Propiedades de los Lenguajes Regulares

Recordamos...

Un lenguaje L es regular **si** existe un AF M que lo reconoce/acepta. Es decir **L(M) = L**. Esto es que los lenguajes aceptados por AFs forman la familia de los **lenguajes regulares**

Algunos ejemplos:

$\{abba\}$ $\{\lambda, ab, abba\}$

$\{a^n b : n \geq 0\}$ $\{awa : w \in \{a,b\}^*\}$

{todos los strings en {a,b}* con prefijo ab }

{todos los strings binarios sin el substring 001}

$\{x : x \in \{1\}^*$ y x es par$\}$

Pero... vemos que para saber si un lenguaje es regular, debemos construir un AFD que lo acepte

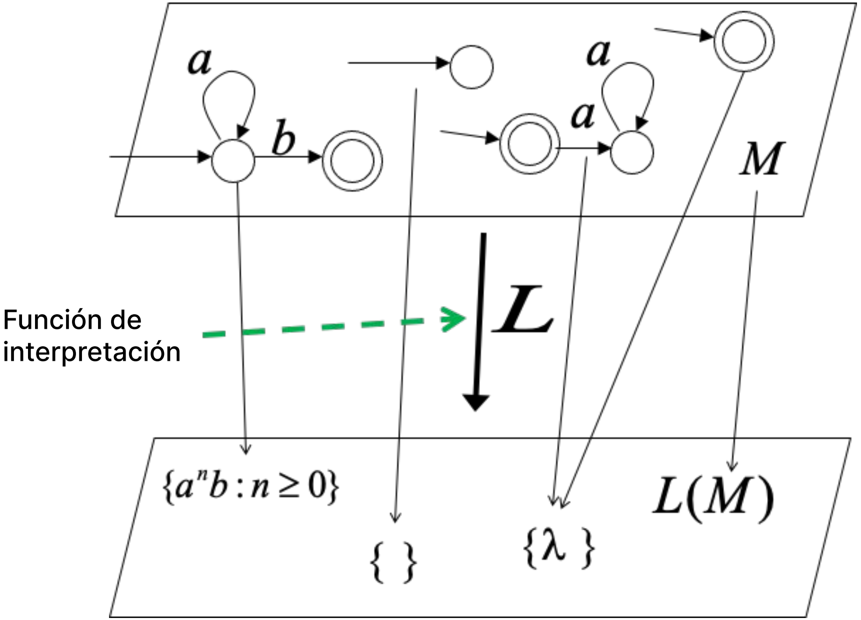
Qué sabemos de los lenguajes regulares ?

Sea L_1 y L_2 dos lenguajes regulares

- **Union** - L_1 U L_2 es un lenguaje regular
- **Intersección** - $L_1 \cap L_2$ es un lenguaje regular
- **Concatenación** - L_1L_2 es un lenguaje regular
- **Clausura de Kleene** - L_1^* es un lenguaje regular
- **Reversa** - L_1^R es un lenguaje regular
- **Complemento** - $L_1 - \Sigma$ es un lenguaje regular

Hoy en día

Sintaxis: Autómatas



Semántica: Lenguajes

Pero qué pasa si no sabemos/queremos hacer el autómata para un lenguaje L ?

Sabemos que si $\Sigma = \{a_1, a_2, ..., a_m\}$

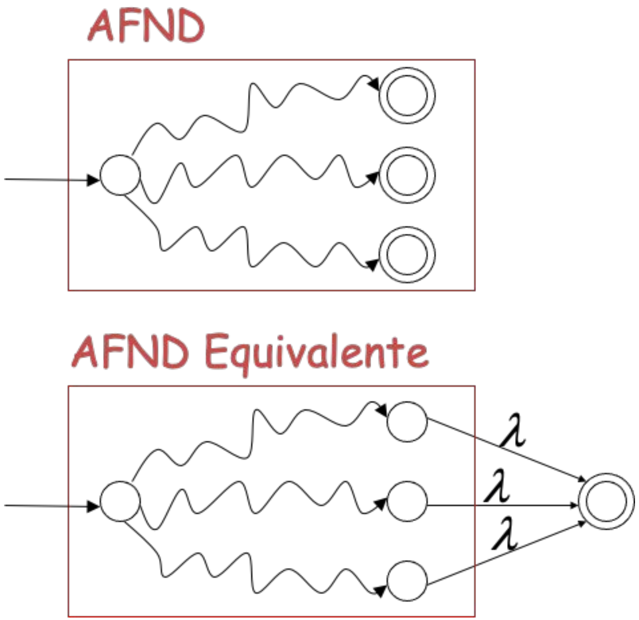
- $\{a_1\}$, $\{a_2\}$, ..., $\{a_n\}$ son lenguajes regulares
- $\varnothing$ es un lenguaje regular
- $\{\lambda\}$ es un lenguaje regular

Vemos que la clase de los lenguajes regulares está cerrada bajo

- Union - L_1 U L_2
- Intersección - $L_1 \cap L_2$
- Concatenación - L_1L_2
- Clausura de Kleene - L_1^*
- Reversa - L_1^R
- Complemento - $L_1 - \Sigma$

Mostramos cómo

Una transformación útil (usar un único estado final)



Lenguaje regular L_1

Lenguaje regular L_2

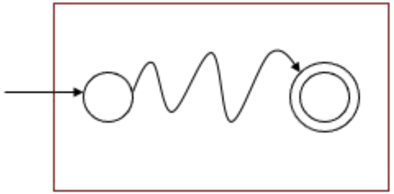
$$L(M_1) = L_1$$

$$L(M_2) = L_2$$

AFND M_1

Único estado final

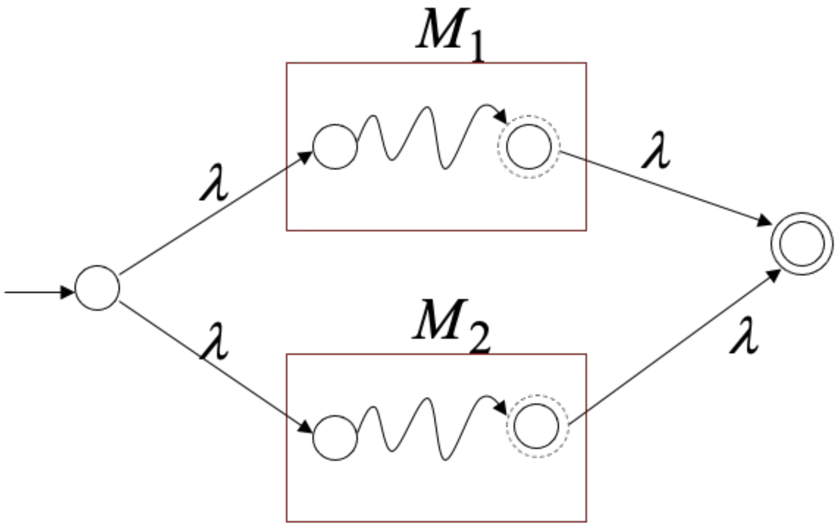
AFND M_2



Único estado final

Unión

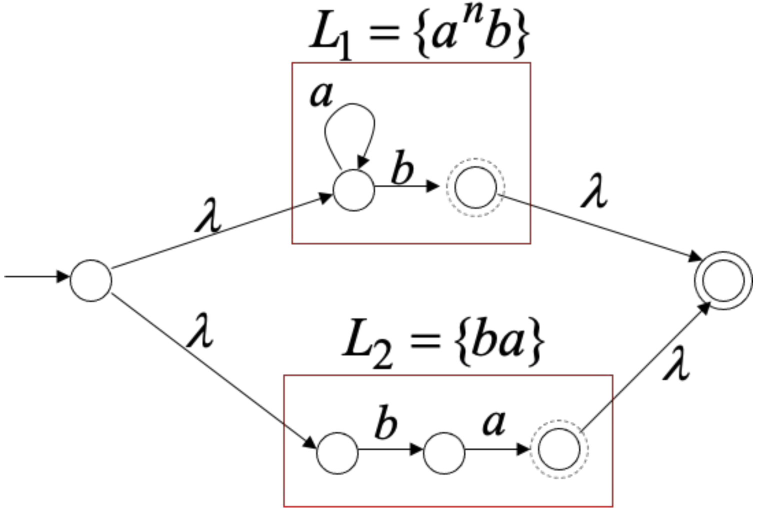
$L_1 \cup L_2$



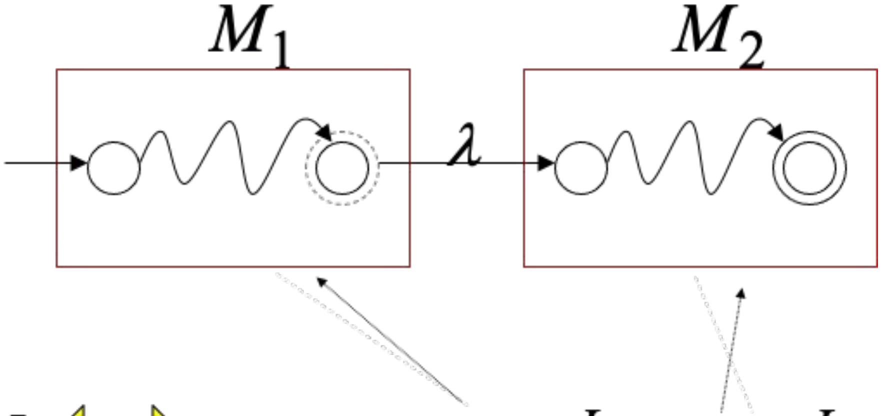
$w \in L_1 \cup L_2 \Leftrightarrow w \in L_1 \vee w \in L_2$

Unión (ejemplo)

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



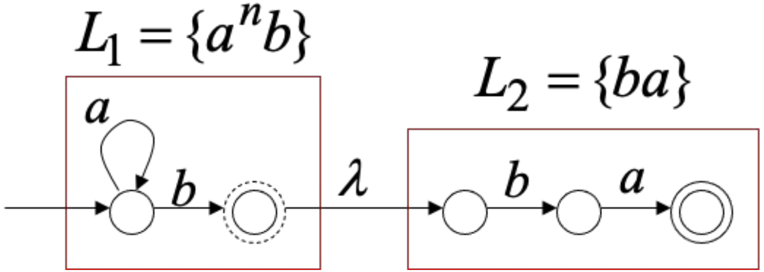
Concatenación



$$w \in L_1 L_2 \Longleftrightarrow w = w_1 w_2 : \; w_1 \in L_1 \;\wedge\; w_2 \in L_2$$

Concatenación (ejemplo)

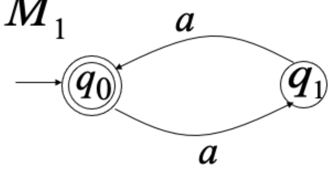
$$L_1 L_2 = \{a^n b\}\{ba\} = \{a^n bba\}$$



Intentamos construir un AFD para $\{a^{2i+3j} \mid i \geq 0, j \geq 0\}$

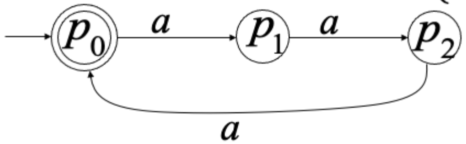
1) Lo descomponemos en problemillas más sencillos

M_1



$\{a^{2i} \mid i \geq 0\}$

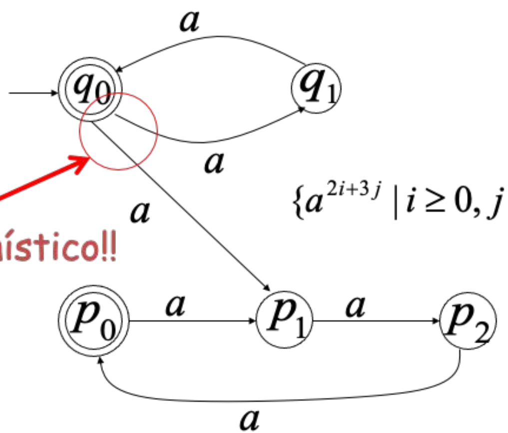
M_2



$\{a^{3j} \mid j \geq 0\}$

2) Los combinamos

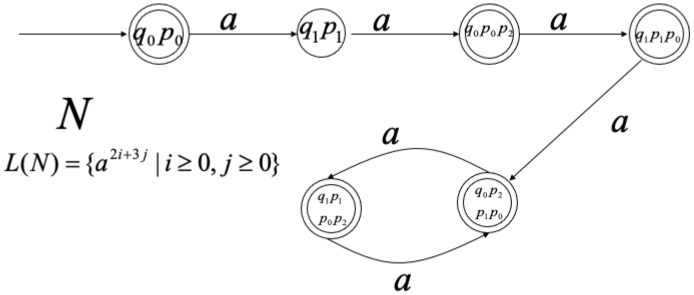
M_{12}



$\{a^{2i+3j} \mid i \geq 0, j \geq 0\}$

No determinístico!!

3) Lo convertimos en un AFD

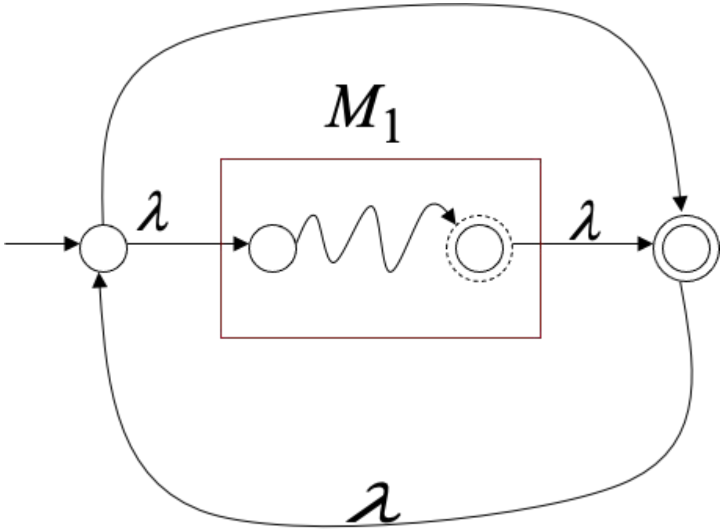


N

$L(N) = \{a^{2i+3j} \mid i \geq 0, j \geq 0\}$

Clausura de Kleene

L_1^*



$w = w_1 w_2 \cdots w_k$

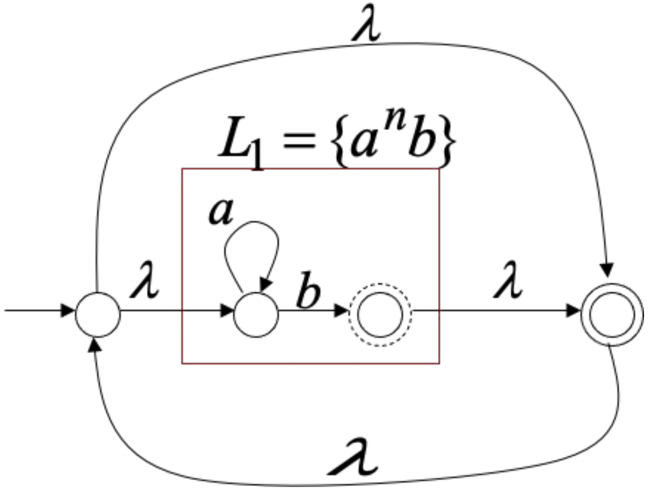
$w_i \in L_1$

$L_1 = L(M_1)$

$\lambda \in L_1^*$

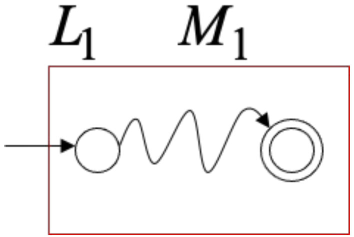
Clausura de Kleene (ejemplo)

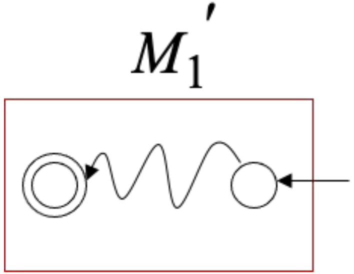
$$L_1^{\ *} = \{a^n b\}^*$$



Reversa

L_1^R



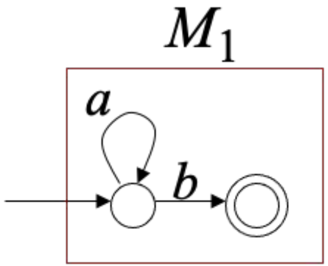


Donde M_1'

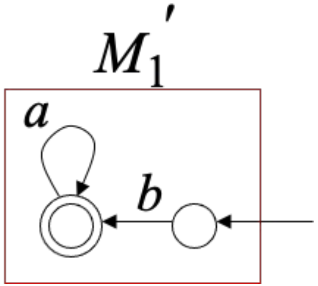
- **Tiene todas las transiciones revertidas y**
- **el estado inicial es estado final y viceversa**

Reversa (ejemplo)

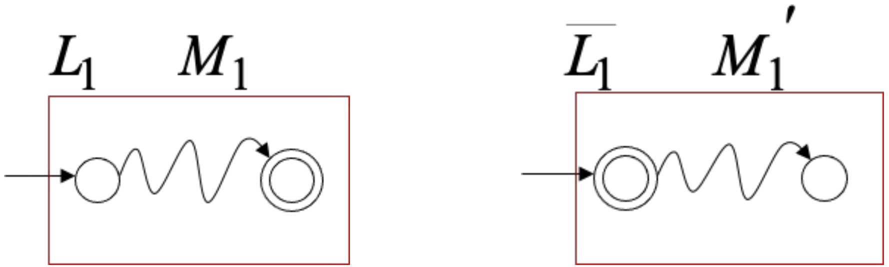
$$L_1 = \{a^n b\}$$



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



Complemento

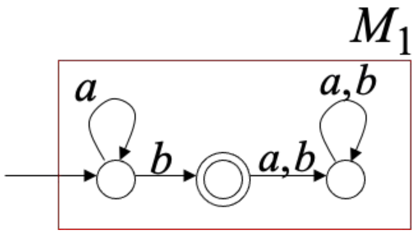


Donde M_1'

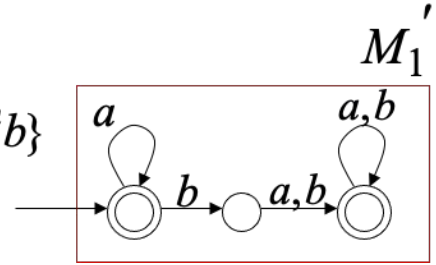
- **Tiene los estados finales como no finales y viceversa**

Complemento (ejemplo)

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



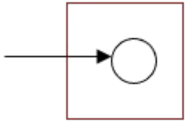
$\overline{L_1} = \{a,b\}^* - \{a^n b\}$



Complemento - Observación

Los AFNDs no pueden usarse para el complemento,

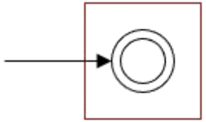
AFND M



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

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

AFND M'



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

No es el complemento

Intersección

Leyes de DeMorgan $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$

$L_1 \, , \; L_2$ regular

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

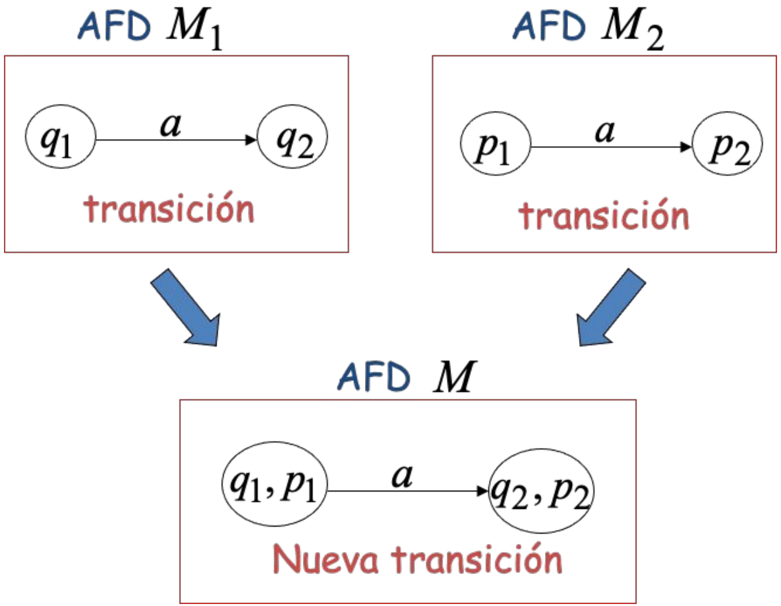
$\Rightarrow \overline{L_1} \cup \overline{L_2}$ regular

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

$\Rightarrow L_1 \cap L_2$ regular

Intersección

Necesitamos simular los dos autómatas



Intersección

Seguimos la siguiente idea

Intersección - Procedimiento

1. Crear el estado inicial

2. Para cada nuevo estado y para cada símbolo, agregar una transición a un estado existente o bien crear un nuevo estado y unirlo a él

3. Repetir el paso 2 hasta que no se agregue ningún nuevo estado

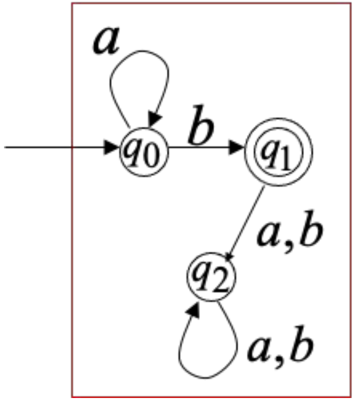
4. Definir los estados finales

Intersección (ejemplo)

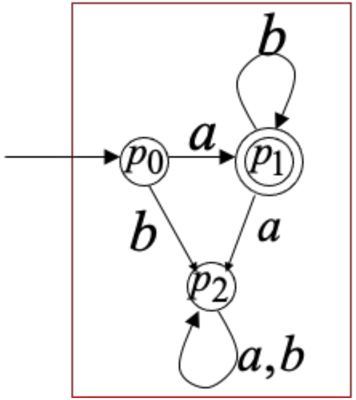
$$L_1 = \{a^n b : n \geq 0\}$$

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

M_1



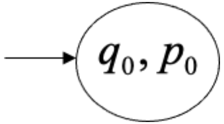
M_2



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

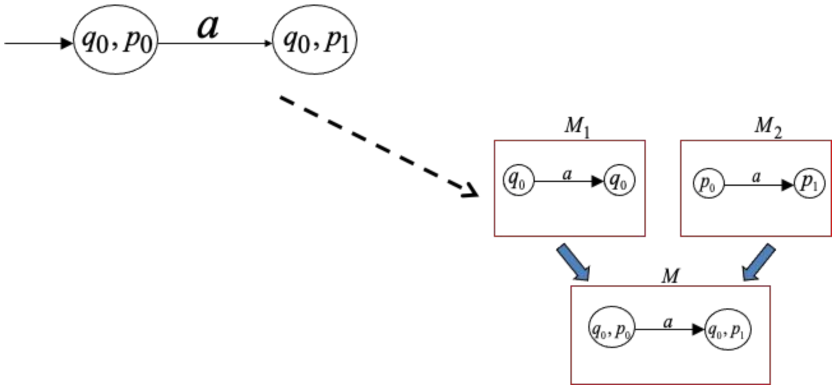
Intersección (ejemplo)

Definimos el estado inicial



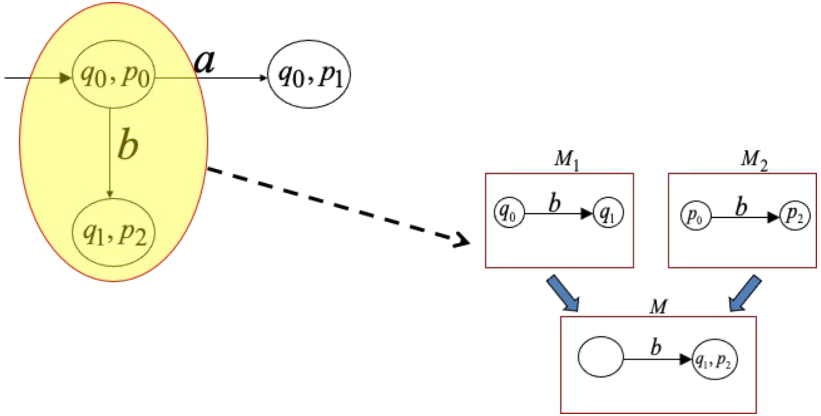
Intersección (ejemplo)

Añadimos una transición y un nuevo estado para el símbolo a



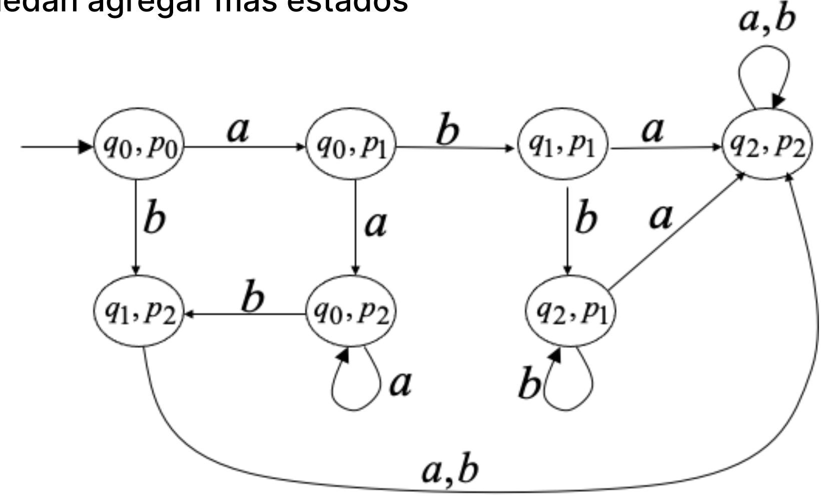
Intersección (ejemplo)

Añadimos una transición y un nuevo estado para el símbolo b



Intersección (ejemplo)

Repetimos hasta que no se puedan agregar más estados



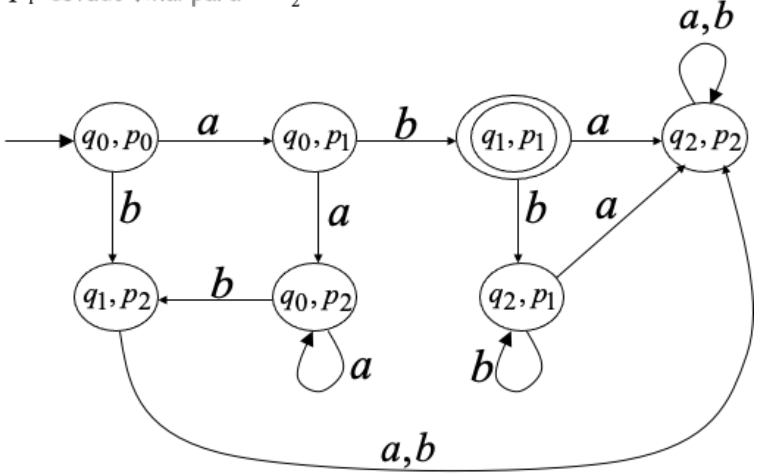
Intersección (ejemplo)

Agregamos los estados finales

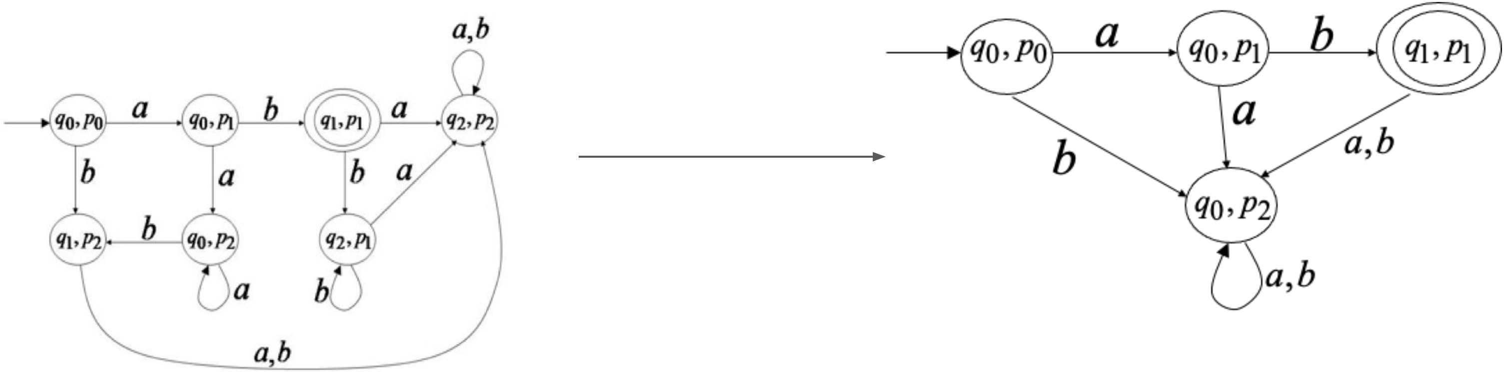
q_1 estado final para M_1

p_1 estado final para M_2

$\Rightarrow$ añadir estado final



Y aplicamos minimización



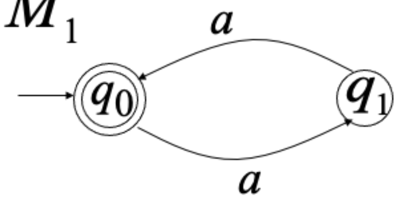
En resumen,

- M simula en paralelo M_1 y M_2
- M acepta el string w si y sólo si M_1 acepta w y M_2 también

Ejemplo

Sean $L_1 = \{a^{2i} \mid i \geq 0\}$ $L_2 = \{a^{3j} \mid j \geq 0\}$ calculamos $L_1 \cap L_2$

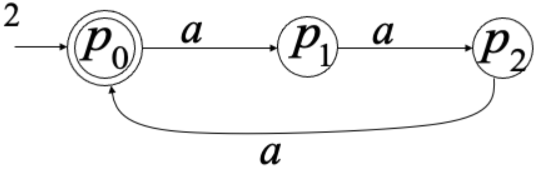
M_1



$L_1 = \{a^{2i} \mid i \geq 0\}$

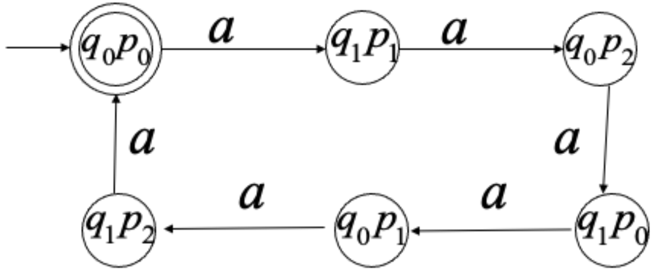
M_2

$L_2 = \{a^{3j} \mid j \geq 0\}$



Q'	q_0p_0	q_0p_1	q_0p_2	q_1p_0	q_1p_1	q_1p_2
$\delta'(q,a)$	q_1p_1	q_1p_2	q_1p_0	q_0p_1	q_0p_2	q_0p_0

M_{12}



$$L(M_{12}) = L(M_1) \cap L(M_2)$$