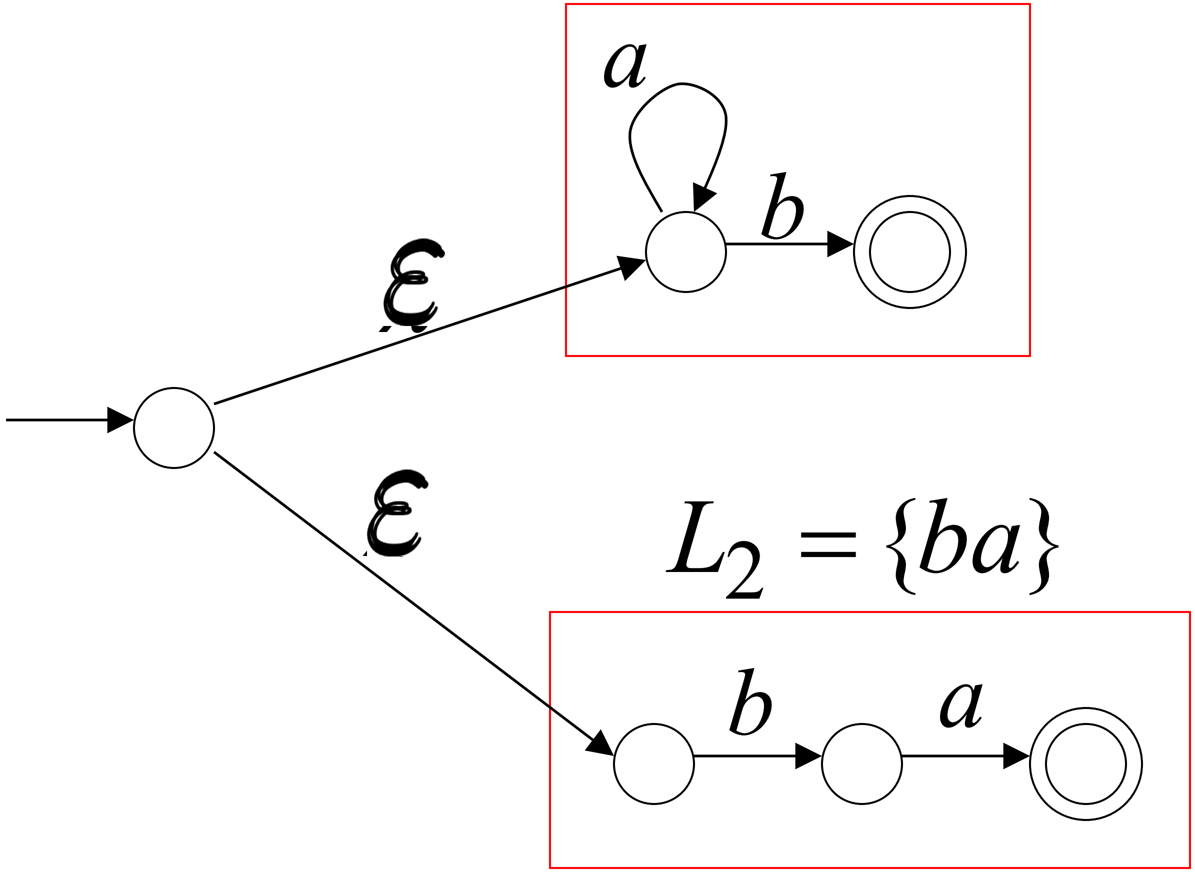


Example

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

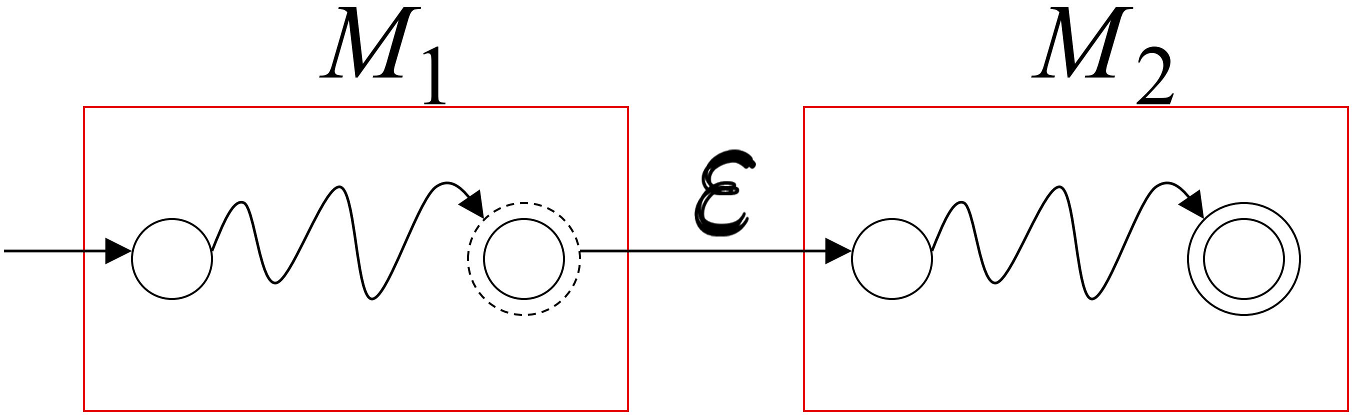
$L_1 = \{a^n b\}$

$L_2 = \{ba\}$



Concatenation

NFA for $L_1 L_2$

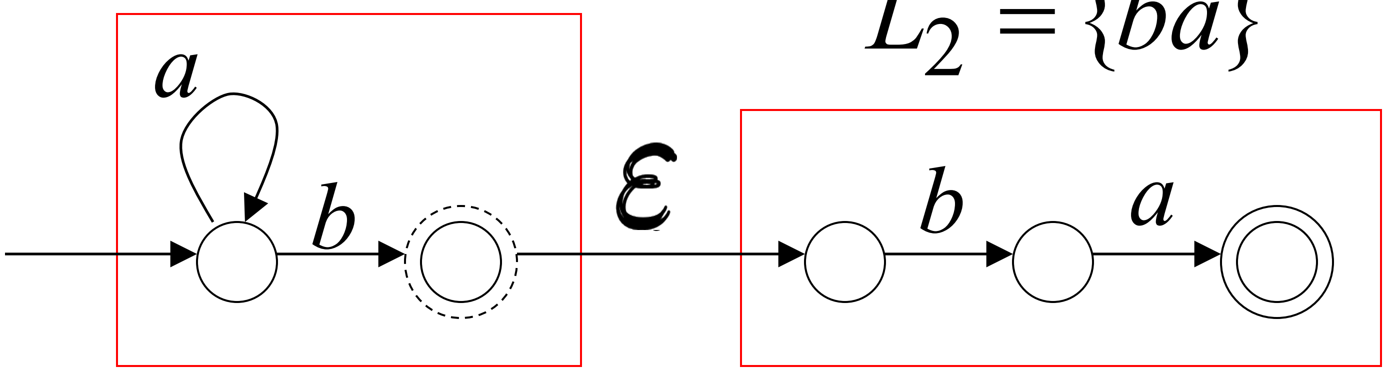


Example

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

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

$L_2 = \{ba\}$

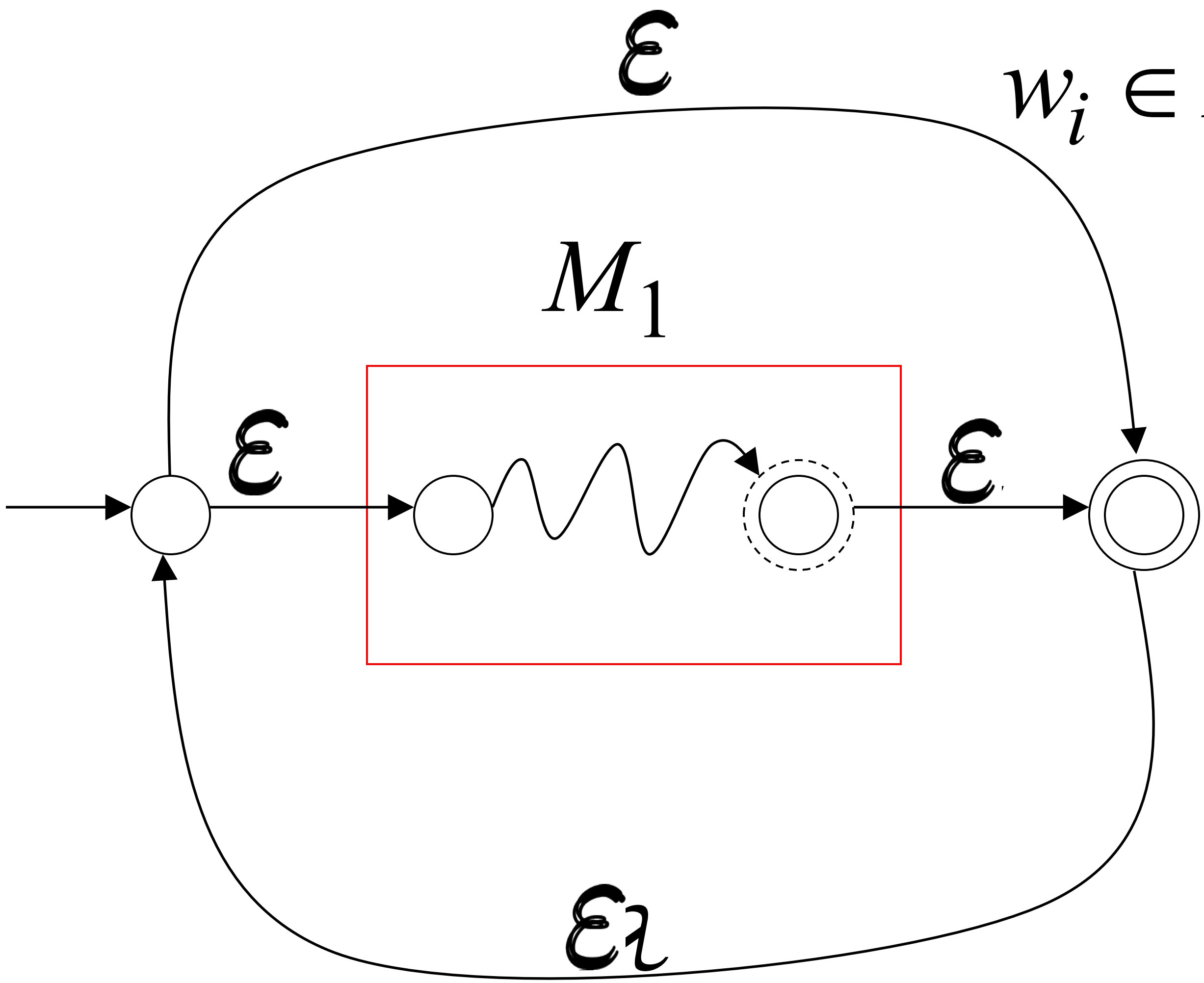


Star Operation

NFA for v$L_1{}^*$

$$w = w_1 w_2 \cdots w_k$$

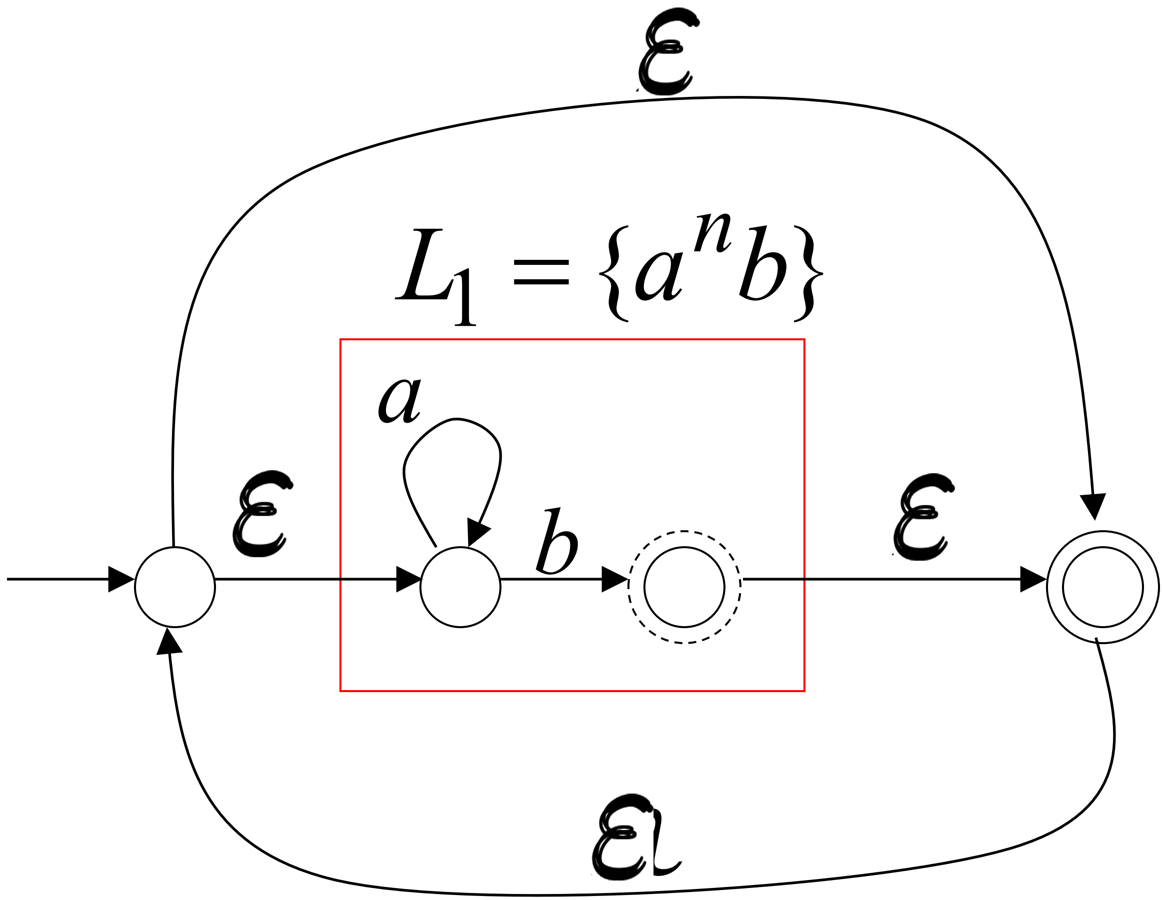
$$w_i \in L_1$$



$$\varepsilon \in L_1{}^*$$

Example

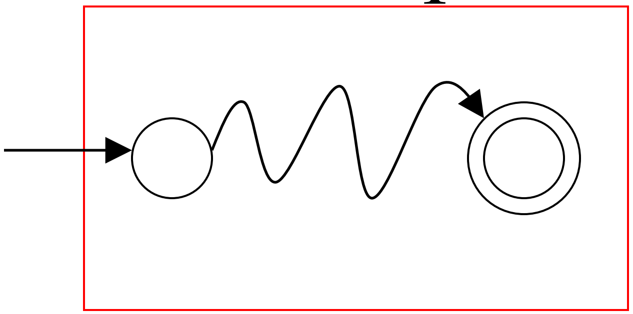
NFA for $L_1{}^* = \{a^n b\}^*$



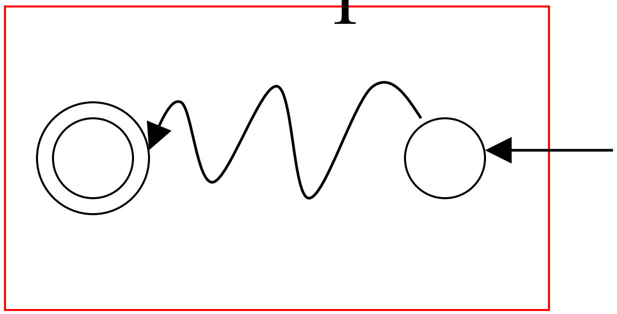
Reverse

NFA for L_1^R

L_1 M_1



M_1'



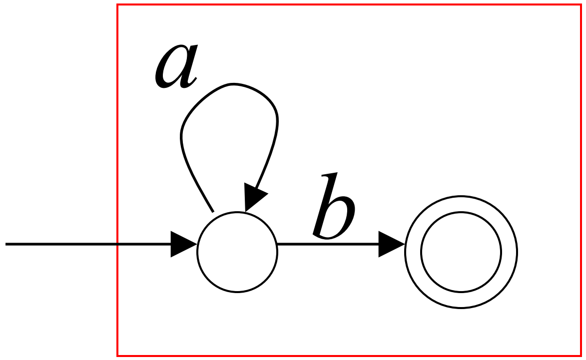
1. Reverse all transitions

2. Make initial state accepting state
   and vice versa

Example

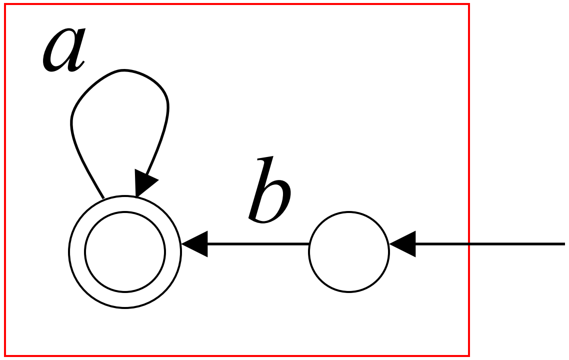
M_1

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

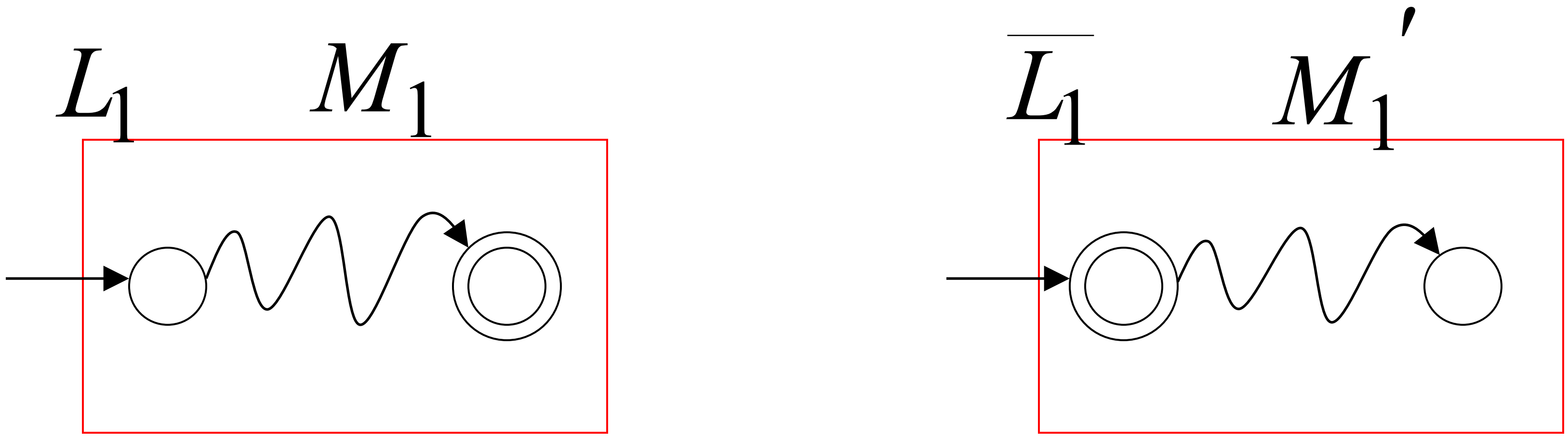


M_1'

$$L_1^R = \{ba^n\}$$



Complement

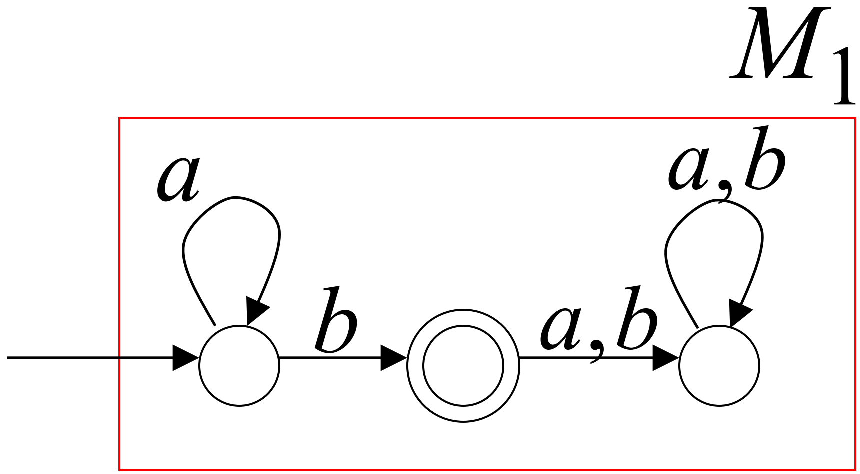


1. Take the DFA that accepts L_1

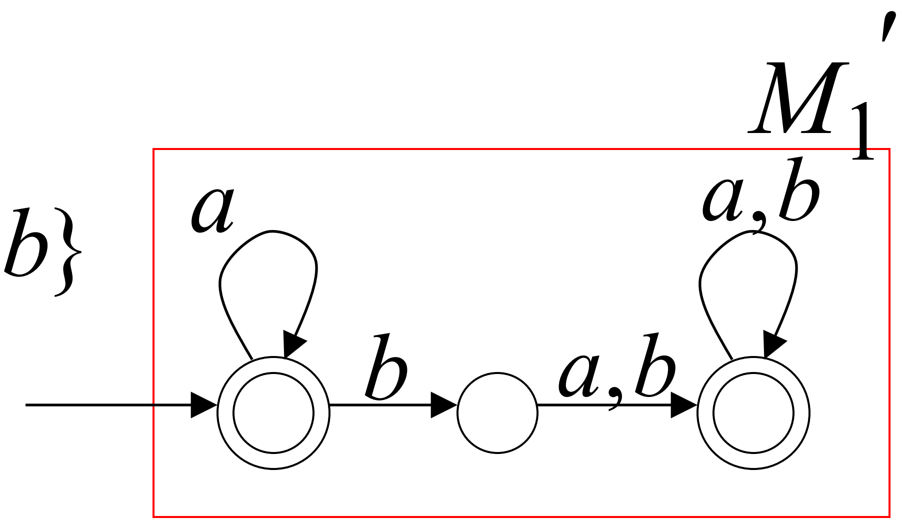
2. Make accepting states non-final, and vice-versa

Example

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



$$\overline{L_1} = \{a,b\}^* - \{a^n b\}$$



Intersection

L_1 regular

L_2 regular

We show

$L_1 \cap L_2$

regular

DeMorgan's Law: $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$

L_1 , L_2 regular

$\overline{L_1}$, $\overline{L_2}$ regular

$\overline{L_1} \cup \overline{L_2}$ regular

$\overline{\overline{L_1} \cup \overline{L_2}}$ regular

$L_1 \cap L_2$ regular

Example

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

$$L_2 = \{ab, ba\}$$ regular

$$\Rightarrow L_1 \cap L_2 = \{ab\}$$

regular

Another Proof for Intersection Closure

Machine M_1

DFA for L_1

Machine M_2

DFA for L_2

Construct a new DFA M that accepts $L_1 \cap L_2$

M simulates in parallel M_1 and M_2

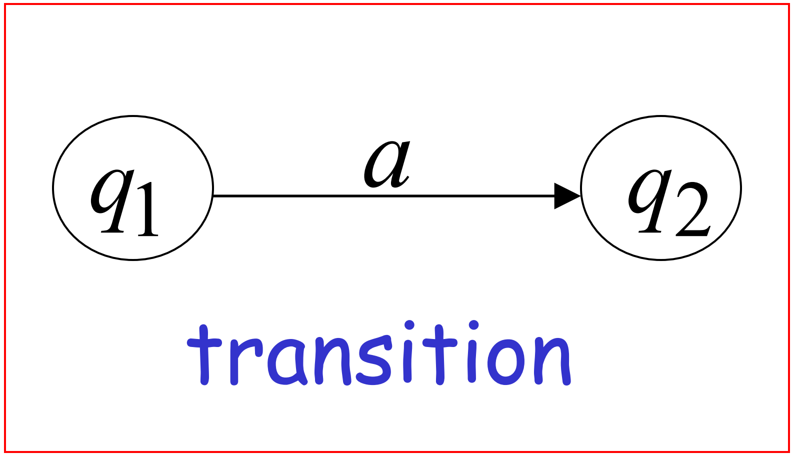
States in M

q_i, p_j

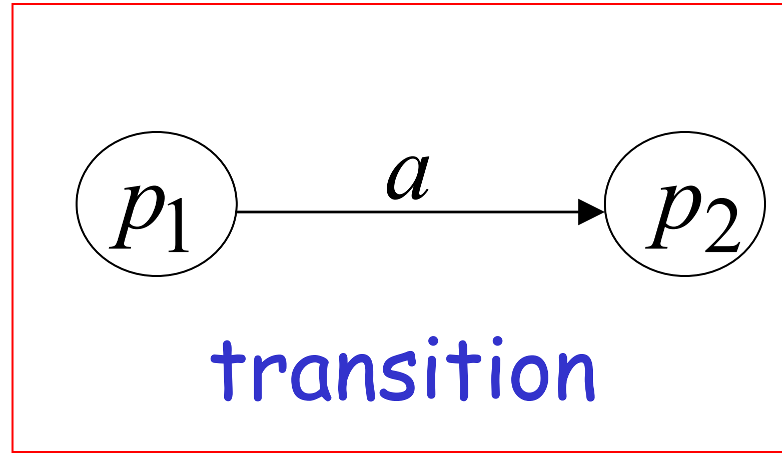
State in M_1

State in M_2

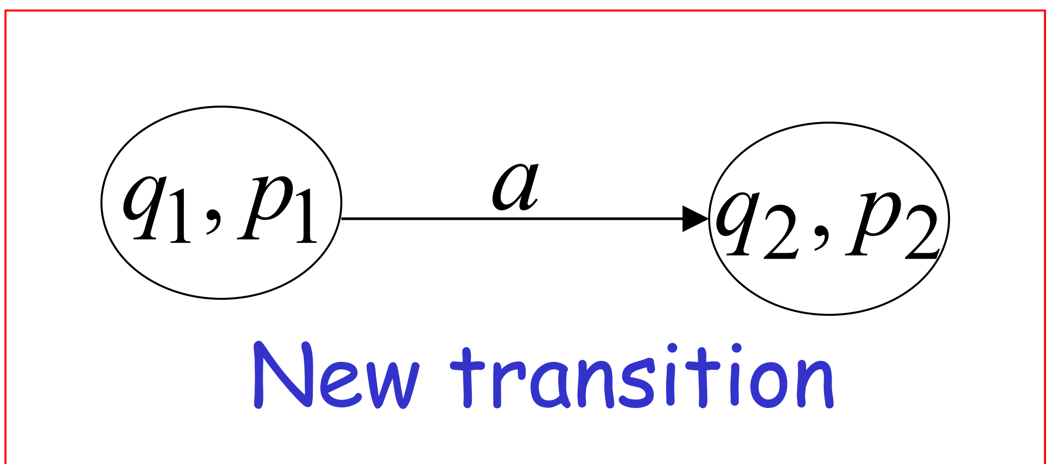
DFA M_1

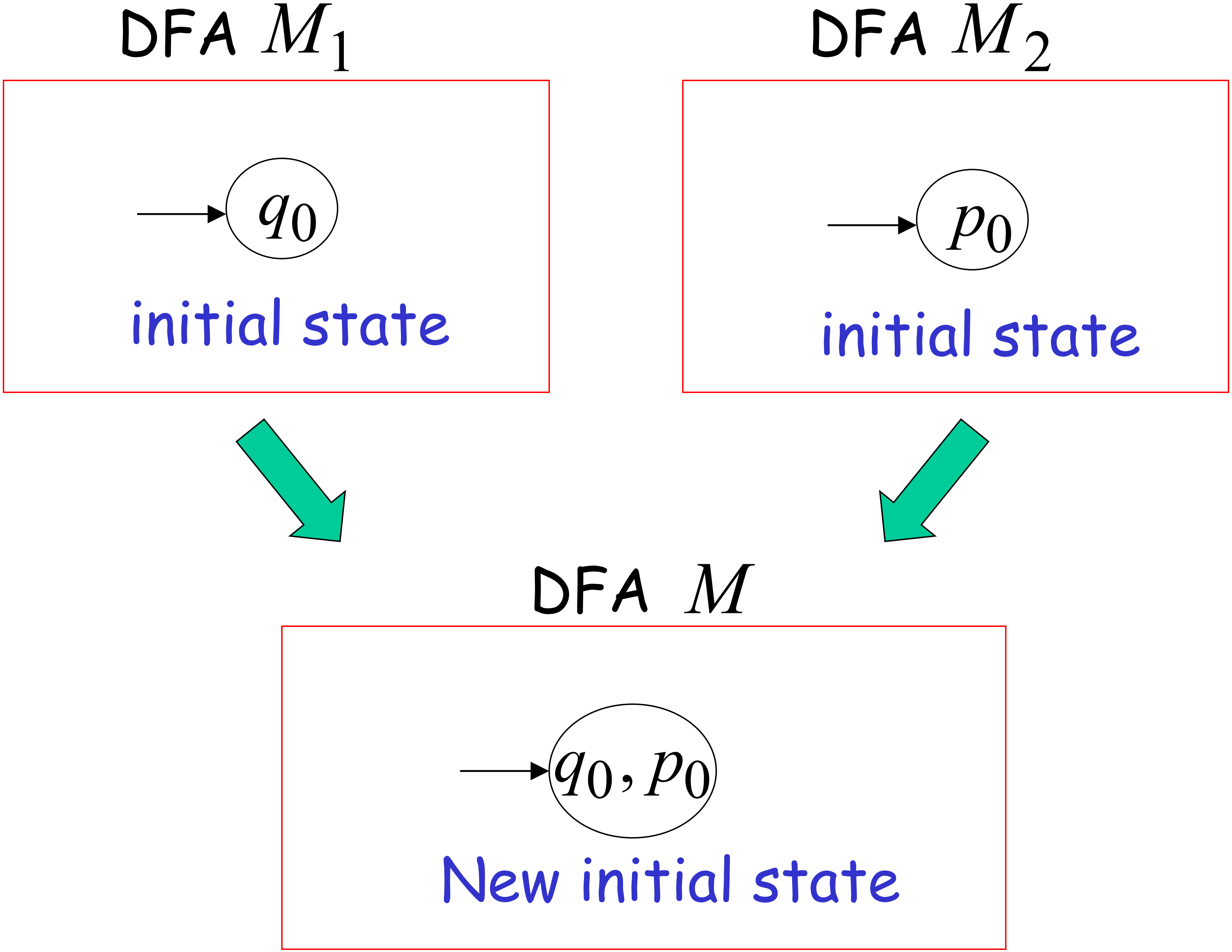


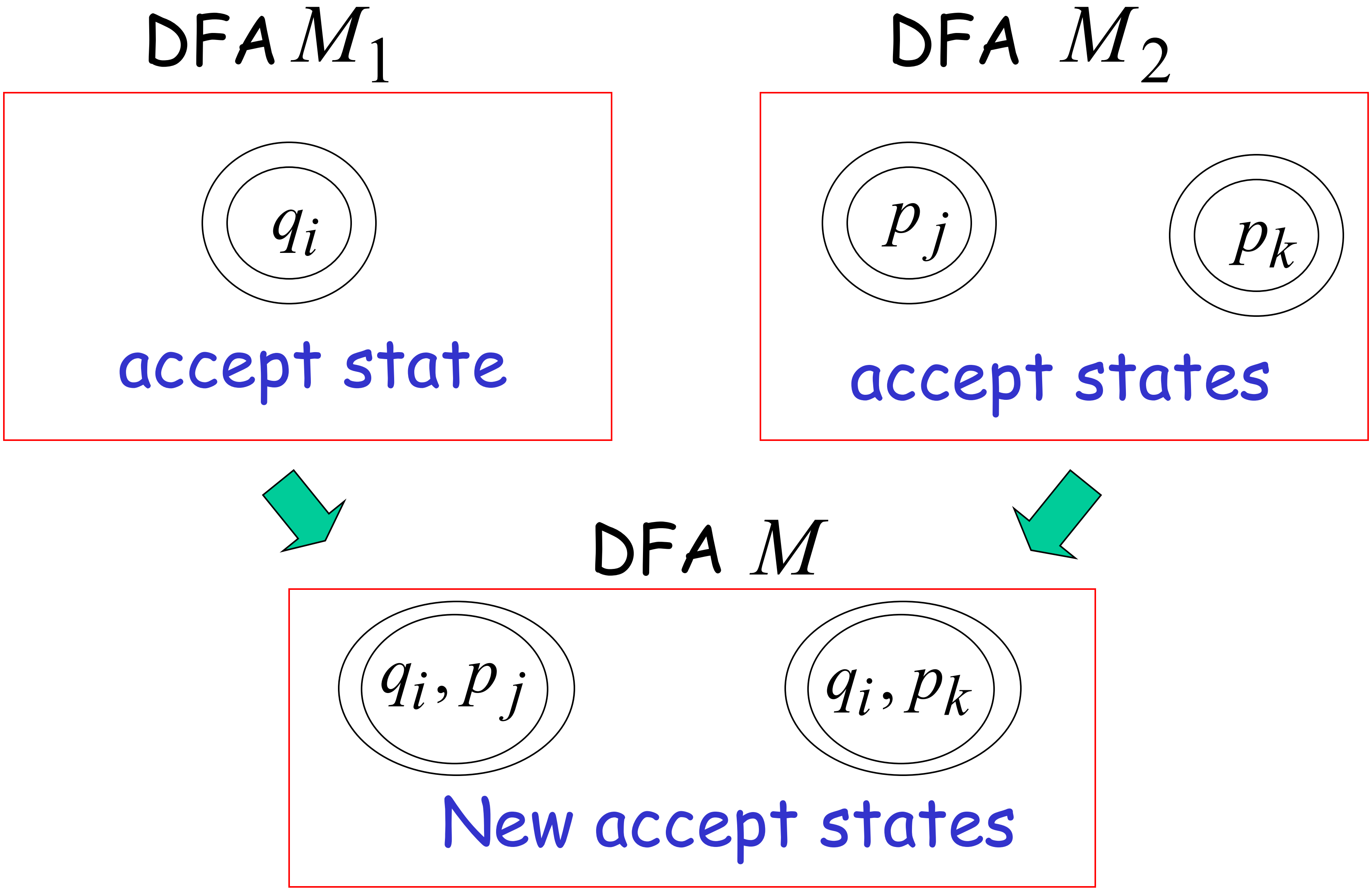
DFA M_2



DFA M




DFA M_1
q_0
initial state
DFA M_2
p_0
initial state
DFA M
q_0, p_0
New initial state



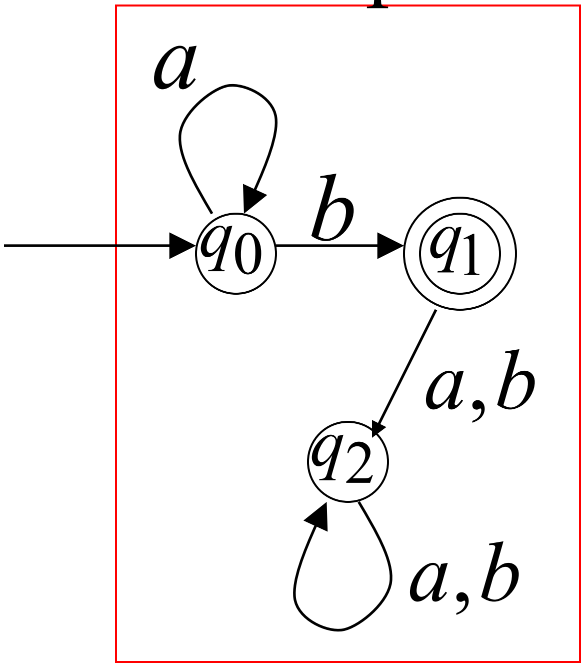
Both constituents must be accepting states

Example:

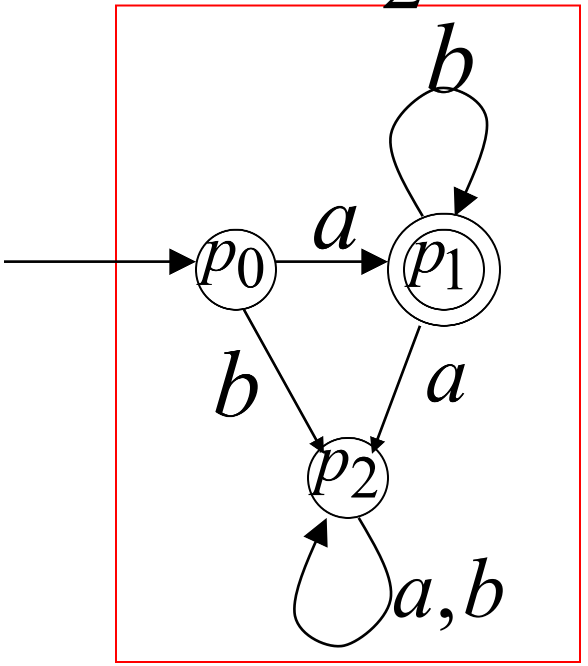
$$L_1 = \{a^n b\} \quad n \geq 0$$

$$L_2 = \{ab^m\} \quad m \geq 0$$

M_1

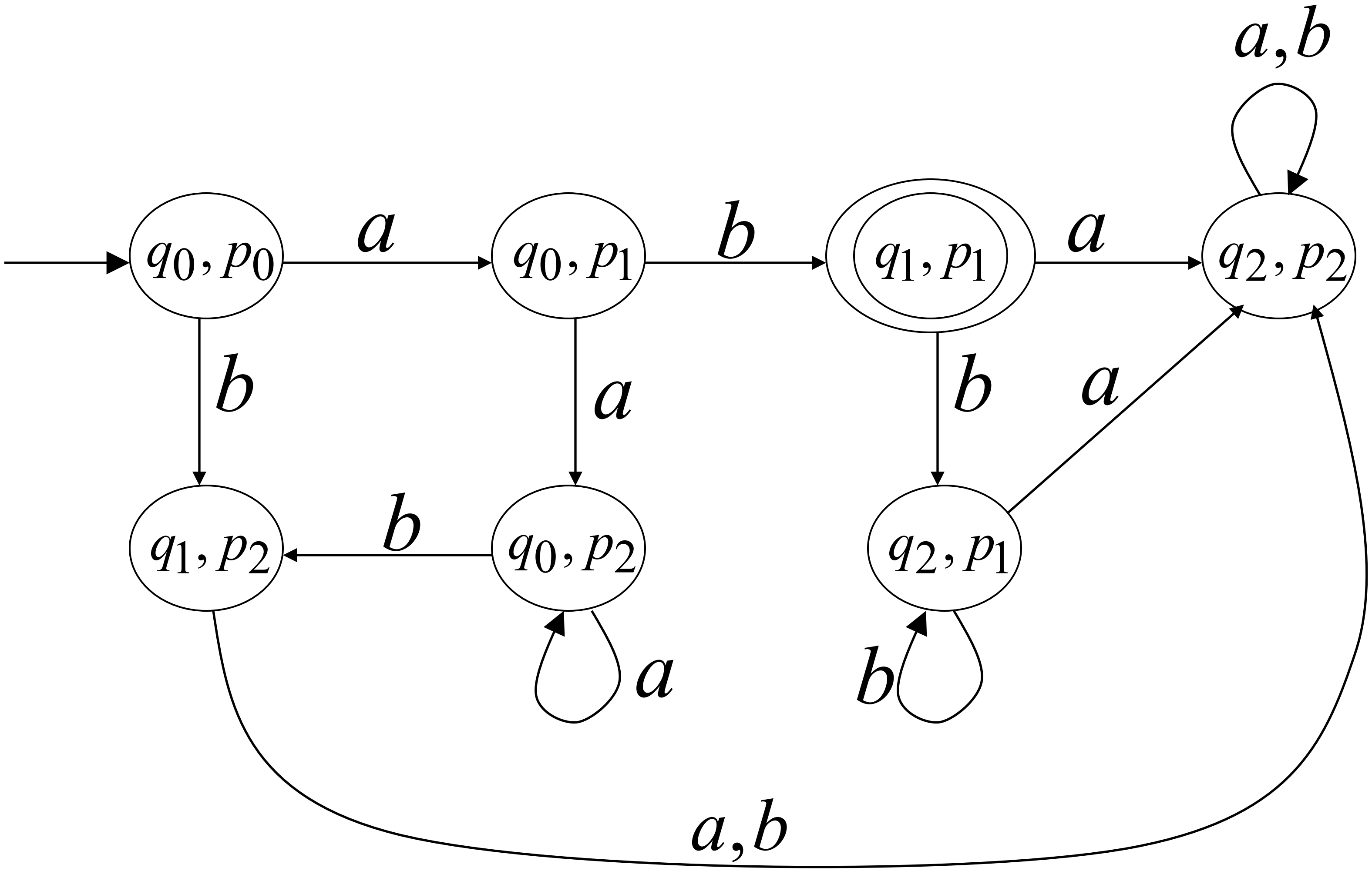


M_2



Automaton for intersection

$$L = \{a^n b\} \cap \{ab^n\} = \{ab\}$$



M simulates in parallel M_1 and M_2

M accepts string w if and only if:

M_1 accepts string w

and M_2 accepts string w

$$L(M) = L(M_1) \cap L(M_2)$$