

Properties of Regular Languages

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^*

Reversal: L_1^R

Complement: $\overline{L_1}$

Intersection: $L_1 \cap L_2$

Are regular
Languages

We say Regular languages are **closed under**

Union: $L_1 \cup L_2$

Concatenation: $L_1 L_2$

Star: L_1^*

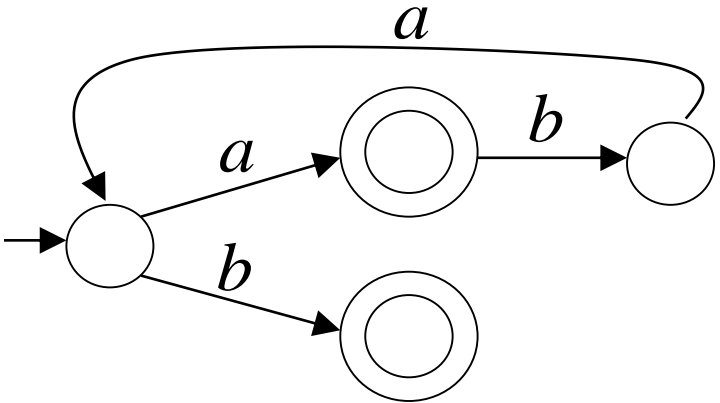
Reversal: L_1^R

Complement: $\overline{L_1}$

Intersection: $L_1 \cap L_2$

A useful transformation: use one accept state

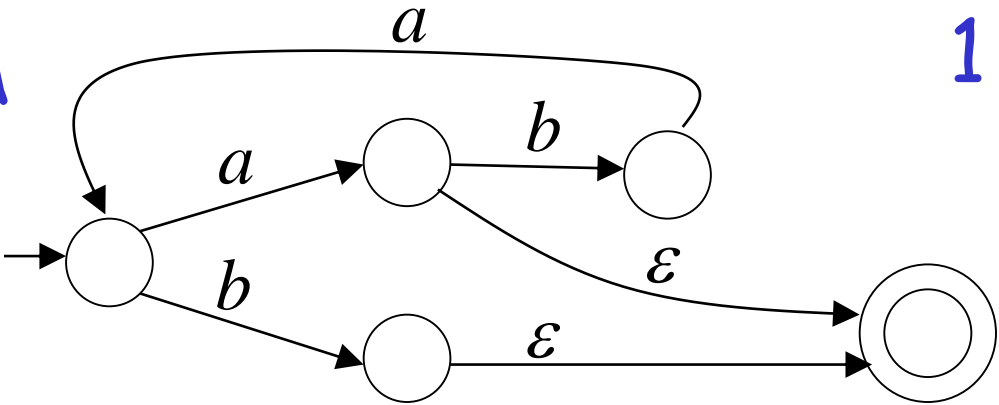
NFA



2 accept states

Equivalent

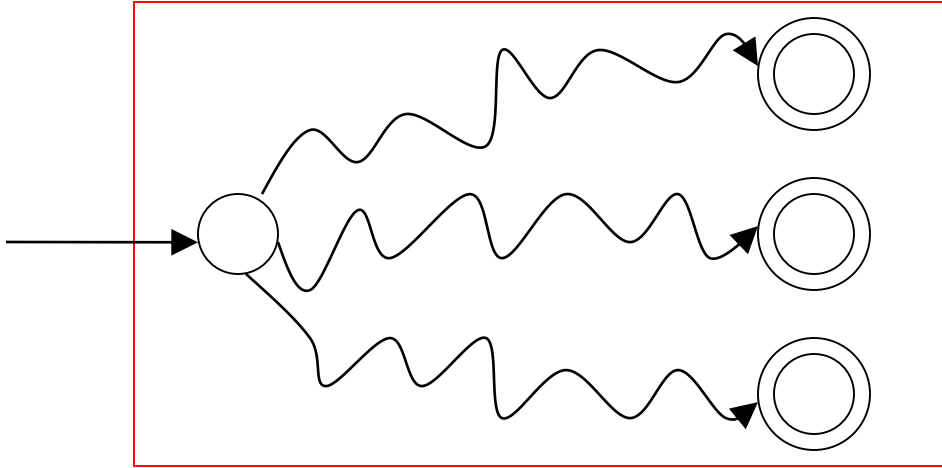
NFA



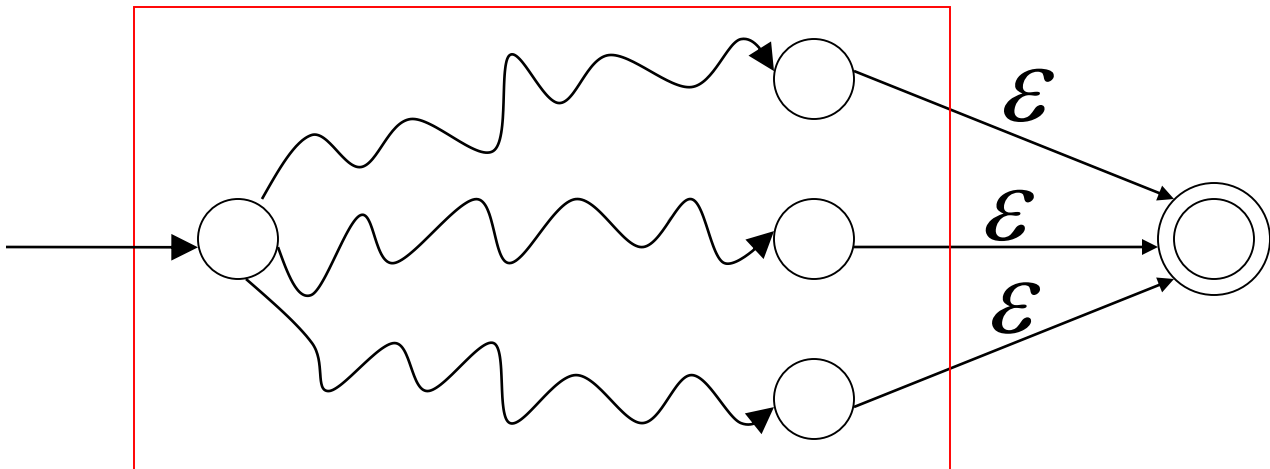
1 accept state

In General

NFA



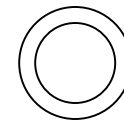
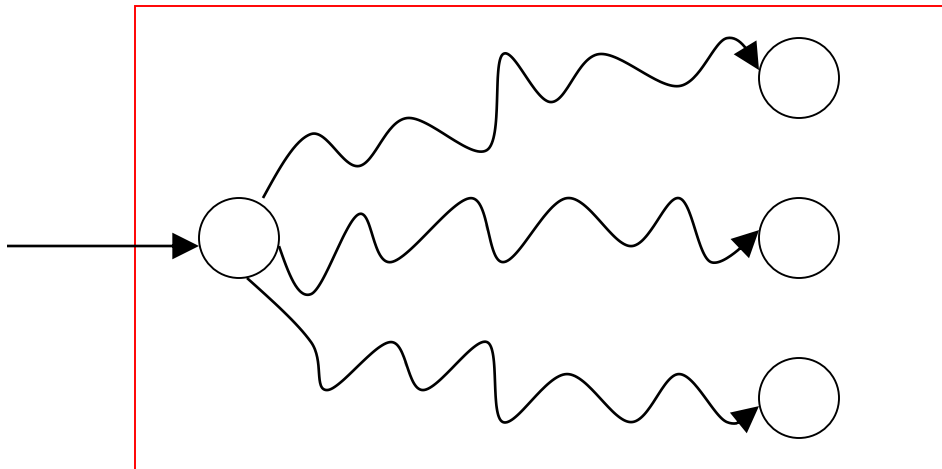
Equivalent NFA



Single
accepting
state

Extreme case

NFA without accepting state



Add an accepting state
without transitions

Take two languages

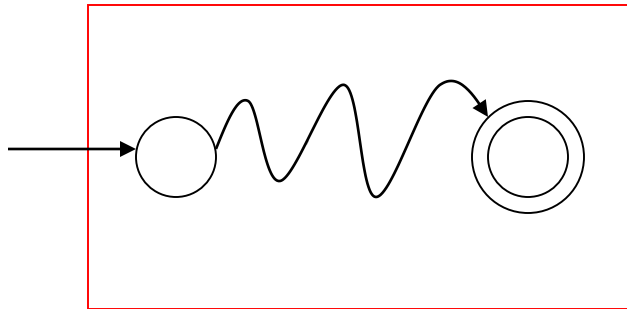
Regular language L_1

Regular language L_2

$$L(M_1) = L_1$$

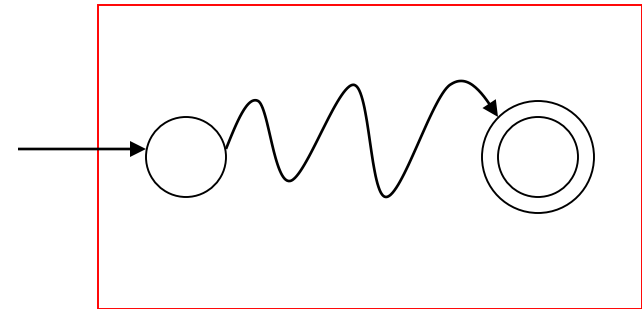
$$L(M_2) = L_2$$

NFA M_1



Single accepting state

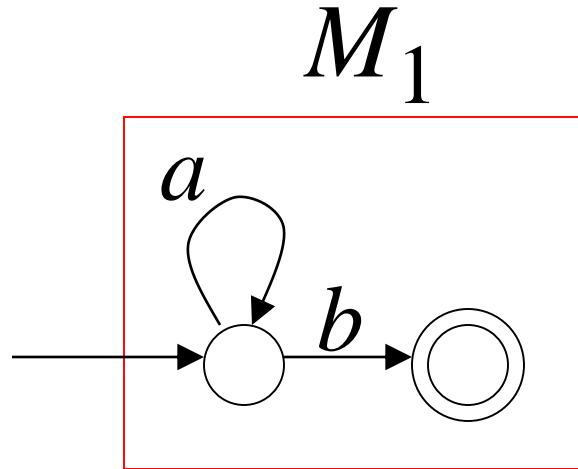
NFA M_2



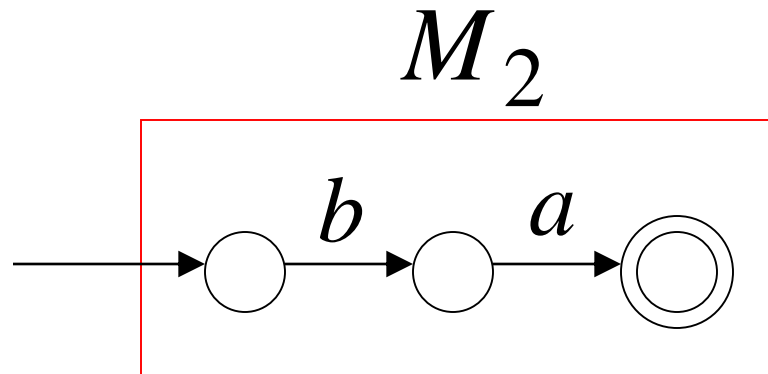
Single accepting state

Example

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

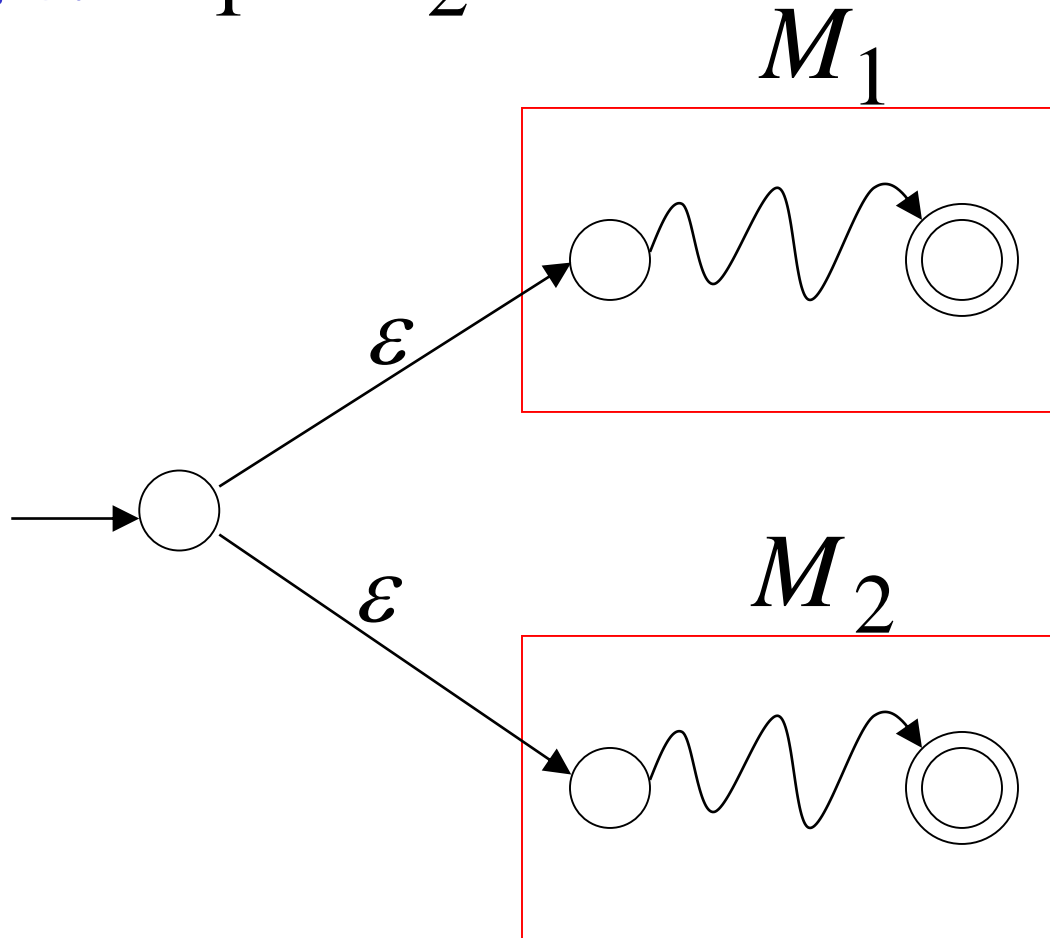


$$L_2 = \{ba\}$$



Union

NFA for $L_1 \cup L_2$

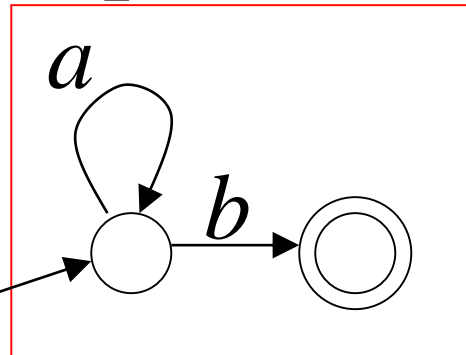


$$w \in L_1 \cup L_2 \iff w \in L_1 \text{ or } w \in L_2$$

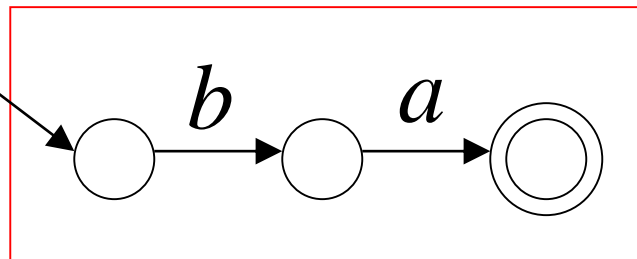
Example

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

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

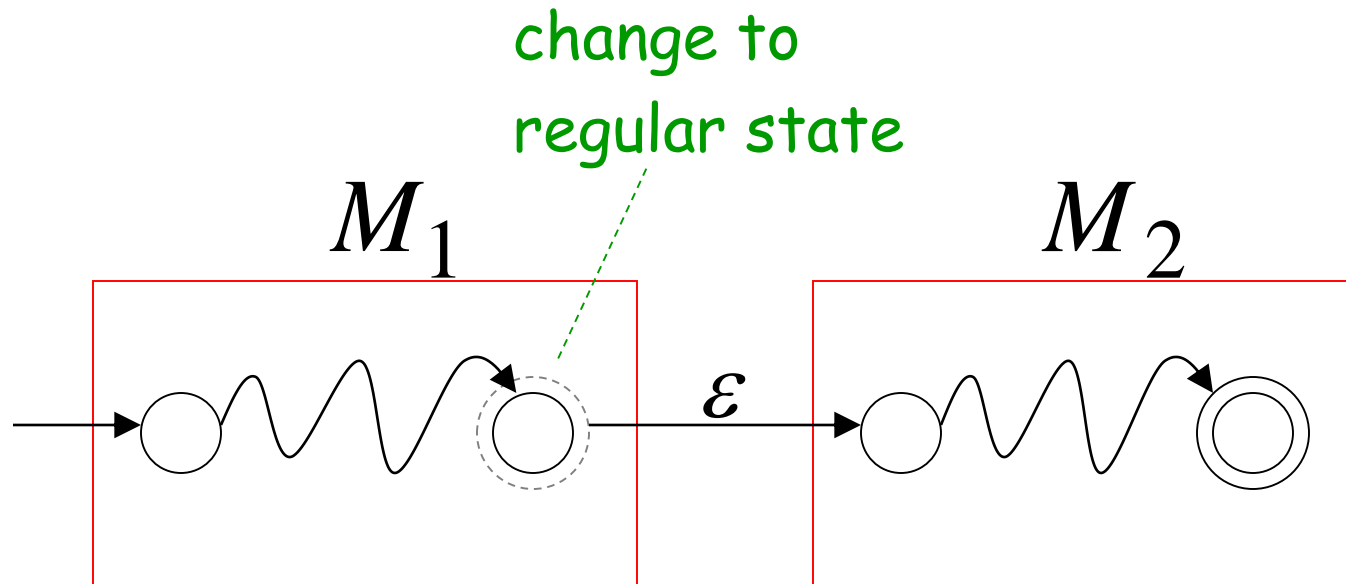


$L_2 = \{ba\}$



Concatenation

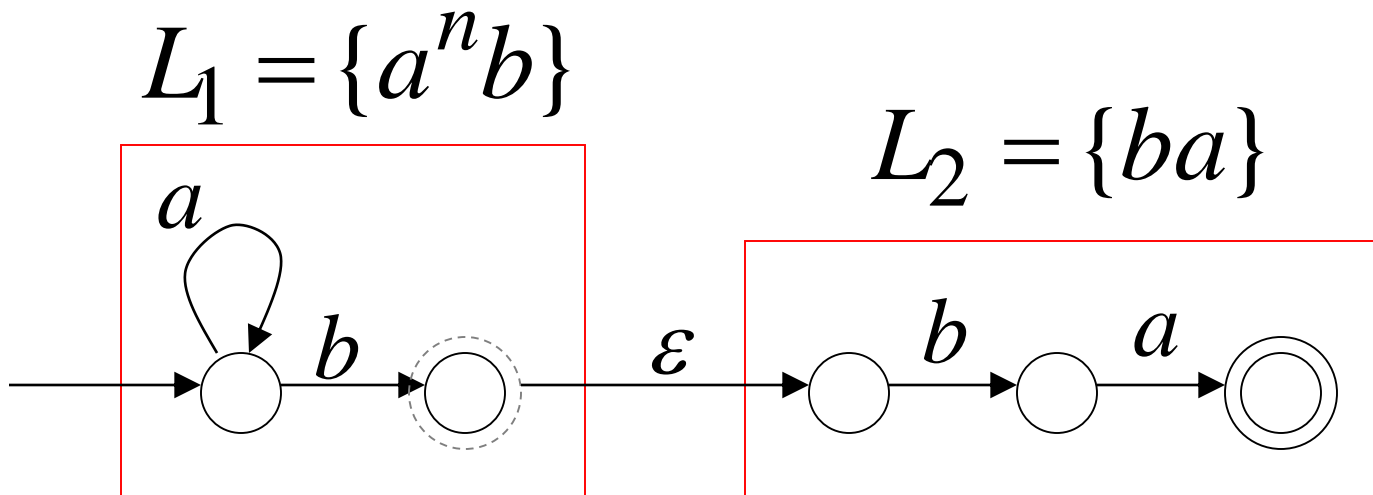
NFA for L_1L_2



$$w \in L_1L_2 \iff w = w_1w_2 : w_1 \in L_1 \text{ and } w_2 \in L_2$$

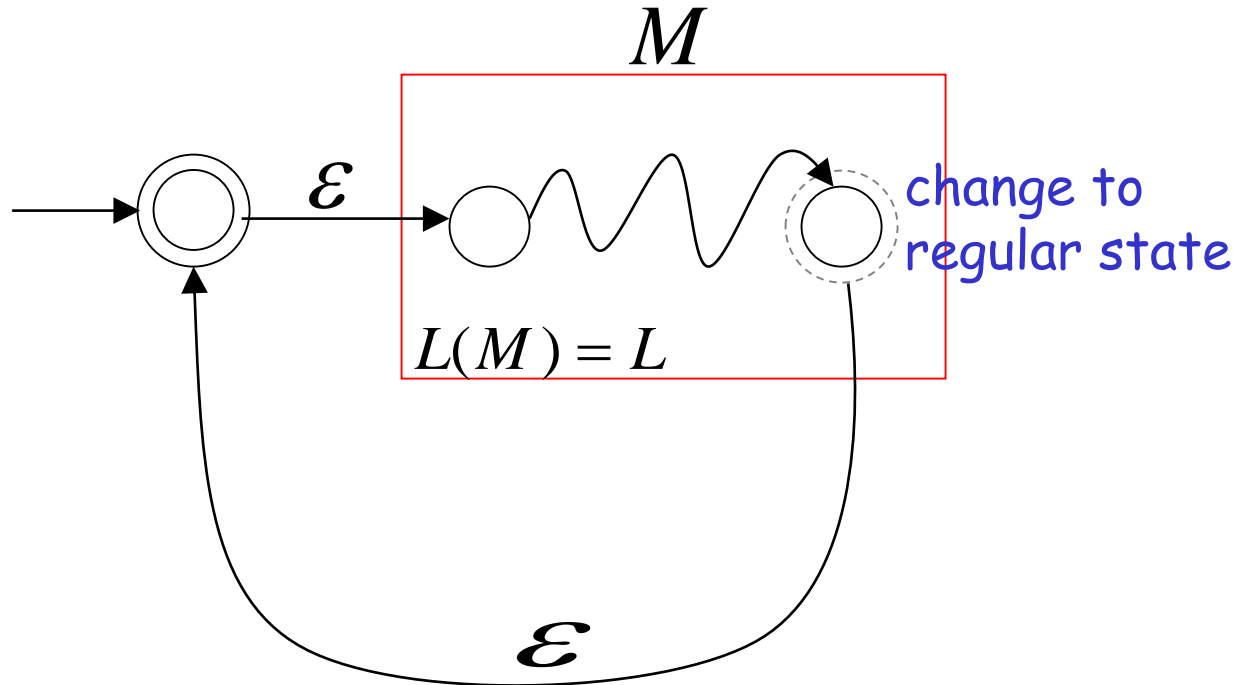
Example

NFA for $L_1L_2 = \{a^n b\} \{ba\} = \{a^n bba\}$



Star Operation

NFA for L^*

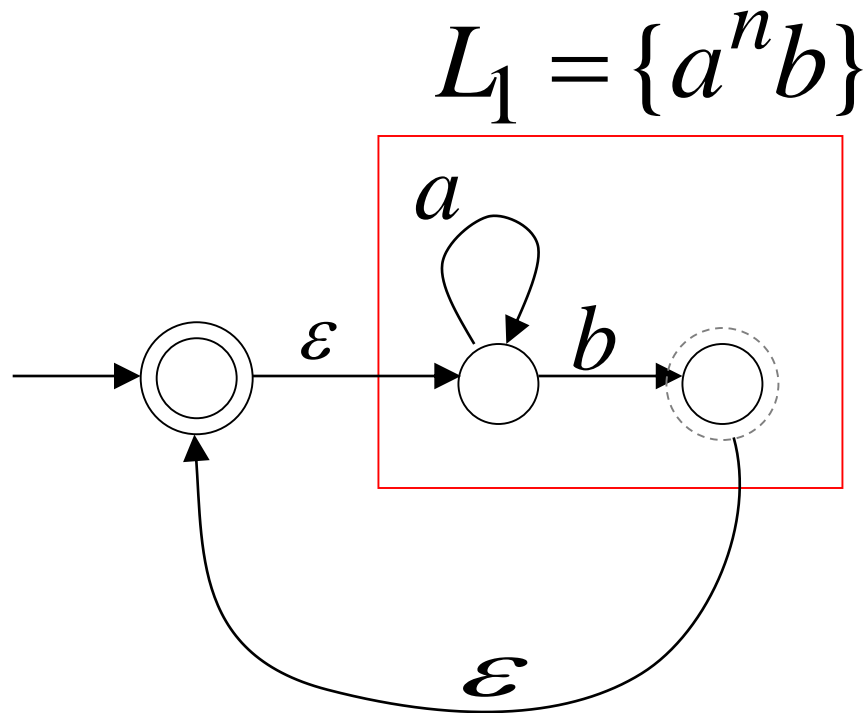


$$w \in L^* \iff w = w_1 w_2 \Lambda w_k : w_i \in L$$

or $w = \epsilon$

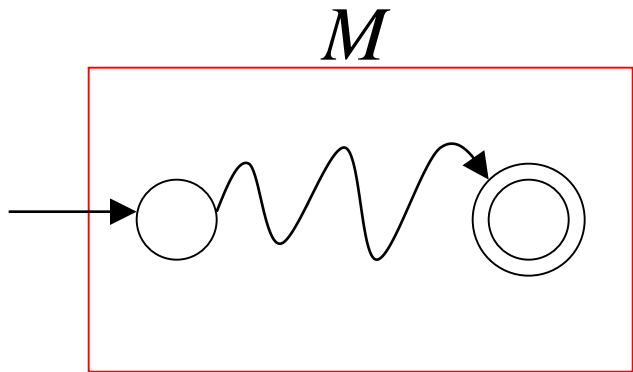
Example

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

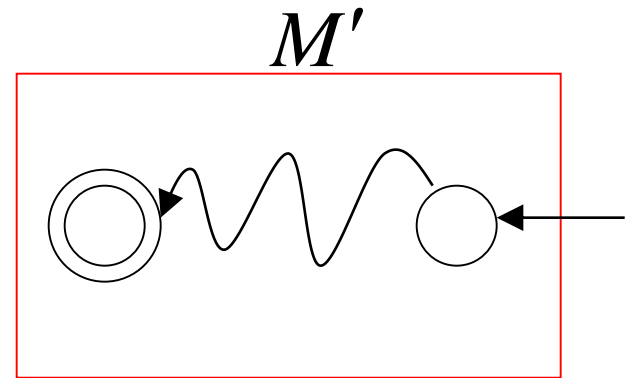


Reverse

NFA for L^R



$$L(M) = L$$

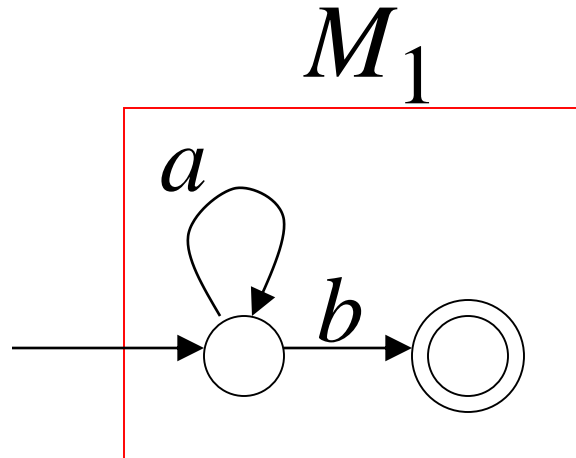


$$L(M') = L^R$$

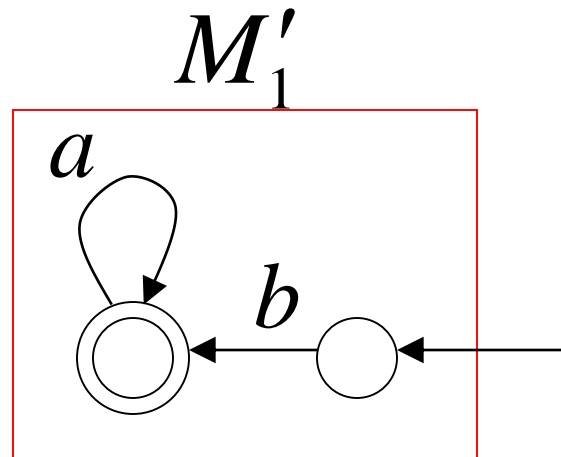
1. Reverse all transitions
2. Make the initial state accept state and the accept state initial state

Example

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

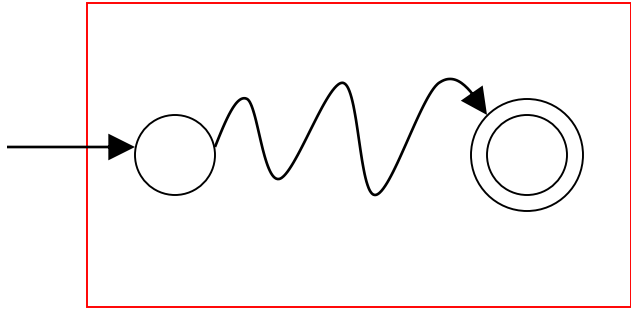


$$L_1^R = \{ba^n\}$$



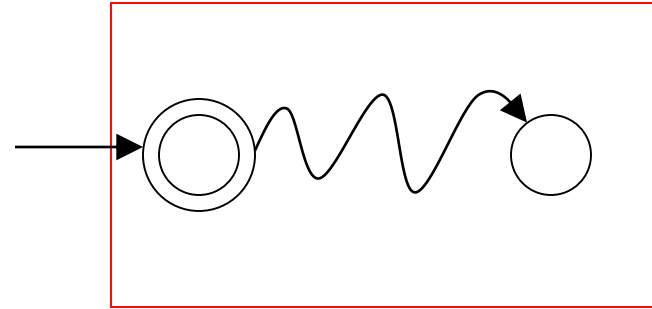
Complement

M



$$L(M) = L$$

M'

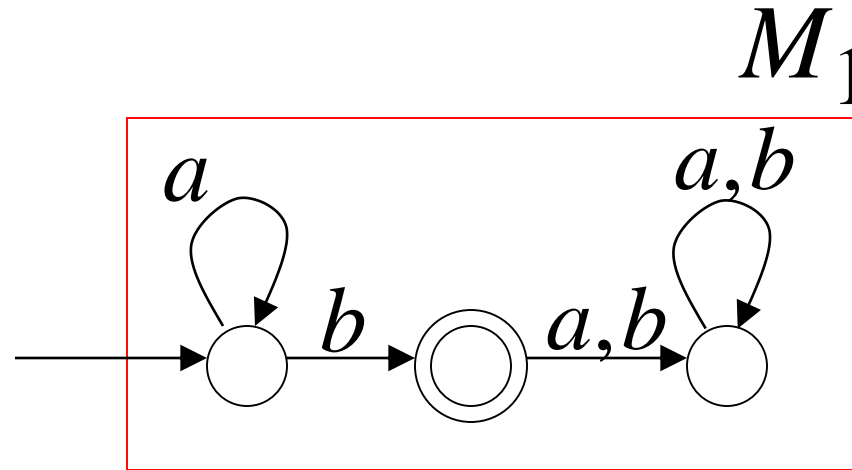


$$L(M') = \bar{L}$$

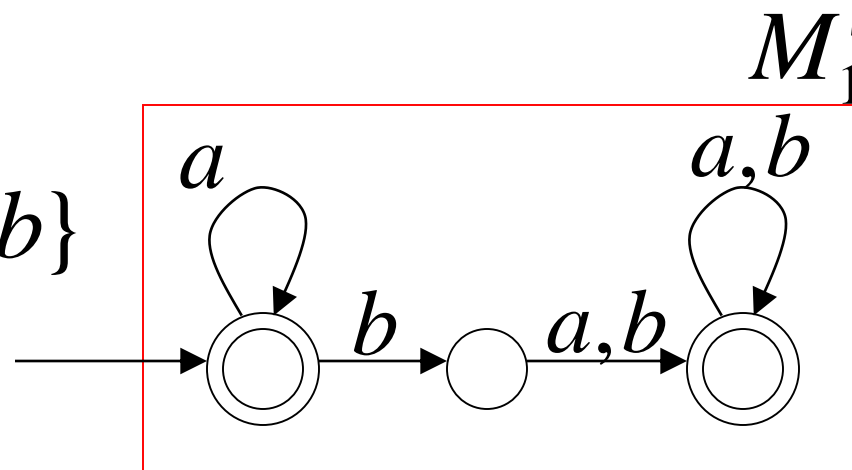
1. Take the **DFA** that accepts L
2. Make accept states regular and vice-versa

Example

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



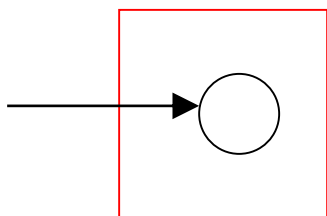
$$\overline{L_1} = \{a,b\}^* - \{a^n b\}$$



NFAs cannot be used for complement

Make accept states regular
and vice-versa

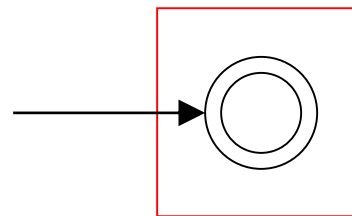
NFA M



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

$$\overline{L(M)} = \Sigma^* = \{a, b\}^*$$

NFA M'



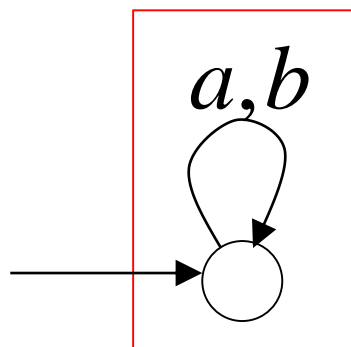
$$L(M') = \{\varepsilon\} \neq \overline{L(M)}$$

it is **not** the
complement

Same example with DFAs

Make accept states regular
and vice-versa

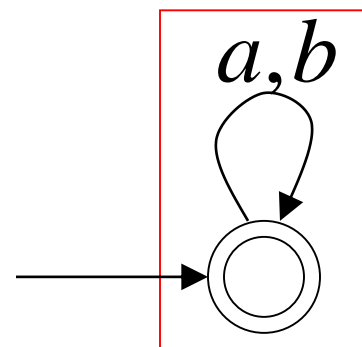
DFA M



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

$$\overline{L(M)} = \Sigma^* = \{a, b\}^*$$

DFA M'

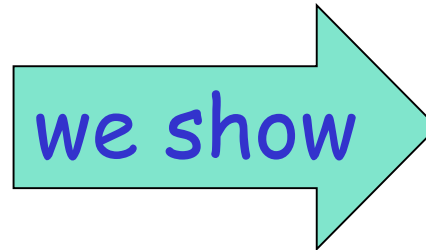


$$L(M') = \{a, b\}^* = \overline{L(M)}$$

it is the
complement

Intersection

L_1 regular



L_2 regular

$L_1 \cap L_2$

regular

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

L_1, L_2 regular, regular

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

$$\begin{array}{l} L_1 = \{a^n b\} \text{ regular} \\ L_2 = \{ab, ba\} \text{ regular} \end{array} \left. \vphantom{\begin{array}{l} L_1 \\ L_2 \end{array}} \right\} \Rightarrow L_1 \cap L_2 = \{ab\} \\ \text{regular}$$

Another Proof for Intersection Closure

Machine M_1

DFA for L_1

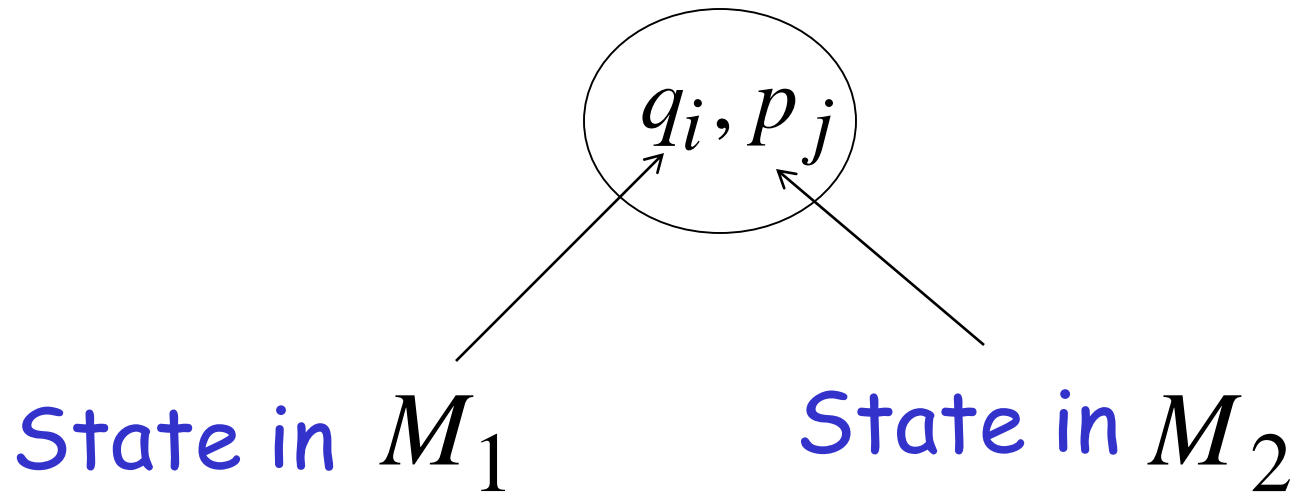
Machine M_2

DFA for L_2

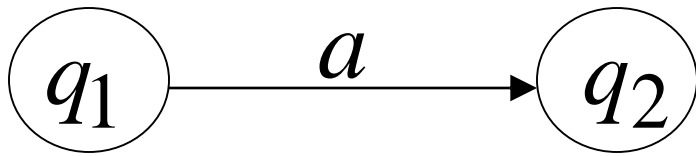
Construct a new DFA M that accepts $L_1 \cap L_2$

M simulates in parallel M_1 and M_2

States in M

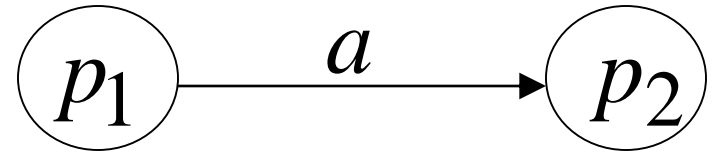


DFA M_1

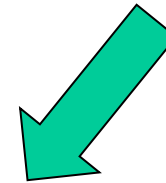


transition

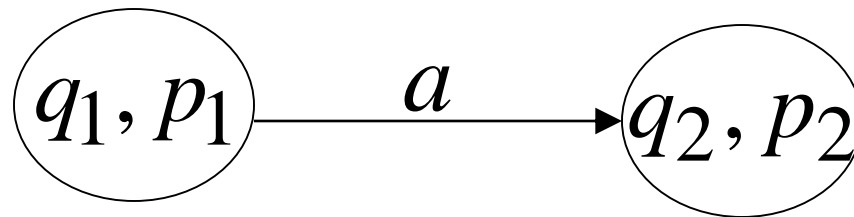
DFA M_2



transition

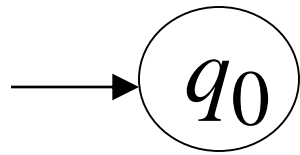


DFA M



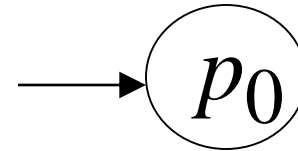
New transition

DFA M_1

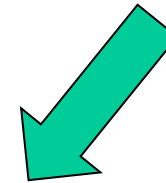


initial state

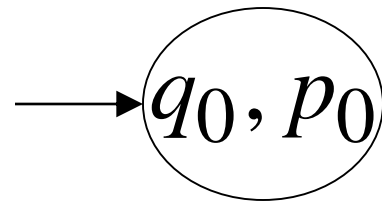
DFA M_2



initial state

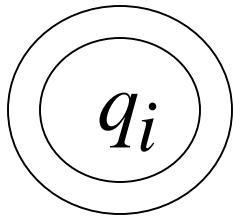


DFA M



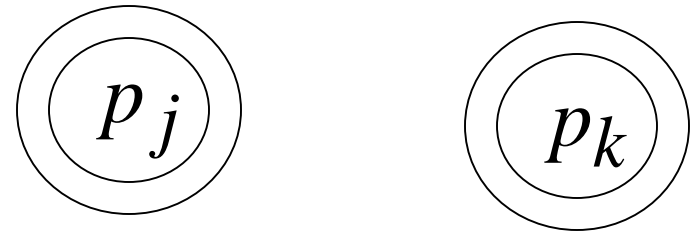
New initial state

DFA M_1



accept state

DFA M_2



accept states



DFA M

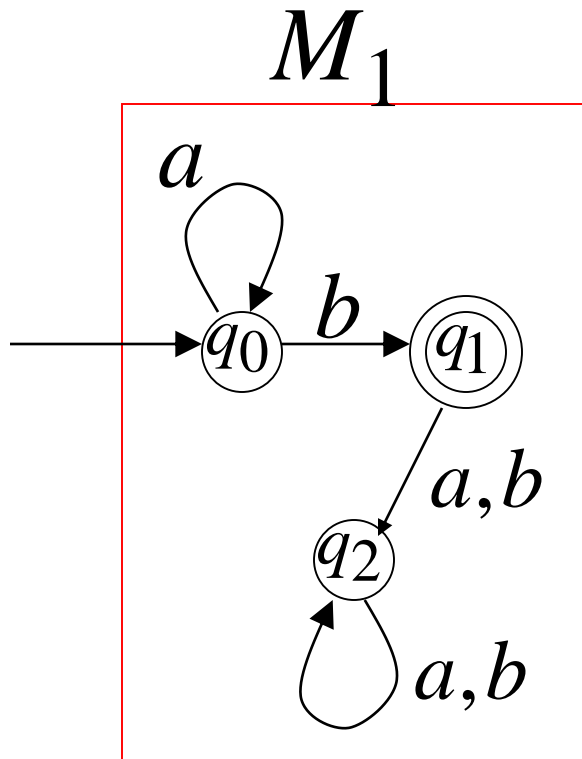


New accept states

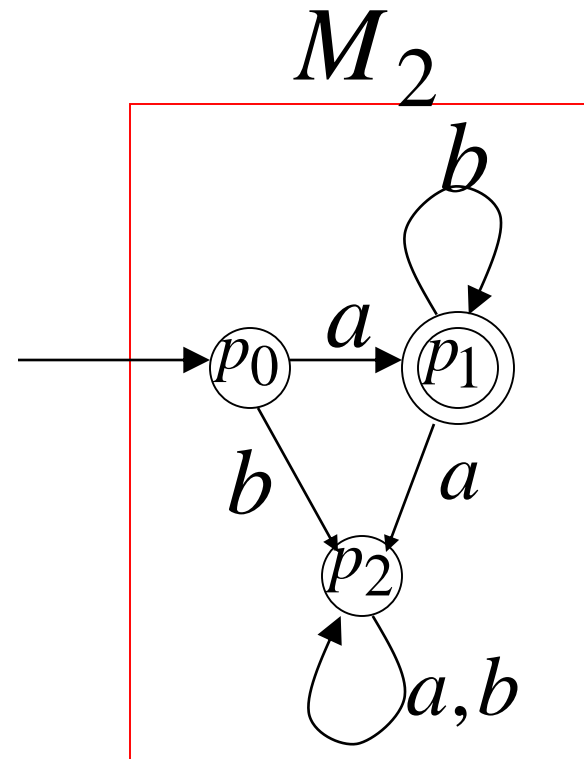
Both constituents must be accepting states

Example:

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

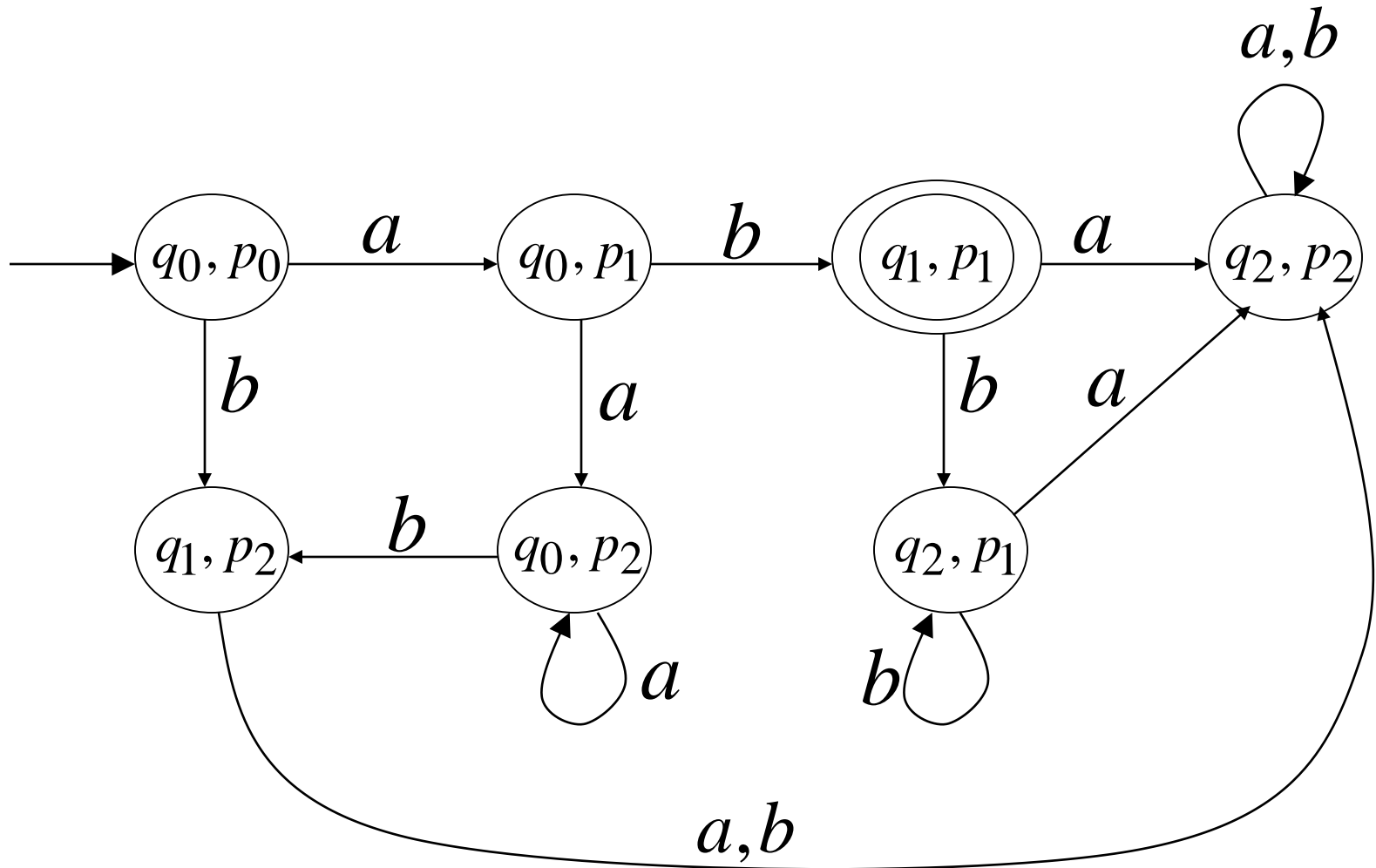


$$L_2 = \{ab^m\} \quad m \geq 0$$



DFA M for intersection

$$L(M) = \{a^n b\} \cap \{ab^m\} = \{ab\}$$

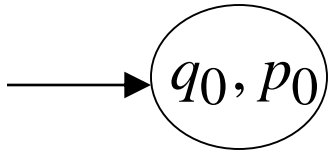


Construction procedure for intersection

1. Build Initial State
2. For each new state and for each symbol add transition to either an existing state or create a new state and point to it
3. Repeat step 2 until no new states are added
4. Designate accept states

Automaton for intersection

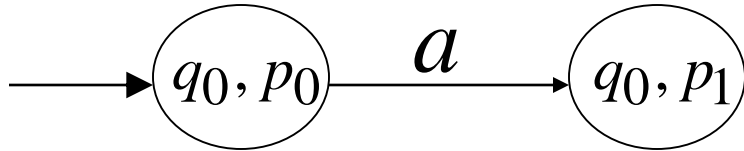
$$L = \{a^n b\} \cap \{ab^m\} = \{ab\}$$



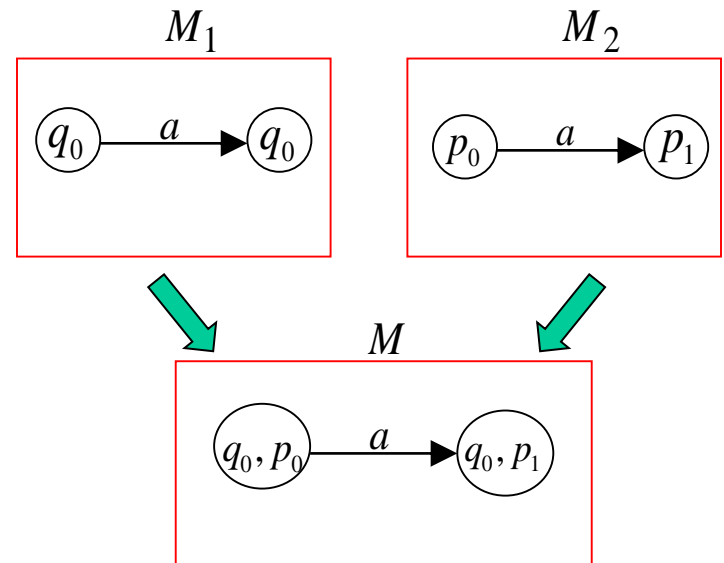
initial state

Automaton for intersection

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

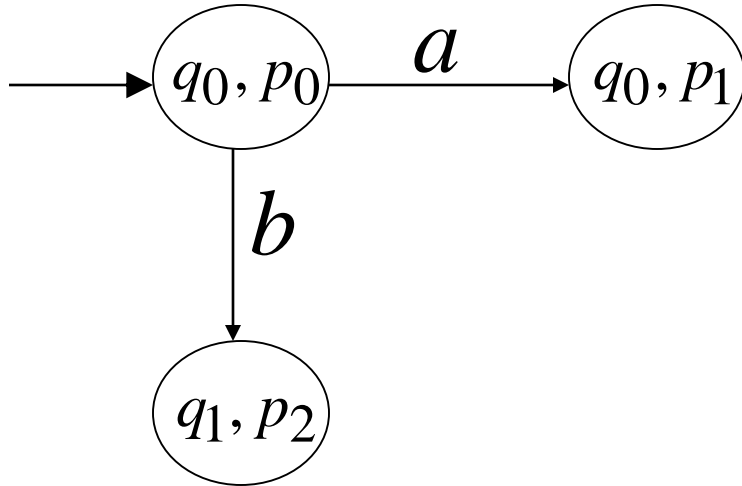


add transition and new state
for symbol a

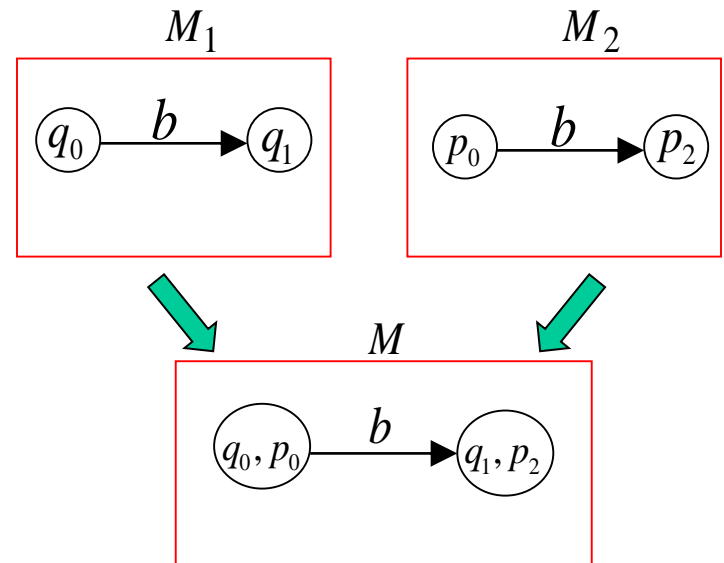


Automaton for intersection

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



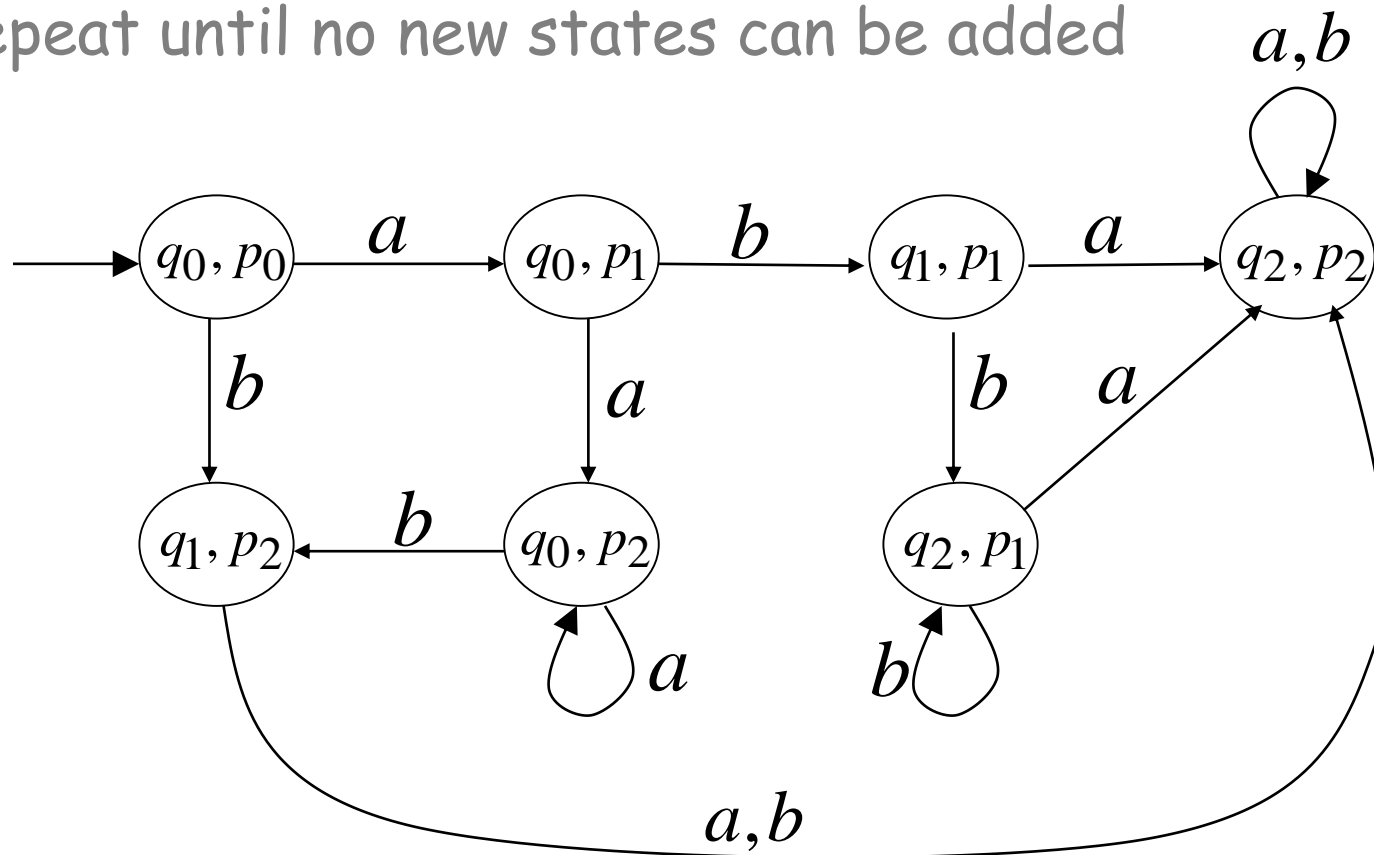
add transition and new state
for symbol b



Automaton for intersection

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

Repeat until no new states can be added

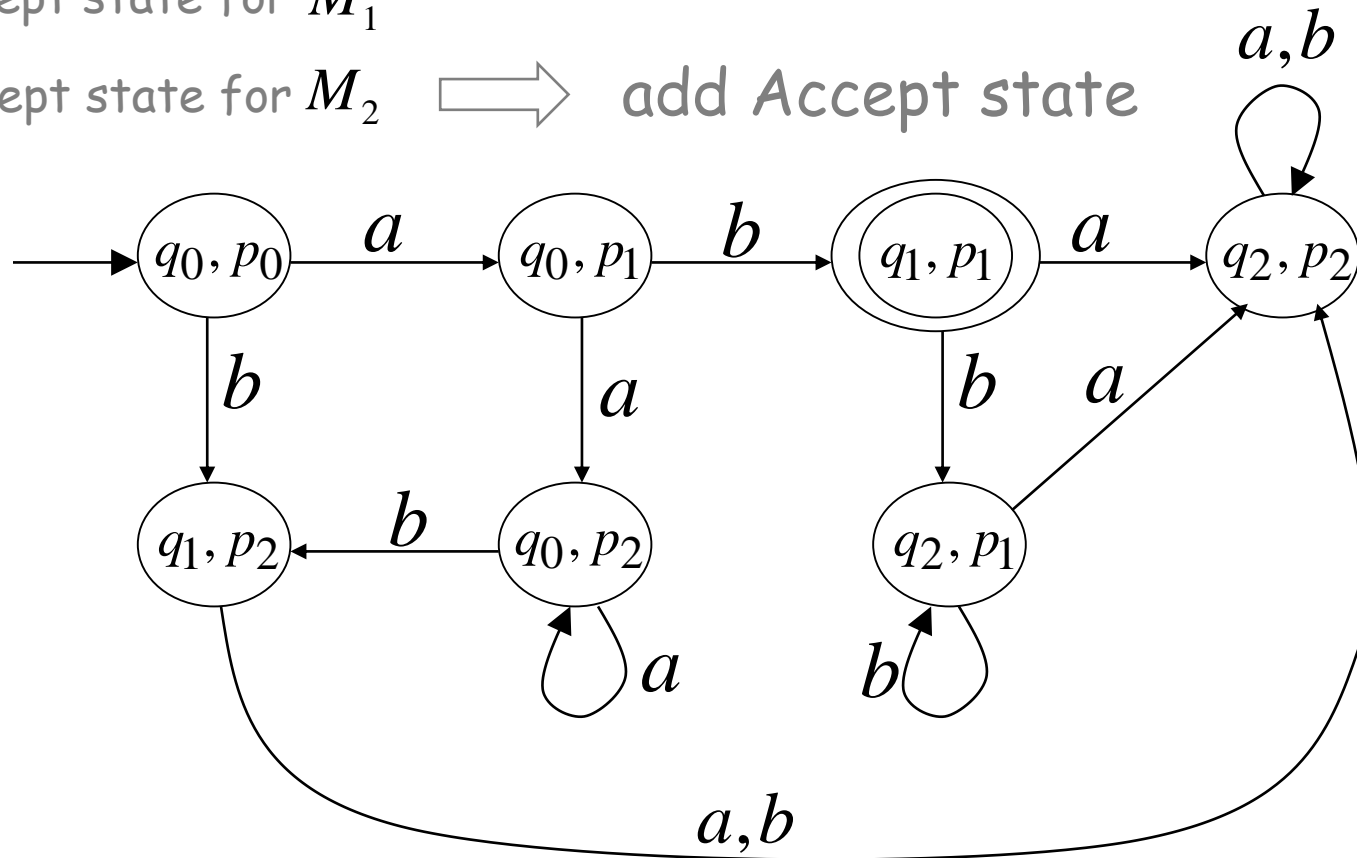


Automaton for intersection

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

q_1 accept state for M_1

p_1 accept state for M_2 \Rightarrow add Accept state



Intersection DFA M :

simulates in parallel M_1 and M_2

accepts string w if and only if:

M_1 accepts string w

and M_2 accepts string w

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