Conjunto finito y No Finito

Paul Grimaldo

$A \subseteq B$ — $|A| \leq |B|$

$A \subset B$ — $|A| < |B|$

$|A| = 0$ — $A = \emptyset$

Equivalencia de Conjuntos

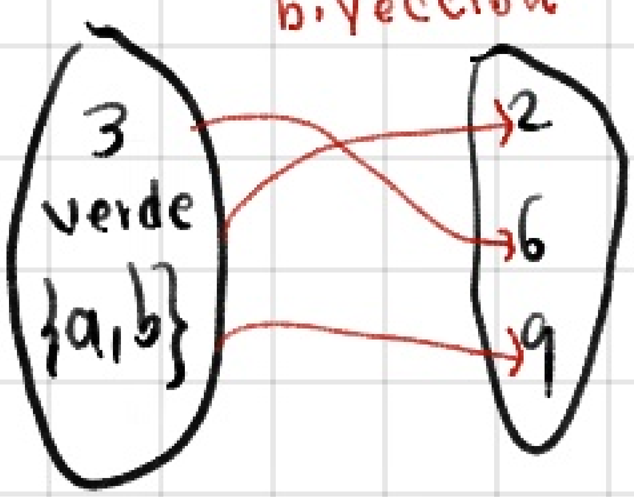
A y B son equivalentes si existe una biyección $f: A \to B$

↓ tiene que existir al menos UNA biyección

$A = \{3, \text{verde}, \{a, b\}\}$

$B = \{2, 6, 9\}$

biyección



A B

$f(3) = 6$

$f(\text{verde}) = 2$ $\Rightarrow$ A y B son equivalente

$f(\{a,b\}) = 9$

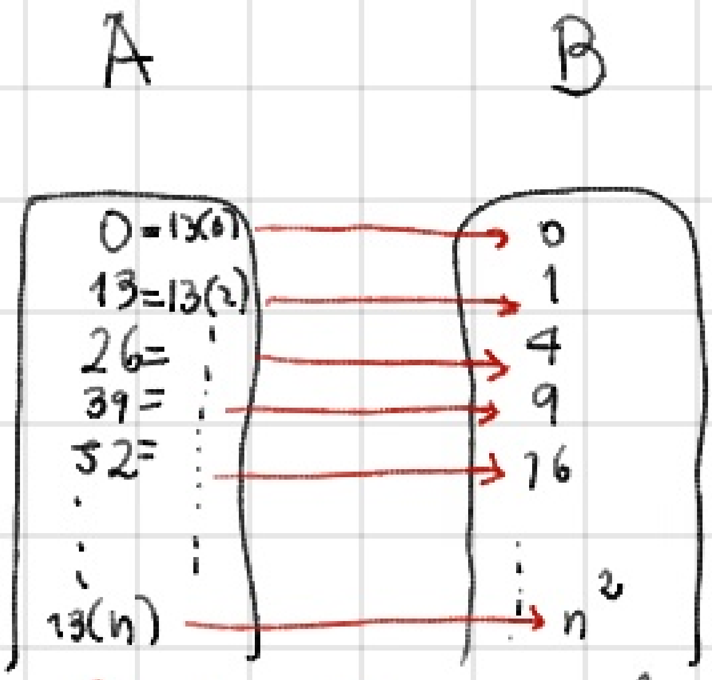
// 2 conjuntos serán equivalentes si tienen el mismo número de elementos.

// inyectiva { elementos en el dominio tiene imagenes diferentes

// biyectividad { cada elemento del codominio corresponde a un elemento en el dominio }

A múltiplo de 13

B cuadrado perfecto.



$f: A \to B$

$f(13) = n^2$

f biyectiva

$\therefore$ A y B son equivalentes

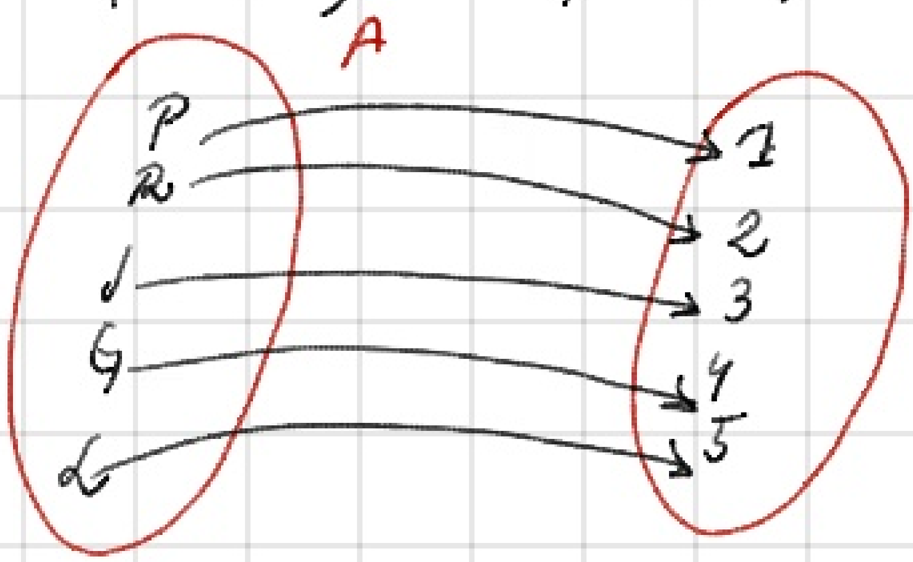
Primer Resultado de Teoría de la computación

Conj. finito: Un conjunto es finito si es equivalente con el conjunto $I = \{1, 2, 3, 4, \ldots, n\}$ para alguna $n \in \mathbb{N}$

E

A = { Paul, Ronald, José, German, Leonel }

I = { 1, 2, 3, 4, 5 }



$f(P) = 1$

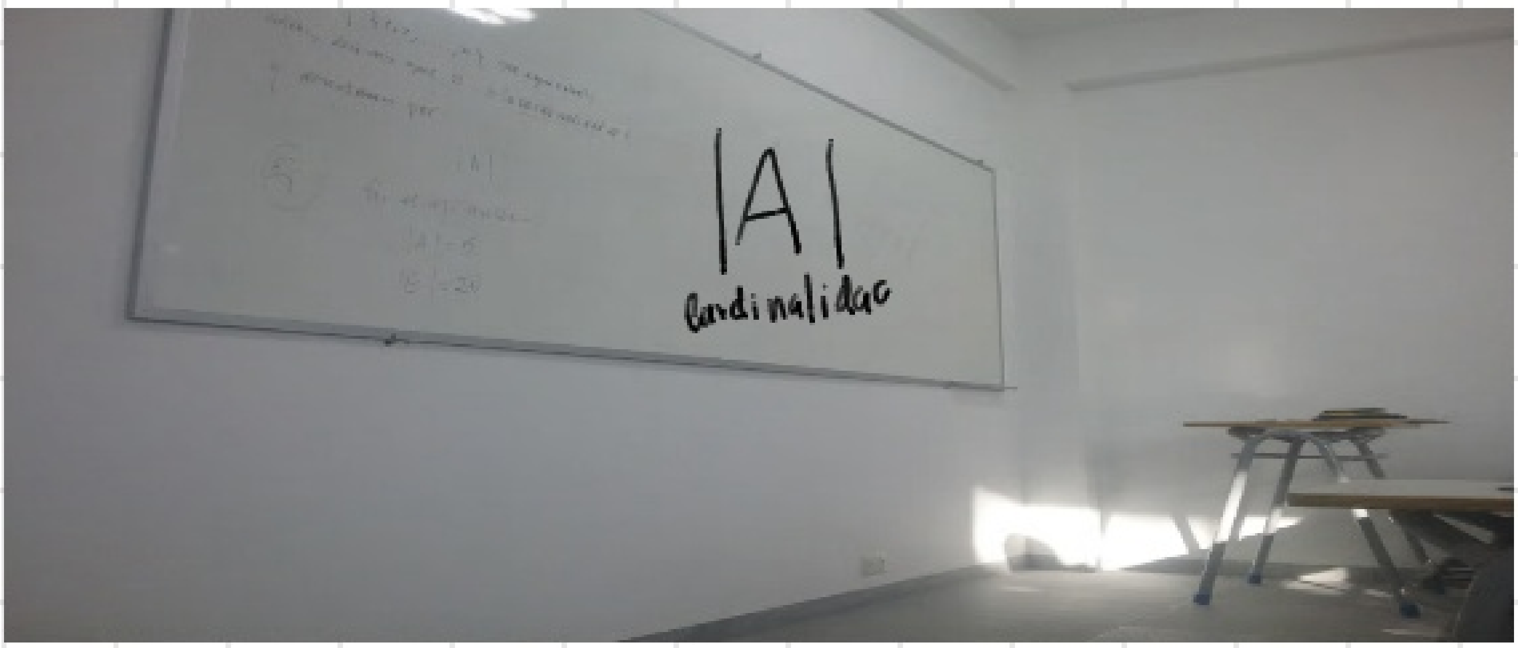
$\vdots$

$f(L) = 5$

Cardinalidad

Si A y $\{1,3,4,5,\ldots,n\}$ son equivalentes entonces n es la cardinalidad de A.

$$|A| = n$$



Conjunto infinito: es infinito si no es finito

Ⓔ $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{R}$, $\mathbb{Q}$

por la experiencia de los multiplos de 13 y cuadrados perfectos se podría pensar que todos los conjuntos infinitos son equivalentes, pero no todos los conjuntos infinitos son equivalentes.

Conjunto Contablemente infinito, un conjunto es CCI si es equivalente con $\mathbb{N}$.

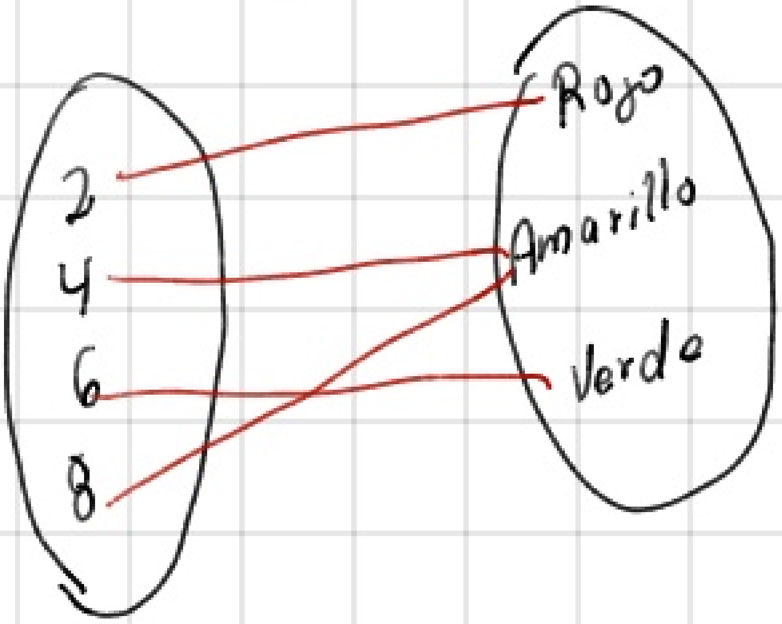
Conjunto Contable, es CC si es finito o CCI

Conjunto incontable es CI si no es CC.

$A = \{2, 4, 6, 8\}$

$B = \{\text{Rojo}, \text{Amarillo}, \text{verde}\}$

Construir una función $f: A \to B$ tal que f sea inyectiva.



No existe una función inyectiva de A en B.

Principio de la Cansillas

Si A y B son conjuntos finitos no vacios y $|A| > |B| \to$ no existe una función inyectiva de A en B

$A=\{2,4\}$

$P(A)=$ Conjunto Potencia de $\mathbb{N}$ (Contiene a todos los subconjuntos de $\mathbb{N}$)

$2^{A}=\{\emptyset,\{2,4\},\{2\},\{4\}\}$

$|A|=2$

$|2^{A}|=4$

$B=\{a,b,c\}$

$2^{B}=0,\{a,b,c\},\{a,\},\{b\},\{c\},\{a,b\},\{a,c\},\{c,b\}\}$

$|b|=3$

$N=\{0,1,2,3,4,5......\}$

conjunto potencia de N

$2^{N}=\{0,\{0\},\{1\},\{2\},\{3\},\{4\},\{5\},.....,\{0,1\},\{0,2\},\{0,3\},\{0,4\},......\}$

Teorema:

El conjunto 2eN es incontable.

PRUEBA:

Supongamos que 2^N es contable
luego 2^N es contablemente infinito
entonces 2^N es equivalente a $\mathbb{N}$
es decir existe $f: N \to 2^N$
tal que $f(i) = S_i$
para algun $i \in N$ y f biyectiva

Construimos D tal que
$D = \{ n \in \mathbb{N} / n \notin S_n \}$

D es subconjunto de $\mathbb{N}$
$\hookrightarrow$ Todos los subconjuntos de $\mathbb{N}$ están en $2^{\mathbb{N}}$
Por lo tanto
D está en 2^N {conjunto potencia}

luego $D = S_K$ para algun $K \in \mathbb{N}$

¿ $K \in S_K$?

Si $\Rightarrow$ $K \in S_K \Rightarrow K \in D \Rightarrow K \notin S_K$

No $\Rightarrow$ $K \notin S_K \Rightarrow K \in D \Rightarrow K \in S_K$

Contradicción

Por tanto
2^N es incontable.

Alfabeto

Un alfabeto es un conjunto finito no vacio.

Ex.

$$\Sigma_1 = \{a, b\}, \Sigma_2 = \{1, 0\}, \Sigma_3 = \{rojo, amarillo, verde\}$$

$$\Sigma_4 = \{+, -, *, /\}$$

$$\Sigma_5 = \{!, (,), ?\}, \Sigma_6 = \{ab, da, ce\}$$

Palabra

Una palabra sobre sigman sucesion finita de simbolos de sigma (alfabeto)

-Es decir: $(\sigma_1, \sigma_2, \sigma_3, \sigma_n)$

para $\sigma_1, \sigma_2, \sigma_3 \in \Sigma$

Tomando por ejemplo el conjunto E3={rojo,amarillo,verde} una palabra segun la definicion vendria a ser:

rojorojorojo.
rojoamarilloverde.
rojo.
amarilloverde.

Cada ejemplo es una sucesion de sigma para cada sigma que pertenece en este ejemplo a E3.

Palabra Vacia

Es la sucesion vacia de simbolos de sigma y se denota mediante Lambda $= \lambda$

Longitud de una palabra

si $\omega = \sigma_1\sigma_2\sigma_3\cdots\sigma_n$; $\sigma_i \in \Sigma$

n es la longitud de la palabra ω y se denota por

$$|\omega| = n$$

nota:

La única palabra de longitud 0 es $|\lambda| = \emptyset$

Notaciones:

Σ^* = Conjunto de todas las palabras sobre sigma Σ

Σ^+ = Conjunto de todas las palabras sobre Σ excepto λ

$\Sigma^k = \{ \omega \in \Sigma^* / |\omega| = k \}$

$\Sigma^* = \Sigma^+ \cup \{\lambda\}$

$\Sigma^+ = \Sigma^* - \{\lambda\}$

Ex.

$\Sigma^1 = \{ \omega \in \Sigma^* / |\omega| = 1 \} = \Sigma$

$\Sigma_0 = \{ \omega \in \Sigma^* / |\omega| = \emptyset \} = \lambda$

$\Sigma = \{a, b\}$

$\Sigma^* = \{\lambda, a, b, aa, ab, ba, bb, aaa$
$aab, abb, bbb, bab, baa \ldots\}$

$\Sigma^+ = \{a, b, aa, ab, ba, bb, aaa$
$aab, abb, bbb, bab, baa \ldots\}$

$\Sigma = \{aa, ab, ba, bb\}$

$\Sigma^0 = \{\lambda\}$

$\Sigma^1 = \{a, b\} = \Sigma$

$A_3 = \{aab, aba, baa\}$
$|A_3| = 3$
$P(3;2;1) = \dfrac{3!}{2!\,1!}$

$|A|_4 = P(4,2,2) = \dfrac{4!}{2!\,2!}$

Concatenación

Sea $u = \sigma_1 \sigma_2 \sigma_3 \ldots \ldots \sigma_n \in \Sigma$
$v = \tau_1 \tau_2 \tau_3 \ldots \ldots \tau \in \Sigma$

Se define la concatenación de u y v
$uv = \sigma_1 \sigma_2 \sigma_3 \ldots \sigma_n \tau_1 \tau_2 \tau_3 \ldots \tau_n \in \Sigma$

Propiedades

1) $|uv| = |u| + |v| = n+m$
2) $|uv|_\sigma = |u|_\sigma + |v|_\sigma$
3) $uv \neq vu$
4) $u\lambda = \lambda u$
5) $(uv)w = u(vw)$

Construir:

a) u tal que $|u| = 5 \wedge |u|_b = 2$ | aaabb | bebes
b) v " " $|v| = 4 \wedge |v|_b = 2$ | abab | bebe
c) $|uv| = |u| + |V| = 5+4 = 9$
d) $|vv|_b = |u|_b + |V|_b = 2+2 = 4$

Principio de Inducción para $\vec{\Sigma}^*$

Sea L un conjunto de palabras sobre sigma con las propiedades:

1) $\lambda \in L$
2) $\omega \in L \wedge a \in \vec{\Sigma} \Rightarrow \omega a \in L$

Entonces $L = \vec{\Sigma}^*$ (es decir todas las palabras de $\vec{\Sigma}^*$ están en L.

$$\left.\begin{array}{l} 1)\ P(1) \\ 2)\ P(K) \Rightarrow P(K+1) \end{array}\right\} \text{Principio de Inducción completa}$$

Previo

Def. de longitud (Recurrencia)

$| \ | : \vec{\Sigma}^* \to \mathbb{N}$ tal que

$$\begin{cases} |\lambda| = 0 \\ |\omega a| = |\omega| + 1 \end{cases}$$

Σ $\omega = baab$

$|\omega| = 4$

$|\omega| = |baab| = |baa| + 1 = |ba| + 2 = |\lambda b| + 3$

$|\lambda| + 4 = 0 + 4 = \underline{4}$ //

(4) Demostrar

$$|uv| = |u| + |v| \quad \forall u, v \in \Sigma^*$$

Prueba:

$$L = \{ v \in \Sigma^* / |uv| = |u| + |v| \}$$

(i) $\lambda \in L$

$$|u\lambda| = |u| = \underbrace{|u| + 0} = \underline{|u| + |\lambda|} \quad \therefore \lambda \in L$$

es neutro

Como $|u|$ es número no se ve afectado al sumarle un 0.

(ii) $w \in L \wedge a \in \Sigma^* \Rightarrow wa \in L$

Supongamos:

$$w \in L \quad \wedge a \in \Sigma^*$$

$$|uw| = |u| + |w| \wedge a \in \Sigma^*$$

P.d $wa \in L$

P.d $|u(wa)| = |u| + |wa|$

$$|u(wa)| = |(uw)a| = |uw| + 1 = (|u| + |w|) + 1 =$$

$$|u(wa)| = |u| + (|w| + 1)$$

$$|u(wa)| = |u| + \underline{|wa|}$$

$$\therefore wa \in L$$

Entonces $L = \Sigma^*$

Def.

Si $w = \sigma_1\sigma_2\sigma_3 \ldots \sigma_n \in \Sigma^*$; $\sigma_i \in \Sigma^*$

a la palabra $w' = \sigma_n \ldots \sigma_2\sigma_1$

se la llama Inversa o transpuesta de w

(es decir a la palabra escrita en orden inverso)

E) $\Sigma = \{a, b\}$

$w = baa$

$w' = aab$

Por recurrencia

$' : \Sigma^* \to \Sigma^*$

$$\begin{cases} \lambda' = \lambda \\ (wa)' = aw' \end{cases}$$

$w' = (baa)' = a(ba)' = aa(\lambda b)' = aab(\lambda)$

$(baa)' = aab\lambda = \underline{aab}$ //

| 1er Parcial 20 septiembre
| 2do Parcia 13 noviembre
| Final 20 noviembre

Ejercicio: Demostrar

$|w|' = |w|$; $\forall w \in \Sigma^*$

$L = \{ w \in \Sigma^* / |w|' = |w| \}$

Prueba.

$$L = \{ w \in \Sigma^* / |w| = |w'| \}$$

i) $\lambda \in L$?

$$\lambda' = \lambda \Rightarrow |\lambda'| = |\lambda|$$

ii) p.d que $w \in L \wedge a \in \Sigma \Rightarrow wa \in L$

Supongamos que

$w \in L \wedge a \in \Sigma$

$|w'| = |w| \wedge a \in \Sigma$ H.I

p.d $wa \in L$

p.d $|wa'| = |wa|$

$|wa'| = |aw'| = |a| + |w'| = |w'| + |a|$

pero $|w'| = |w|$

$|w| + |a| = |wa|$

$\therefore wa \in L$

ENTONCES $L = \Sigma^*$ (Todas cumplen la propiedad)

Crecimiento por izquierda

(iii) $w \in L \wedge a \in \Sigma \Rightarrow aw \in L$

Potencia de una Palabra

$$w^n = \underbrace{ww \ldots w}_{n \text{ veces}} \qquad w^1 = w \qquad w^0 = \lambda$$

Por recurrencia

$$\begin{cases} w^0 = \lambda \\ w^{n+1} = w w^n \end{cases}$$

1) $|w^n| = n|w|$
2) $w^m w^n = w^{m+n}$
3) $(w^n)^m = w^{nm}$
4) $\lambda^n = \lambda$

a) Demostrar todas las propiedades

b) Buscar todas las palabras u, v sobre $\Sigma = \{a,b\}$ talque $u^2 v^2 \neq (uv)^2$

Prefijo, Sufijo y subpalabras

Sean $v, z \in \Sigma^*$

- Se dice que v es prefijo de z ssi existe $u \in \Sigma^*$ talque $z = vu$ y se escribe v pref. z
- Se dice que v es sufijo de z ssi existe $w \in \Sigma^*$ talque $z = wv$ y se escribe v suf. z
- Se dice que v es subpalabra de z ssi existe $u_1, u_2 \in \Sigma^*$ talque $z = u_1 v u_2$ y se escribe v subp z

E) $\Sigma = \{a,b\}$

$z = aaabbbb$

$z = \lambda aaabbb$

$z = \underbrace{aa}_{u_1} \lambda \underbrace{abbbb}_{u_2}$

1) Cuantos prefijos, sufijos, subpalabras tiene la palabra

$z = a_1 a_2 \ldots\ldots a_n \in \Sigma^*$

n prefijos, n sufijos

2) Demostrar

a) x pref. $y \wedge y$ pref $x \Rightarrow x = y$

b) x pref $y \wedge y$ pref $z \Rightarrow x$ pref z

Lenguajes

Def. Sea Σ un alfabeto, un lenguaje sobre Σ es cualquier subconjunto de Σ^*

ε

$$\Sigma = \{a, b\}$$

$$\Sigma^* = \{\lambda, a, b, ab, ba, aa, bb, aaa, bbb, \ldots\}$$

Ejemplos:

$\emptyset$ y Σ^* porque $\rightarrow A \subseteq B$

$$\emptyset \subseteq A$$
$$A \subseteq A$$

$$L_4 = \{ba, baa, baaa, \ldots\ldots\}$$

$$L_4 = \{w \in \Sigma^* / w = ba^n, n > 0\}$$

Operaciones

1) union
2) Intersección
3) Diferencia
4) Complemento
5) Concatenación

Sean $L_1, L_2 \subseteq \Sigma^*$

$$L_1 L_2 = \{ w \in \Sigma^* / w = xy,\ x \in L_1 \wedge y\ L_2 \}$$

6) Transposicion

Sea $L \subseteq \Sigma^*$

$$L' = \{ w' \in \Sigma^* / w \in L \}$$

7) Estrella de Kleene $\rightarrow$ $L^* = \bigcup_{n=0}^{\infty} A^n$

Sea $L \subseteq \Sigma^*$

$$L^* \ \{ w \in \Sigma^* / \begin{array}{l} w = w_1 w_2 \ldots w_n, \text{ Para algunos} \\ w_1, w_2 \ldots, w_k \in L \text{ y } \exists n \in \mathbb{N} \end{array} \}$$

Sea $\Sigma = \{a, b\}$ y sean $P = \{a, ab\}$, $Q = \{\lambda, a, ba\}$

determinar:

(a) $P \cup Q$ (b) $P \cap Q$ (c) $P - Q$ d) $Q - P$ (e) P' (f) Q'

(g) $P \cdot Q$ (h) $P^2 = PP$

② Sean $A, B, C \subseteq \Sigma^*$ Demostrar

(a) $A \cdot \emptyset = \emptyset A = \emptyset$

(b) $A\{\lambda\} = \{\lambda\}A = A$

(c) $A(BC) = (AB) \cdot C$

(d) $(A \cup B)\, C = AC \cup B \cdot C$

(e) $A(B \cup C) = A \cdot B \cup A \cdot C$

③

(c) $P - Q = \{ab\}$, (d) $Q - P = \{\lambda, ba\}$

(f) $Q' = \{\lambda, a, ab\}$

(h) $P \cdot Q = \{a, aa, aba, ab, abba\}$

(g) $P^2 = \{aa, aab, aba, abab\}$

Sea $L_1 L_2 \subseteq \Sigma^*$. Demostrar

$$(L_1 \cap L_2)^c = L_1^c \cup L_2^c$$

<u>Prueba</u>:

p.d. $(L_1 \cap L_2)^c = L_1^c \cup L_2^c$

p.d $\underbrace{(L_1 \cap L_2)^c \subseteq (L_1^c \cup L_2^c)}_{\mathrm{I}} \wedge \underbrace{(L_1^c \cup L_2^c) \subseteq (L_1 \cap L_2)^c}_{\mathrm{II}}$

I) $(L_1 \cap L_2)^c \subseteq (L_1^c \cup L_2^c)$ { Condicional

p.d $w \in (L_1 \cap L_2)^c \Rightarrow w \in L_1^c \cup L_2^c$ { Suponer el antecedente demostrar el consecuente }

Sea $w \in (L_1 \cap L_2)^c \overset{\text{compl}}{\Rightarrow} w \notin L_1 \cap L_2$

$\overset{\text{Notación}}{\Longrightarrow} \sim [w \in (L_1 \cap L_2)]$

$\overset{\text{def. inters}}{\Longrightarrow} \sim [w \in L_1 \wedge w \in L_2]$

$\Rightarrow w \notin L_1 \vee w \notin L_2$ D'morgan

$\Rightarrow w \in L_1^c \vee w L_2^c$ Def. Complemento

$\Rightarrow w \in L_1^c \cup L_2^2$ Def. Unión

$\therefore w \in (L_1 \cap L_2)^c \Rightarrow w \in L_1^c \cup L_2^c$

$\Longleftrightarrow \boxed{(L_1 \cap L_2)^c \subseteq (L_1^c \cup L_2^c)}$ //

II / $(L_1^c \cup L_2^c) \subseteq (L_1 \cup L_2)^c$

(p.d) $w \in (L_1^c \cup L_2^c) \Rightarrow w \in (L_1 \cup L_2)^c$

Sea $w \in (L_1^c \cup L_2^c) \Longrightarrow w \in L_1^c \vee w \in L_2^c$ def. Unión

$\Rightarrow w \notin L_1 \vee w \notin L_2$ def. Complemento

$\Rightarrow \sim[w \in L_1 \wedge w \in L_2]$ D'morgan

$\Rightarrow \sim[w \in L_1 \cap L_2]$ Def. Intersección

$\Rightarrow w \notin L_1 \cap L_2$ Notación

$\Rightarrow w \in (L_1 \cap L_2)^c$ Def. Complemento

$\therefore w \in (L_1^c \cup L_2^c) \rightarrow w \in (L_1 \cap L_2)^c$

$\Longleftrightarrow \boxed{(L_1^c \cup L_2^c) \subseteq (L_1 \cap L_2)^c}$ //

I , II $\Longleftrightarrow (L_1^c \cup L_2^c) = (L_1 \cap L_2)^c$

Sean $L_1, L_2, L_3 \subseteq \Sigma^*$

$$L_1 \subseteq L_3 \land L_2 \subseteq L_3 \Rightarrow L_1 \cup L_2 \subseteq L_3$$

Prueba:

P.d $L_1 \cup L_2 \subseteq L_3$

P.d $x \in L_1 \cup L_2 \Rightarrow x \in L_3$

1)	$L_1 \subseteq L_3 \land L_2 \subseteq L_3$	R.P
2)	$L_1 \subseteq L_3$	sim 1
3)	$L_2 \subseteq L_3$	simp 2
4)	$x \in L_1 \Rightarrow x \in L_3$	def. $\subseteq$ 2
5)	$x \in L_2 \Rightarrow x \in L_3$	def. $\subseteq$ 3
6)	$x \in L_1 \cup L_2$	R.P
7)	$x \in L_1 \lor x \in L_2$	"$\cup$" 6
8)	$x \in L_3 \lor x \in L_3$	Sim Dis 4, 5, 7
9)	$x \in L_3$	sim Disy 9
10)	$x \in L_1 \cup L_2 \Rightarrow x \in L_3$	DC 6, 10
12	$L_1 \cup L_2 \subseteq L_3$	def. $\subseteq$ 10
13	$L_1 \subseteq L_3 \land L_2 \subseteq L_3 \Rightarrow L_1 \cup L_2 \subseteq L_3$	

$A \subseteq B \Rightarrow A \cup B = B$

Prueba:

pd: $A \cup B = B$

$A \cup B \subseteq B \wedge B \subseteq A \cup B$

I pd $x \in A \cup B \Rightarrow x \in B$

Sea $x \in A \cup B \Rightarrow x \in A \vee x \in B$ def. unión

$\Rightarrow x \in B \vee x \in B$ Hip.

$\Rightarrow x \in B$ simp disy.

$\therefore x \in A \cup B \Rightarrow x \in B \Leftrightarrow A \cup B \subseteq B$

Demostrar

$$A\{\lambda\} = \{\lambda\}A = A$$

Prueba:

(pd) $A\{\lambda\} = A$

(p.d) $\underbrace{A\{\lambda\} \subseteq A}_{\text{I}} \wedge A \subseteq A\{\lambda\}$

(pd) $w \in A\{\lambda\} \Rightarrow x \in A$

si $\exists x = \{\lambda\}$
$\Rightarrow x = \lambda$

Sea $w \in A\{\lambda\} \xRightarrow{\text{Def. concatenación}} w = xy \wedge x \in A \wedge y \in \{\lambda\}$

$\xRightarrow{\{\lambda\}} w = xy \wedge x \in A \wedge y = \lambda$

$\xRightarrow{y=\lambda} w = x\lambda \wedge x \in A$

$\xRightarrow{\lambda \text{ neutro}} w = x \wedge x \in A$

$\xRightarrow{w=x} w \in A$

$\therefore$ Dc $w \in A\{\lambda\} \Rightarrow w \in A \Leftrightarrow A\{\lambda\} \subseteq A$

falta (2)

Expresiones regulares son palabras sobre un alfabeto

La función L:

$$\mathcal{R} \longrightarrow \mathcal{L}$$

$$\phi \longrightarrow \phi$$
$$a \longrightarrow \{a\}$$
$$(\alpha \cup \beta) \longrightarrow L\{\alpha\} \cup L\{\beta\}$$
$$(\alpha \beta) \longrightarrow L\{\alpha\} L\{\beta\}$$
$$\alpha^* \longrightarrow L\{\alpha\}^*$$

1. $L(\phi) = \phi$
2. $L(a) = \{a\} \; ; \forall a \in \Sigma$
3. $L(\alpha \cup \beta) = L(\alpha) \cup L(\beta)$
4. $L(\alpha\beta) = L(\alpha)L(\beta)$
5. $L(\alpha^*) = L(\alpha)^*$

E Sea $\Sigma \{a, b\}$

y sea $\alpha = a^*$ una E.R escribir por comprensión $L(\alpha)$.

$$L(\alpha) = L(a^*) = L(a)^* = L\{a\}^* = \{\lambda, a, aa, aaa, aaaa \ldots\ldots\}$$

$$= \{w \in \Sigma^* / w = a^n ; n \geq 0\}$$

$\alpha = (a^* b)$

Escribir $L(\alpha)$ por comprensión

$L(\alpha) = L((a^* b)) = L(a^*) L(b)$

$= L(a^*) L(b)$

$= L(a)^* L\{b\}$

$= L\{a\}^* L\{b\}$

$= \{\lambda, a, aa, aaa \ldots\}\{b\}$

$= \{b, ab, aab, aaab, \ldots\}$

$\{w \in \Sigma^* / w = a^n b, n \geq 0\}$

(3) Sea $\Sigma = \{a, b\}$

y $\alpha = b(a \cup b)^* b$

escribir $L(\alpha)$ por comprensión

$L(\alpha) = L((\underbrace{b(a \cup b)}^* b)) = L(b(a \cup b)^*) L(b)$

$= L(b) L(a \cup b)^* L\{b\}$

$= L\{b\} L(a \cup b)^* L\{b\}$

$= \{b\} (L(a) \cup L(b))^* \{b\}$

$= \{b\} (\{a\} \cup \{b\})^* \{b\}$

$= \{b\} \{a, b\}^* \{b\}$

$= \{b\} \{\lambda, a, aa, ab, abb, \ldots\} \{b\}$

$= \{w \in \Sigma^* / w = bub, u \in \Sigma^*\}$ //

Lenguaje Regular

Un lenguaje es regular ssi es generado por una expresión regular

T.

La clase de L.R es la clase mas pequeña que contiene al vacío $\emptyset$ y conjuntos unitarios $\{a\}$ $\forall a \in \Sigma^*$ y es cerrada bajo la unión, concatenación y estrella de Kleene.

$$\Sigma = \{a, b\}$$

$$L(a \cup b) = L(a) \cup L(b)$$
$$= L\{a\} \cup L\{b\}$$
$$= \underline{\underline{\{a, b\}}} \longrightarrow \text{generado por } (a \cup b)$$

$$\Rightarrow L = \{b, ba\}$$
$$\downarrow$$
$$b \cup ba$$

$$\Rightarrow L = \{a, ab, ba, b\}$$
$$a \cup ab \cup ba \cup b$$

Cualquier lenguaje finito es un lenguaje regular, puede ser escrito por una expresión regular.

Para ver la expresión regular juntar todos los elementos con '$\cup$'.

$$u_1 \cup u_2 \cup u_3 \cup u_4$$

Modulos
Similar a una correa.

- Parada
- Mov. Izquierda.
- Mov. derecha.

Control: llave

- Posición superior
- Pos. Inferior.

llave / Posic.	Inf.	Sup.
Parada	Izq	Derecha
Izq.	Izq	"
Der	Parada	"

Correa : P, I, D
Llave : – +

llave / Pos	–	+
p	i	d
I	i	d
d	p	d

Def. Los últimos símbolos describen el alfabeto de entrada. (p, i, d, ~, +)

Def. Un modulo es una tripleta

$$\mathcal{D} = (K, \Sigma, f) \text{ donde:}$$

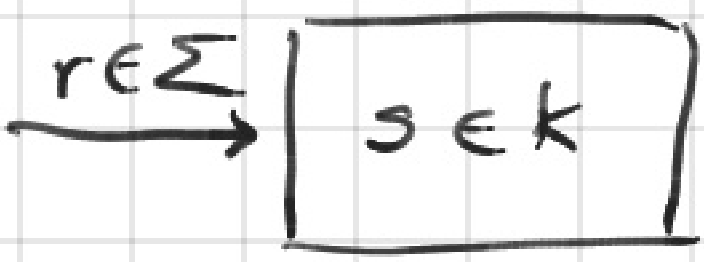
K = Es un conjunto finito no vacío, llamado conjunto de estados.

Σ = Es un conjunto finito no vacío, llamado alfabeto de entrada

$f: K \times \Sigma \to K$, llamada función de transición

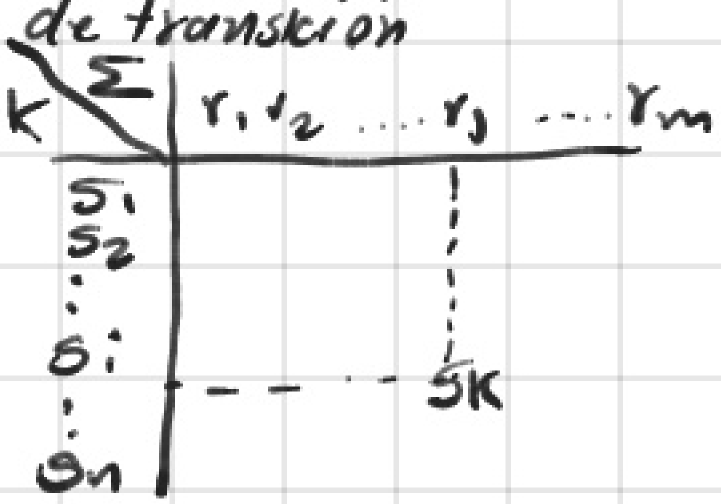
Interpretación:

Un modulo se puede interpretar como un dispositivo que en determinados instantes de tiempo recibe señales (símbolos del alfabeto). Que producen cambios en su configuración interna.



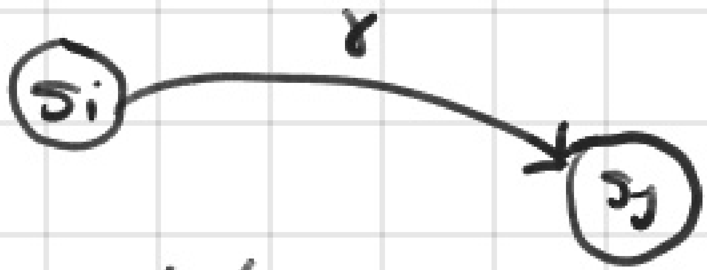
Representación:

1) Tabla de transición



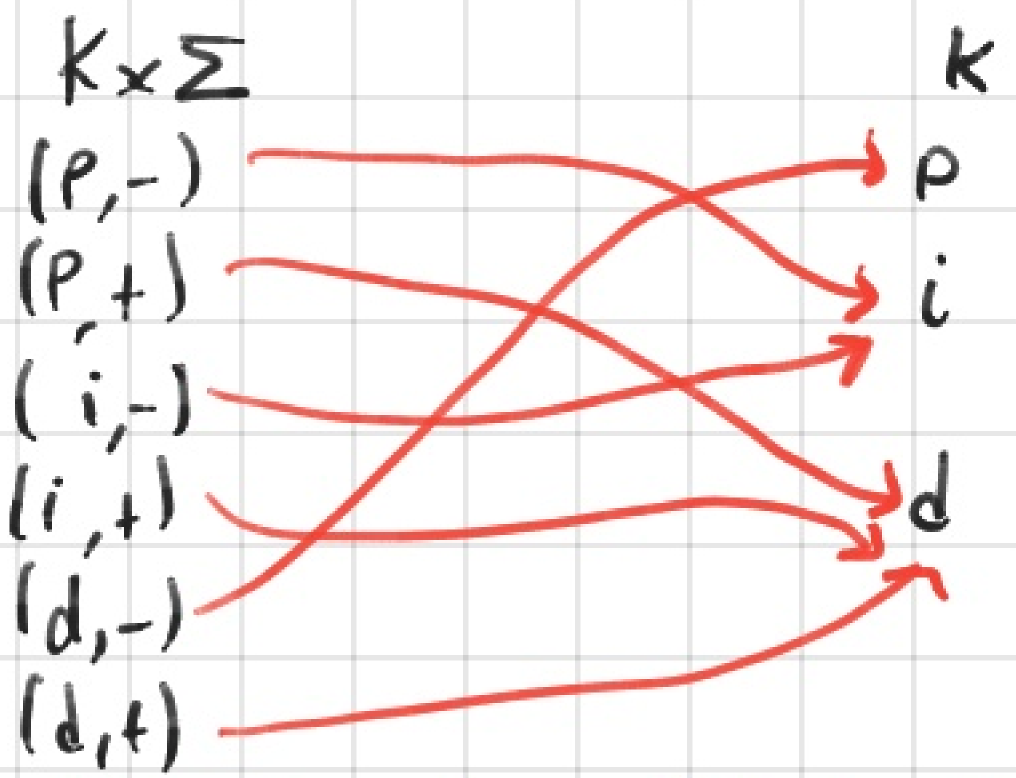
$$s_k = f(s_i, r_j)$$

Grafo de transición:



Los vertices rotulados por los estados.

$$K = \{p, i, d\}$$
$$\Sigma = \{+, -\}$$
$$f = K \times \Sigma \to K$$
$$K \times \Sigma = \{(p,-), (p,+), (i,-), (i,+), (d,-), (d,+)\}$$



$f(p,-) = i$ $\quad$ $f(d,-) = p$

$f(p,+) = d$ $\quad$ $f(d,+) = \emptyset$

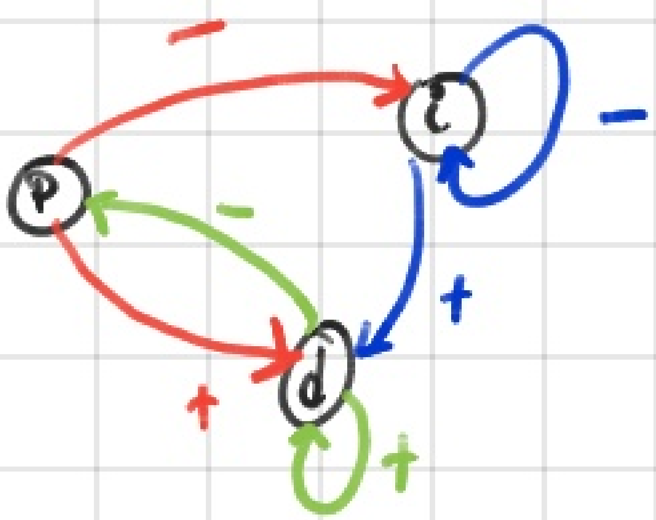
$f(i,-) = i$

$f(i,+) = d$

a) Tabla

K \ Σ	−	+
p	i	d
i	i	d
d	p	d

Grafo de transición:



Comportamiento Dinámico

$$\begin{array}{ccccccc} t_2 & & t_2 & \ldots & t_i & \ldots & t_K \end{array}$$

$$s_1 \xrightarrow{\sigma_1} s_2 \xrightarrow{\sigma_2} s_i \longrightarrow s_K$$

Depende del estado de entrada y la palabra de entrada (Sucesión finita de símbolos de sigma).

$$s_i = f(s_{i-1}, \sigma_{i-1})$$

Def. Una función de estado terminado del módulo $\mathcal{D}=(K,\Sigma,f)$ es una única función $\hat{f}: K \times \Sigma^* \to K$

tal que para todo $s \in K, \omega \in \Sigma^*, \sigma \in \Sigma$

$$\begin{cases} \hat{f}(s,\lambda) = s \\ \hat{f}(s,\sigma\omega) = \hat{f}[f(s,\sigma),\omega] \end{cases}$$

---

si $\omega = \lambda$

$$\hat{f}(s,\sigma) = \hat{f}(s,\sigma\lambda) = \hat{f}[\hat{f}(s,\sigma),\lambda] = f(s,\sigma)$$

Cuando es un solo símbolo, directo la misma función

2) $f_\omega : K \to K \quad ; \forall \omega \in \Sigma^*$

$$f_\omega(s) = \hat{f}(s,\omega)$$

Ejemplo:

$K = \{p, s, t\}$

$\Sigma = \{+, -\}$

$K \backslash \Sigma$	+	−
p	s	p
s	t	p
t	t	s

3 señales de subida y 1 de bajada

$$\hat{f}(p, +++-) = s$$

$$p \xrightarrow{+} s \xrightarrow{+} \xrightarrow{+} t \xrightarrow{-} s$$

$$\hat{f}(p, +++-) = \hat{f}[\hat{f}(p,+), ++-]$$

$$\hat{f}[s, ++-]$$

$$\hat{f}[\hat{f}(s,+), +-]$$

$$\hat{f}[t, +-]$$

$$\hat{f}[\hat{f}(t,+), -]$$

$$\hat{f}(p, +++-) = \hat{f}[t, -] = s$$

• Demostrar

Sea $\mathcal{D} = (K, \Sigma, f)$

1. Demostrar $\hat{f}(s, uv) = \hat{f}[\hat{f}(s,u), v]$; $\forall u,v \in \Sigma^*$
2. $\tilde{f} : K \times \Sigma^* \to K$ / $\forall s \in K, w \in \Sigma^*, r \in \Sigma$

$$\begin{cases} \tilde{f}(s, \lambda) = s \\ \tilde{f}(s, wr) = f[\tilde{f}(s,w), r] \end{cases}$$

Demostrar : $\tilde{f}(s,u) = \hat{f}(s,u)$; $\forall u \in \Sigma^*$

5) Sea $\overline{f}: K * \Sigma^* \to K^* / \forall s \in K, \omega \in \Sigma^*, \sigma \in \Sigma$

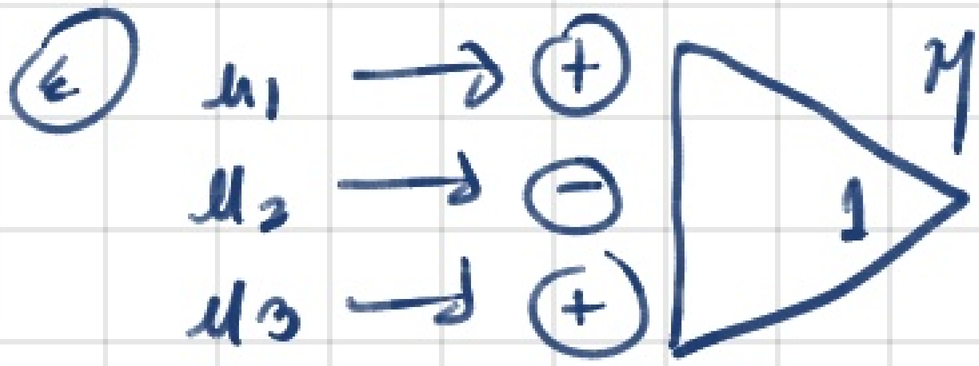
$$\overline{f}(s, \lambda) = \lambda$$
$$\overline{f}(s, \sigma \omega) = f(s, \sigma) \overline{f}[f(s, \gamma), \omega]$$

Demostrar:

a) $\overline{f}(s, \sigma) = f(s, \sigma)$

b) $|\overline{f}(s, u)| = |u|$; $\forall u \in \Sigma^*$

c) $\overline{f}(s, u\sigma) = \overline{f}(s, u) \hat{f}(s, u\sigma)$; $\forall u \in \Sigma^*$

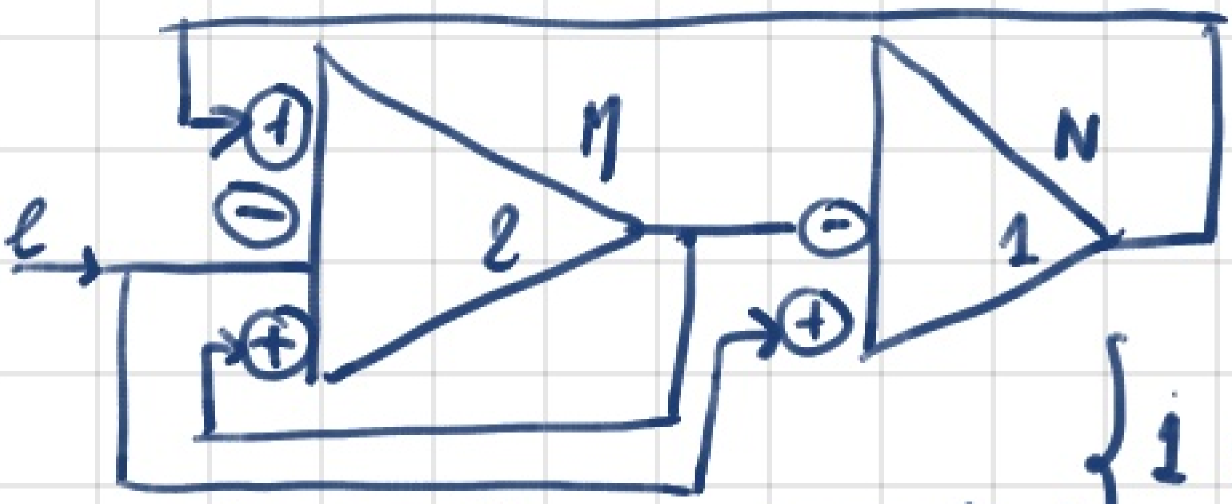


$$S(K+1) = \begin{cases} 1 & \text{si } u_1(k) + u_3(k) - u_2(k) \geq 1 \\ 0 & \text{en otro caso.} \end{cases}$$

estará activa si la suma de estimos exitatorios menos la suma de limites inhibitorios alcanza o Supera su límite de actividad (1)

K \ Σ	(0,0,0)	(0,0,1)	(0,1,0)	(1,0,0)	(1,1,0)	(1,0,1)	(0,1,1)	(1,1,1)
1	0	1	0	1	0	1	0	1
0	0	1	0	1	0	1	0	1

de acuerdo a los resultados no ocupe al obtener los valores, los ocuparía en el caso de que esten definidos en la función $S(i+1)$ { ......



$M \to s(i)$

$N \to t(i)$

$$M \to s(i+1) \begin{cases} 1 & \text{si } t(i)+s(i)-e(i) \geq 2 \\ 0 & \text{en otro caso} \end{cases}$$

$$N \to t(i+1) \begin{cases} 1 & \text{si } e(i)-s(i) \geq 1 \\ 0 & \text{en otro caso} \end{cases}$$

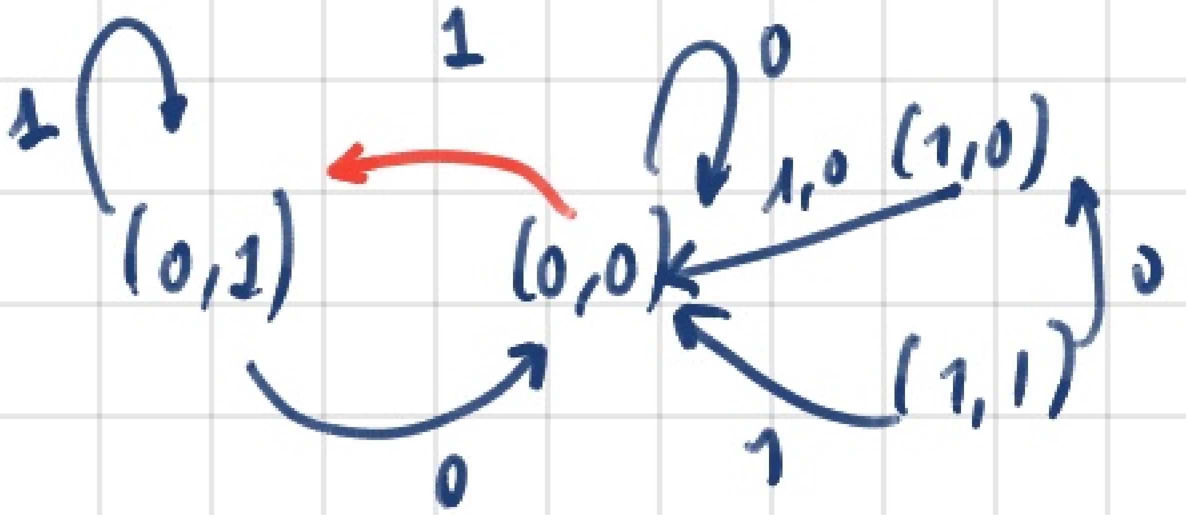
M N

Σ \ K	(0,0)	(0,1)	(1,0)	(1,1)
1	(0,1)	(0,1)	(0,0)	(0,0)
0	(0,0)	(0,0)	(0,0)	(1,)

Son los estado de las neuronas.

0+0−0

entrada → e



desde (1,1) puedo llegar a (0,1) con la palabra

1,1

También puedo realizarlo con longitud de palabras

$\exists$ (1,1) a (0,1) con la palabra: |6|

$$1,\underbrace{0,0,0,0}_{(0,0)},1$$

- $T = \{(0,0), (0,1)\}$

$$\hat{f}\,[(0,0), w] \in T, \underbrace{\forall w \in \Sigma^*}$$

Demostrar.

Mediante P/C

Alcanzabilidad

Def. Sea $\mathcal{D} = (K, \Sigma, f)$ un módulo y sean $s, t, \in K$

Decimos que t es alcanzable es alcanzable a partir

$$\hat{f}(s,w) = t$$

y se escribe $s \xrightarrow{*} t$

Decimos que t es k-alcanzable a partir de s ssi existe $w \in \Sigma^*$ tal que $|w| = K$ y $\hat{f}(s,w) = t$ y se escribe

$s \xrightarrow{K} t$

Demostrar.

1) si $s \xrightarrow{i} q$ y $q \xrightarrow{J} t$ entonces $s \xrightarrow{i+J} t$

2) si $s \xrightarrow{K} t$ y $K = i + J$ entonces existe $q \in K$ tal que $s \xrightarrow{i} q$ y $q \xrightarrow{J} t$

3) Del modulo del ejemplo anterior determinar determinar todos los estados alcanzables a partir de $(1, 0)$

2 personas lanzan una moneda repetidamente en cada instante el dispositivo M:

- Recibe uno de los símbolos a (simboliza cara) o b (simboliza cruz)
- Emite un símbolo (1 o 2) indicando que jugador 1 o 2 recibe 1 punto.
- Puede estar en uno de los sgt. 3 estados.

C: indicando que el lanzamiento anterior fue cara.

p: indicando que la cruz que salió en el lanzamiento anterior fue la primera de una sucesión de cruces.

d: indicando que ya salieron 2 o más cruces en la sucesión

el funcionamiento del sistema será el siguiente.

- recibiando el símbolo a, pasará al estado c independiente del estado actual y emitirá el 2 si estaba en el estado c y 1 en caso contrario.
- Recibiendo el símbolo b emitirá el símbolo 1, quedando en el mismo estad. si estaba en d. Caso contrario la salida será 2, si esta en C pasará a p; si estaba en p, pasará a d.

Este funcionamiento puede ser descrito por 2 funciones.

$f: K \times \Sigma^* \to K$ para ser función debe cumplir 2

$K = \{c, p, d\}$ condiciones, totalidad y unicidad.

$\Sigma^* = \{a, b\}$

$K \times \Sigma^* = \{(c,a), (c,b), (p,a), (p,b), (d,a), (d,b)\}$

$f(c,a) = c$ · $g(c,a) = 2$

$f(c,b) = p$ $g: K \times \Sigma \to \Delta$ $g(c,b) = 2$

$f(p,a) = c$ · $g(p,a) = 1$

$f(p,b) = d$ para las $g(p,d) = 2$

$f(d,a) = c$ salidas ya que · $g(d,a) = 1$

$f(d,b) = d$ en los modulos no estaban. $g(d,b) = 1$

Las salidas aquí son 1 y 2.

Máquina: Una máquina es una quíntupla

$M = (K, \Sigma, \Delta, f, g)$

K = Conjunto finito no vacío llamado conjunto de estados

Σ = Conjunto finito no vacío llamado Alfabeto de entrada.

Δ = Conjunto finito no vacío llamado alfabeto de salida

$f: K \times \Sigma \to K$ llamado función de transición

$g: K \times \Sigma \to \Delta$ llamado función de salida

Interpretación: Se puede interpretar como un dispositivo que en determinados instantes de tiempo recibe señales (simbolos de entrada) que producen cambios en su conf. interna y emiten señales (simbolos de salida).

$$\sigma \in \Sigma \longrightarrow \boxed{s \in K} \longrightarrow \delta \in \Delta$$

Representación:

a) Tabla de transicion

$K \backslash \Sigma$	σ_1	σ_2	...	σ_j	...	σ_m
s_1						
s_2						
⋮						
s_i				s_k/δ_k		
⋮						
s_n						

$f(s_i, \sigma_j) = s_k$

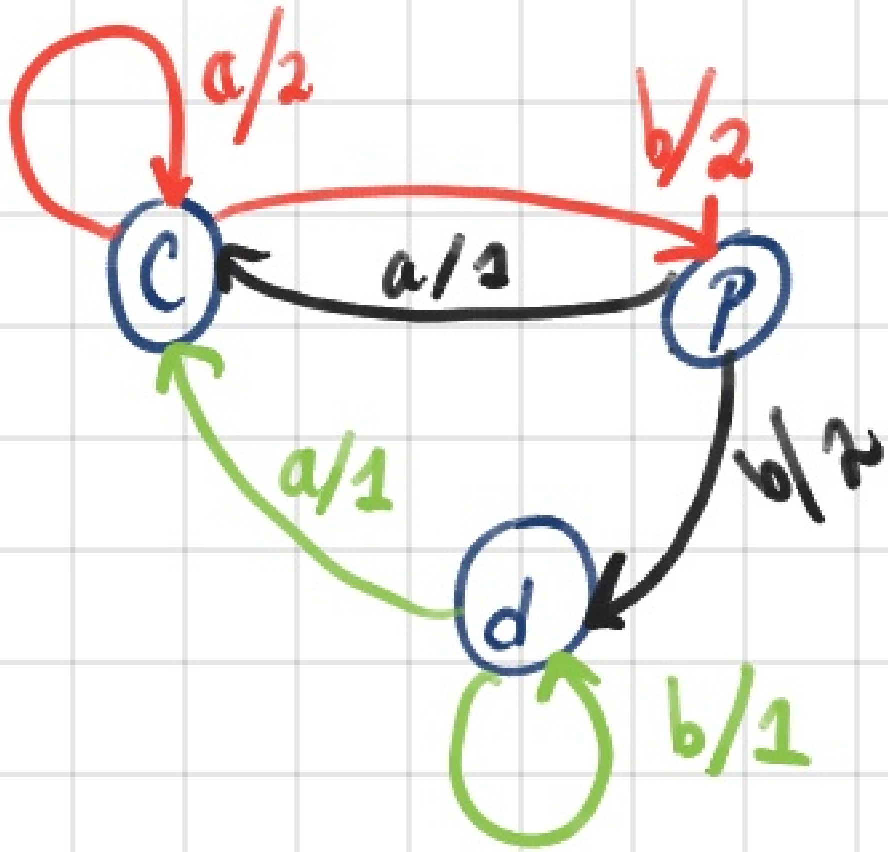
$g(s_i, \sigma_j) = \delta_k$

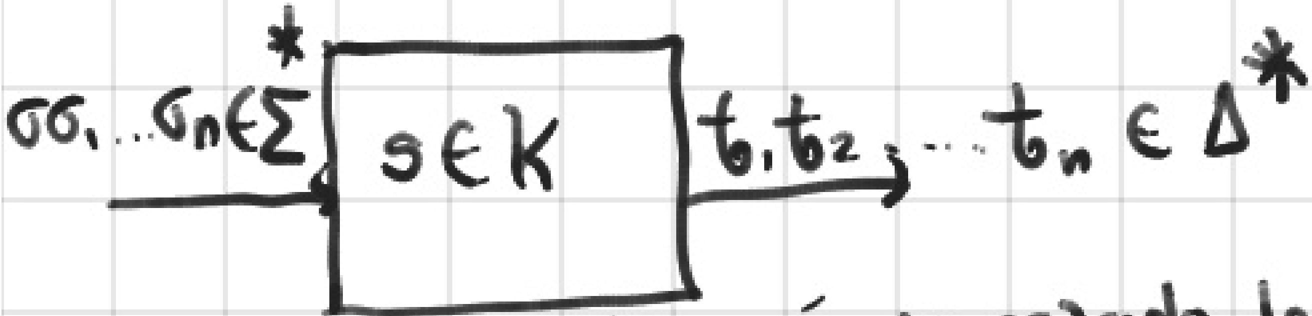
b) grafo)

$$s_i \xrightarrow{\sigma/\delta} s_j$$

$f(s_i, \sigma) = s_j \wedge g(s_i, \sigma) = \delta$

K \ Σ	a	b
c	c/2	p/2
p	c/1	d/2
d	c/1	d/1





función de estado terminado

$$\hat{f} : K \times \Sigma^* \to K$$

$$\bar{g} = K \times \Sigma^* \to \Delta^*$$

función palabra de salida

Comportamiento dinámico

sea $M = (K, \Sigma, \Delta, f, g)$

$t_1 \quad t_2 \quad \ldots \quad t_i \quad \ldots \quad t_k$

$s_1 \xrightarrow{\sigma_1/\delta_1} s_2 \xrightarrow{\sigma_2/\delta_2} s_i \cdots \xrightarrow{\sigma_{k-1}/\delta_{k-1}} s_k$

$s_i = f(s_{i-1}, \sigma_{i-1}) \qquad s_k = \hat{f}(s_1, \sigma_1 \sigma_2 \ldots \sigma_{k-1})$

$\delta_i = g(s_i, i) \qquad s_k = \hat{f}(s_1, u)$

$u = \sigma_1 \sigma_2 \ldots \sigma_{k-1}$

$\bar{g}(s_1, \sigma_1 \sigma_2 \ldots \sigma_{k-1})$

$\bar{g}(s_1, u) = \delta_1 \delta_2 \ldots \delta_k$

Def. Sea $M = (K, \Sigma, D, f, g)$ una maquina

a) Una función de estado terminado de s, la función de estado terminal

$\hat{f} : K \times \Sigma^* \to K$ del módulo (K, Σ, f)

b) Una función palabra de salida de M es una maquina función $\bar{g} : K \times \Sigma^* \to \Delta^*$ tal que para todo $s \in K$

$w \in \Sigma^*$, $\sigma \in \Sigma$

$$\begin{cases} \bar{g}(s, \lambda) = \lambda \\ \bar{g}(s, \sigma w) = g(s, \sigma)\, \bar{g}[f(s, \sigma), w] \end{cases}$$

Nota:

1) $w = \lambda$

$$\bar{g}(s,\sigma) = \bar{g}(s, \sigma\lambda) = g(s,\sigma)\,\bar{g}[f(s,\sigma), \lambda]$$ funcion de palabra de salida

$$g(s,\sigma)\lambda = g(s,\sigma)$$ función de salida

2) Para cada $s \in K$

$$g_s : \Sigma^* \to \Delta^*$$
$$g_s(w) = \bar{g}(s,w) \quad ; \forall w \in \Sigma^*$$

Ejercicio.

1) sea la maquina

	a	b
p	p/α	r/β
q	r/α	p/α
r	r/β	$p/\gamma\rho$
s	r/γ'	p/α

¿Cual es la salida correspondiente a baba desde p?

2) Demostrar

a) $|\bar{g}(s,w)| = |w| \; ; \forall w \in \Sigma^*$

b) $\bar{g}(s,u\sigma) = \bar{g}(s,u)\,\bar{g}[\hat{f}(s,u), \sigma] ; \forall u,\sigma \in \Sigma^*$ función de estado terminado

$$\bar g(p, baba) = g(p,b)\,\bar g[f(p,b), aba] =$$

$$\beta\,\bar g(r, aba) = \beta\, g(r,a)\,\bar g(f(r,a), ba)$$

$$= \beta\beta\,\bar g(r, ba) = \beta\beta\, g(r,b)\,\bar g[f(r,b), a]$$

$$= \beta\beta\gamma'\,\underbrace{\bar g(p,a)} = \boxed{\beta\beta\gamma'\alpha}$$

$$\hookrightarrow \bar g(p,a) = g(p,a)$$

Síntesis, Análisis, Verificación

	I/O	máquina
Síntesis	✓	?
Análisis	✓	?
Verif.	✓ ⟷	✓

Ejercicio

① sea M:

	a	b
p	q/α	q/α
q	q/β	q/α

M. comenzando en p. representa el siguiente comportamiento: el 1er símbolo de la palabra de salida es α, y cada símbolo siguiente es el 'contrario' del símbolo correspondiente de entrada. (es decir, es α, si la entrada es B y es β si es a.)

Sea. M

	a	b
p	q/β	q/β
q	q/α	q/α

Verificar si la función palabra de salida $\bar{g}$ de la maquina M, tiene la sig. propiedad

$\bar{g}(p, \sigma u) = \beta \alpha^k$

$\forall \sigma \in \{a, b\}, u \in \{a, b\}^k, k \in \mathbb{N}$

Eje.

Sea $\hat{g} : K \times \Sigma^* \rightarrow \Delta / \forall s \in K, w \in \Sigma^*, \sigma \in \Sigma$

$$\hat{g}(s, w\sigma) = g[\hat{f}(s, w), \sigma]$$

Demostrar:

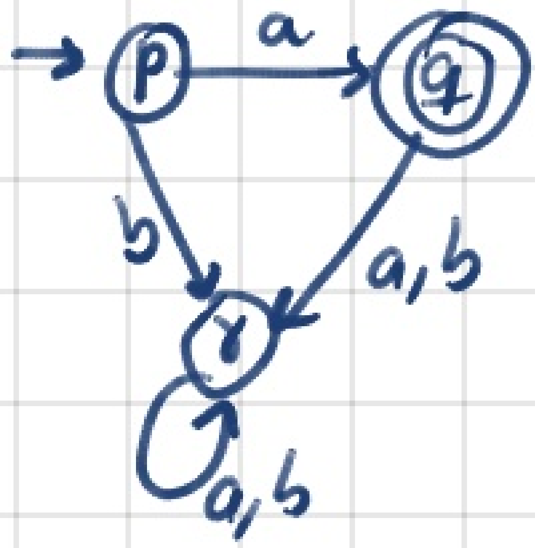
$$\bar{g}(s, u\sigma) = \bar{g}(s, u)\hat{g}(s, u\sigma)$$

$$\forall s \in K, u \in \Sigma^*, \sigma \in \Sigma$$

Sea $L_1 = \{a\}$

y sea M.

	a	b
$\rightarrow p$	q	r
q	r	r
r	r	r



Demostrar

$L(M) = L_1$

Prueba $\Sigma = \{a, b\}$

$\Sigma^* = \{\lambda, a, b, aa, ab, \ldots\}$

(i) $|w| < 2$

(ii) $|w| \geq 2$

$\hat{f} : K \times \Sigma^* \rightarrow K$

$f : K \times \Sigma \rightarrow K$

(i) si $|w| < 2$

si $w = \lambda \Rightarrow \hat{f}(p, \lambda) = p \notin F \quad \lambda \notin L(M)$

si $w = a \Rightarrow \hat{f}(p, a) = f(p, a) = q \in F \therefore a \in L(M)$

si $w = b \Rightarrow \hat{f}(p, b) = f(p, b) = r \in F \therefore b \notin L(M)$

Solo 'a' esta palabra es aceptada por el automata.

(ii) si $|w| \geq 2$

$w = \sigma \tau u \; ; \; \sigma, \tau \in \Sigma, \; u \in \Sigma^*$

$\hat{f}(p, w) = \hat{f}(p, \sigma \tau u) =$

$= \hat{f}[f(p, \sigma), \tau u] = \hat{f}(t, \tau u) \; ; \; t \in (q, r)$

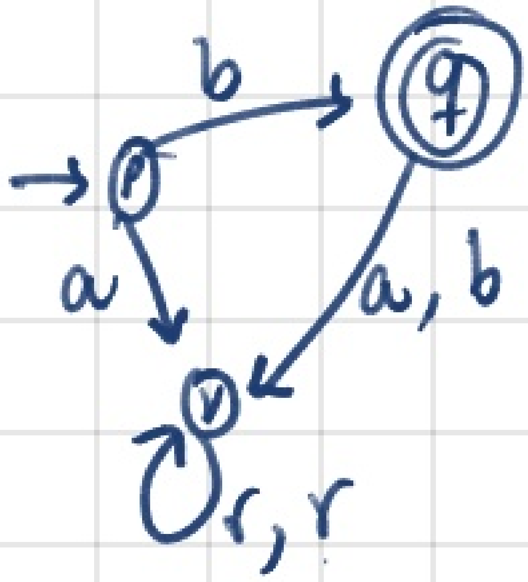
$= \hat{f}[f(t, \tau), u] = \hat{f}(r, u) = r \notin F \therefore w \notin L(M)$

$\therefore L(M) = L_1$

Sea $L_2 \{b\}$

y sea M

	a	b
$\rightarrow p$	r	q
<u>q</u>	r	r
r	r	r



Demostrar que $L(M)=L_2$

Prueba: $\Sigma \{a,b\}$

$\Sigma^* \{\lambda, a, b, aa, bb, \ldots\}$

(i) $|w|<2$

(ii) $|w|\geq 2$

si $|w|<2$

si $w=\lambda \Rightarrow \hat{f}(p,\lambda) = p \notin F \therefore \lambda \notin L(M)$

si $w=a \Rightarrow \hat{f}(p,a) = f(p,a) = r \notin F \therefore a \notin L(M)$

si $w=b \Rightarrow \hat{f}(p,b) = f(p,b) = q \notin F \therefore b \in L(M)$

si $|w|\geq 2$

si $w=\sigma\tau\mu$; $\sigma,\tau \in \Sigma$, $\mu \in \Sigma^*$

$$\hat{f}(p,\sigma\tau\mu) = \hat{f}[f(p,\sigma),\tau\mu] = \hat{f}(t,\tau\mu)\ ;\ t\in\{q,r\}$$

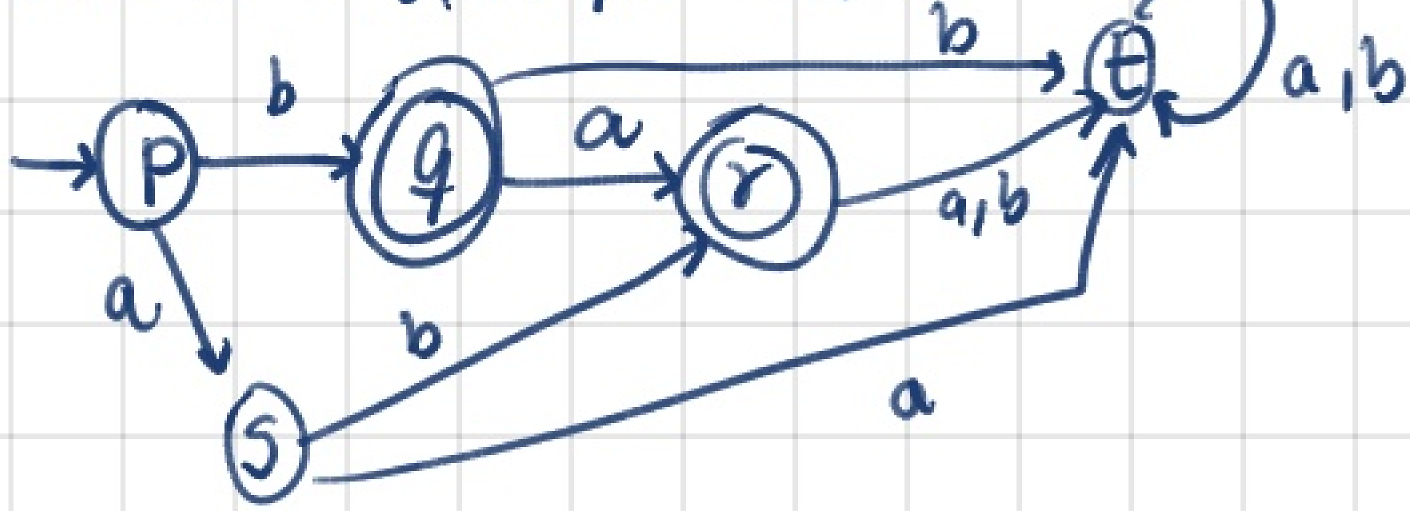
$$\hat{f}[f(t,\tau),\mu] = \hat{f}(r,\mu) = r \notin F \therefore w \notin L(M)$$

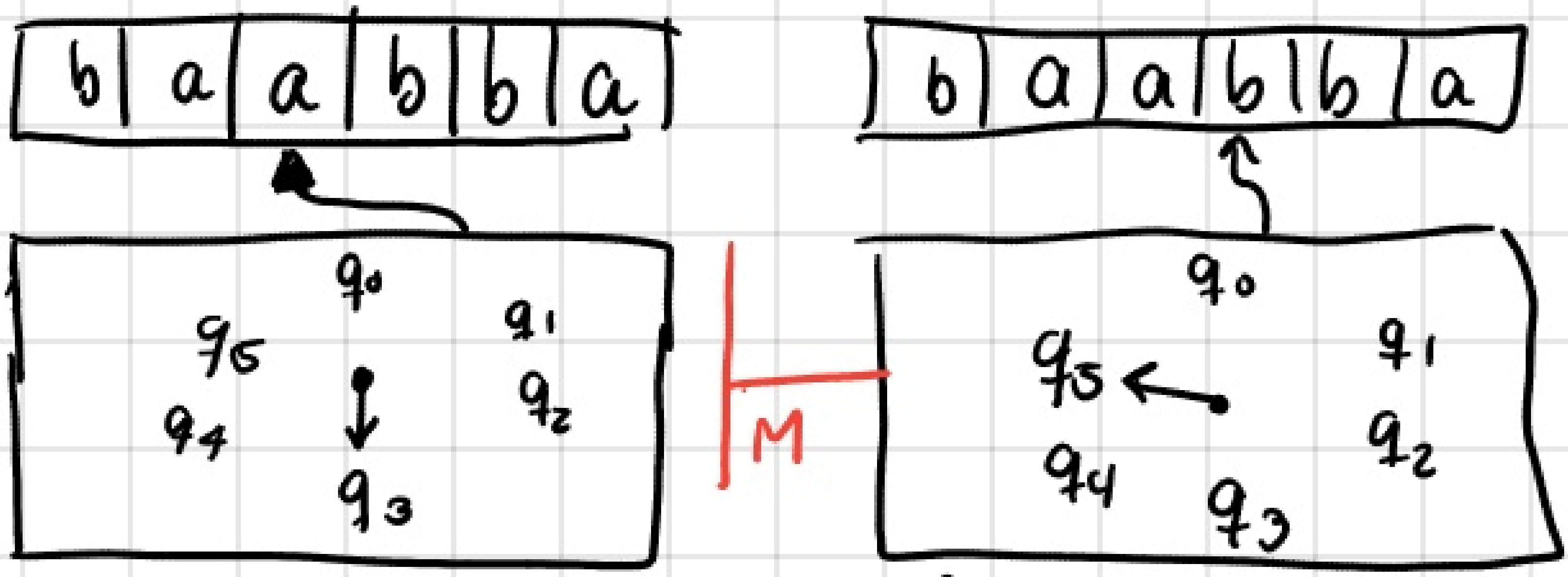
$\therefore L(M)=L_2$

$\Sigma = \{a, b\}$

$L = \{b, ab, ba\}$

Construir un AFD, tal que $L(M) = L$





(q_5, bba)

configuración:

$(q_3, abba)$

Sea $M = (K, \Sigma, \overset{\text{'f'}}{\delta}, s, F)$

Configuración es un par ordenado de $K \times \Sigma^*$

Se sustituira $\hat{f}(s, w) \in F$

la relación $\vdash_M$ → "conduce a" en un paso

Sea (q, w) y (q', w') 2 configuraciones

$(q, w) \vdash_M (q', w') \iff w = \sigma w'$, para algun $\sigma \in \Sigma$

y $\delta(q, \sigma) = q'$

	a	b
⋮		
q_3	q_5	

$(q_3, abba) \vdash_M (q_5, bba) \Longleftrightarrow$

$abba = \sigma bba$, $abba = abba$ ademas

$\delta(q_3, a) = q_5$

El paso de esa configuración a la otra se justifica con la función de transición

Denotamos por $\vdash_M^*$ es la clausura reflexiva transitiva del $\vdash_M$ y se lee "conduce a" en cero o mas pasos.

* sea $M = (K, \Sigma, \delta, s, F)$ un AFD

sea $\omega \in \Sigma^*$

se dice que M acepta ω ; $\underbrace{(ssi\ \hat{f}(s, \omega) \in F)}_{(s,\omega) \vdash_M^* (q, \lambda) \text{ para algun } q \in F}$

Lenguaje aceptado por M

$L(M) = \{\omega \in \Sigma^* / M \text{ acepta } \omega\}$

$L(M) = \{\omega \in \Sigma^* / (s, \omega) \vdash_M^* (q, \lambda); q \in F\}$

δ	a	b
$\rightarrow \underline{q_0}$	q_0	q_1
q_1	q_1	q_0

a) Procesar baabbab en términos de configuraciones.

b) Escribir $L(M)$ por comprensión

$$\begin{cases} \hat{f}(s,\lambda) = s \\ \hat{f}(s,\sigma w) = \hat{f}[f(s,\sigma), w] \end{cases}$$

sea $(q_0, baabbab) \vdash_M (q_0, aabbab) \vdash_M (q_1, abbab)$
$\vdash_M (q_1, bbab) \vdash_M (q_0, bab) \vdash_M (q_1, ab)$
$\vdash_M (q_1, b) \vdash_M (q_0, \lambda)$

$(q_0, baabbab) \overset{*}{\underset{M}{\vdash}} (q_0, \lambda) \wedge q_0 \in F$

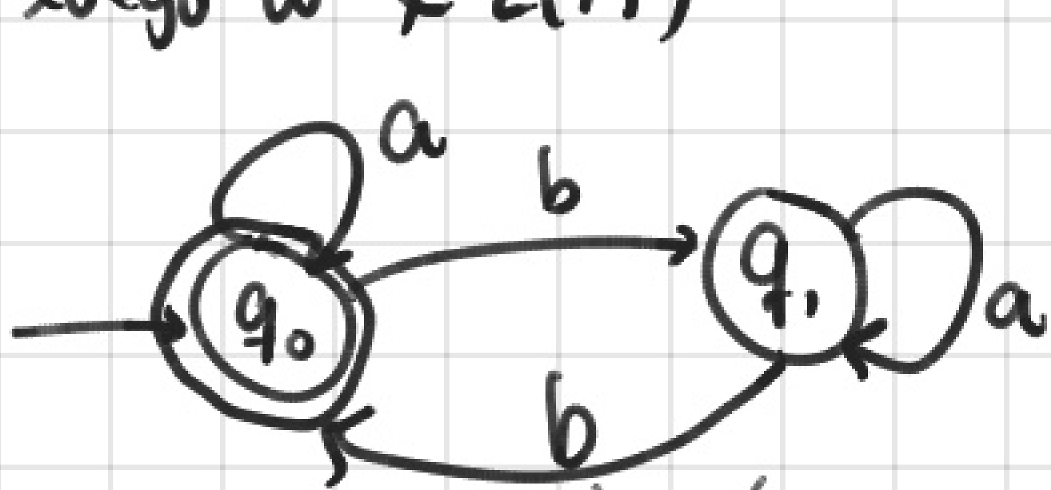
luego $baabbab \in L(M)$

q_2

$w = babb$

$(q_0, babb) \vdash_M (q_1, abb) \vdash_M (q_1, bb) \vdash_M (q_0, b)$
$\vdash_M (q_1, \lambda) \wedge q_1 \notin F$

luego $w \notin L(M)$



$L(M) = \{ w \in \Sigma_b^* / |w|_b = 2k ; k \in \mathbb{N} \}$

Ejercicio

	a	b
$\to q_0$	q_0	q_1
q_1	q_0	q_2
q_2	q_0	q_3
q_3	q_3	q_3

a) Procesar bababb

b) " " baabbba

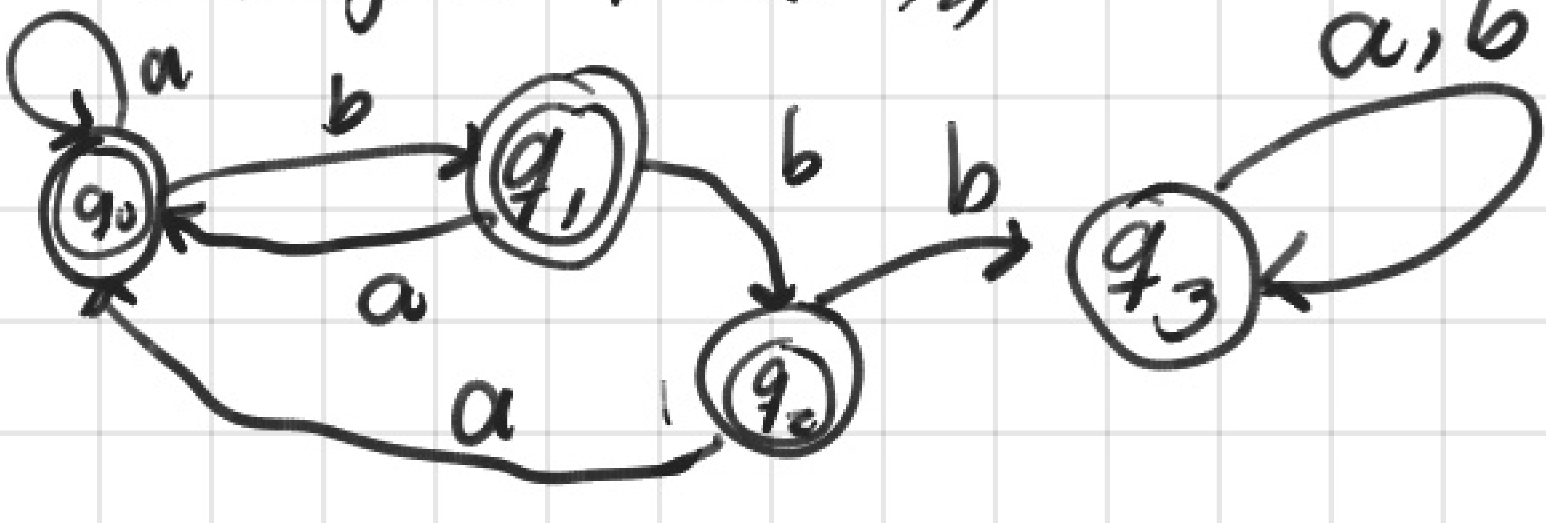
c) escribir $L(M)$ por comprensión

$(q_0, bababb) \vdash_M (q_1, ababb) \vdash_M (q_0, babb)$
$\vdash_M (q_1, abb) \vdash_M (q_0, bb) \vdash_M (q_1, b) \vdash_M$
$(q_2, \lambda) \wedge q_2 \in F$

luego $w \in L(M)$

$(q_0, baabbba) \vdash_M (q_1, aabbba) \vdash_M (q_0, abbba) \vdash_M$
$(q_0, bbba) \vdash_M (q_1, bba) \vdash_M (q_2, ba)$
$\vdash_M (q_3, a) \vdash_M (q_3, \lambda) \wedge q_3 \notin F$

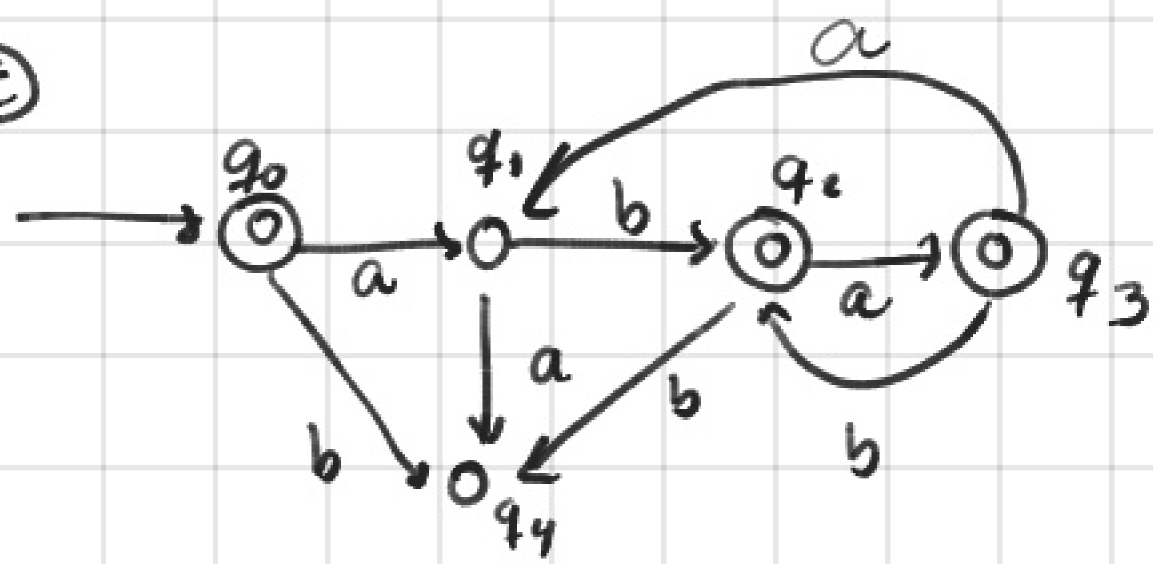
$\therefore$ luego $w \notin L(M)$



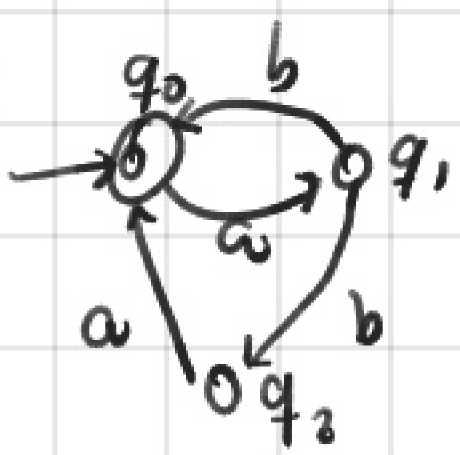
Automatas F. N. Deterministas

$L(\alpha) = \{\lambda, ab, aba, abab, ababa, \dots\}$

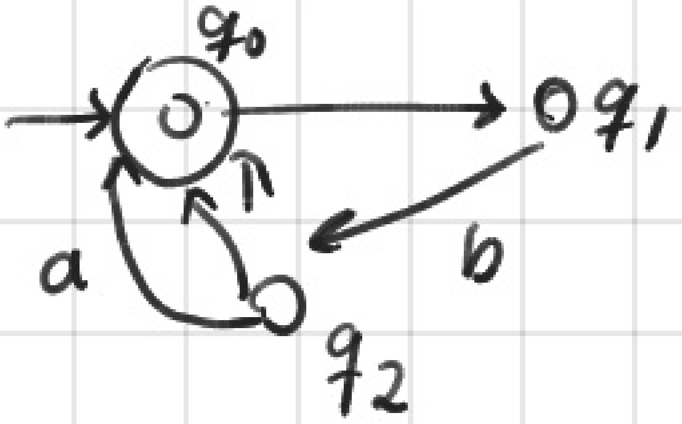
I



II



III



IV

Definición: Un automata finito no deteiministico es una quintubla $M=(K,\Sigma,\Delta,s,F)$ donde

- K: Conjunto finito no vacio (Conjunto de estados)
- Σ: Conjunto finito no vacio (Lenguaje de entrada)
- $s\in K$: es un elemento de K
- $F\subseteq K$: Conjunto de estados finales
- Δ: es un subconjunto finito de $K\times\Sigma^*\times K$ (relación de transición)

un Elemento de Δ:

$$(q, w, q')\in\Delta \Leftrightarrow \quad q \xrightarrow{w} q'$$

Configuración:

Una Configuración es un elemento de $K\times\Sigma^*$

La relación $\vdash_M$, sea $M=(K,\Sigma,\Delta,s,F)$ un AFN y sea (q,w) y (q',w') 2 configuraciones

$$(q,w)\vdash_M (q',w') \quad w=uw' ;\ \text{para algun } u\in\Sigma^*$$
$$\text{y } (q,u,w')\in\Delta$$

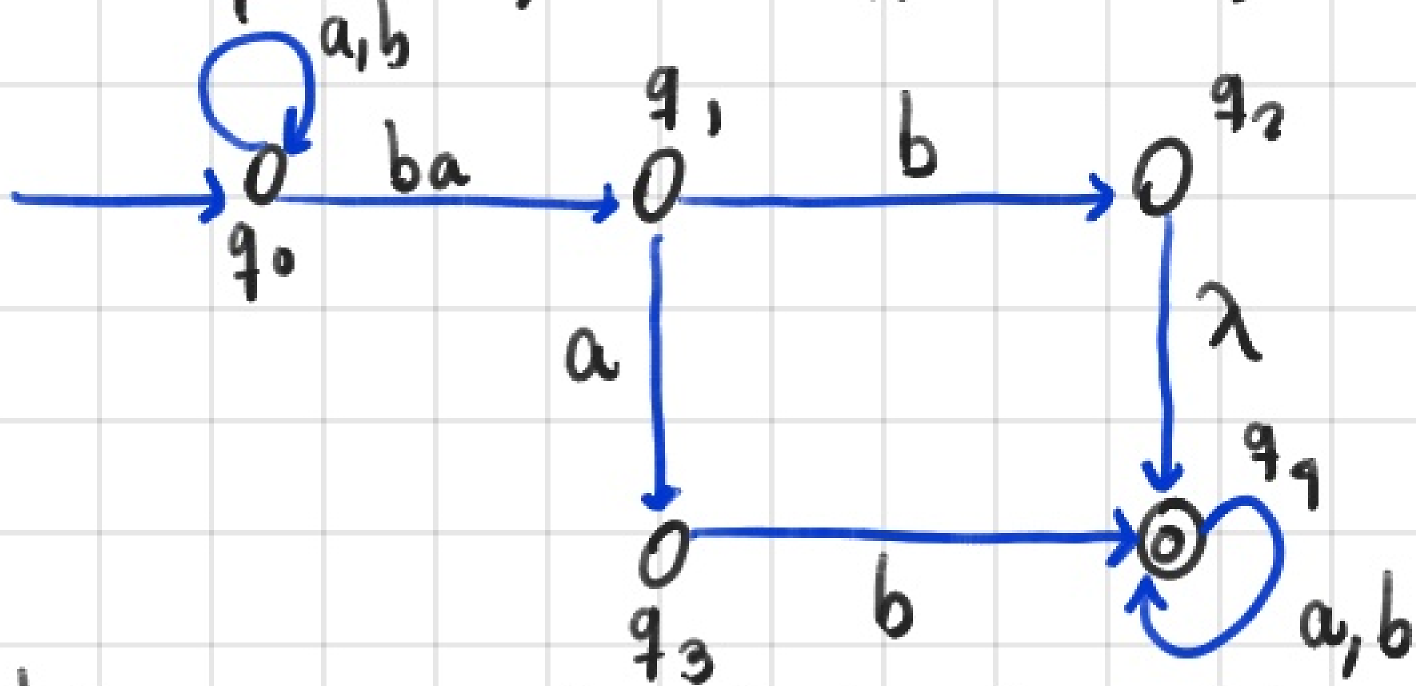
Denotamos por $\vdash_M^*$ a la clausura reflexiva transitiva de $\vdash_M$

si $w \in \Sigma^*$

M acepta $w \Leftrightarrow (s, w) \vdash_M^* (q, \lambda)$; $q \in F$

$L(M) = \{w \in \Sigma^* / M$ acepta $w\}$

$L(M) = \{w \in \Sigma^* / (s, w) \vdash_M^* (q, \lambda)\ ; q \in F\}$



a) Procesar $w = baababaab$ en términos de configuraciones

b) $L(M)$?

sea $(q_0, baababaab) \vdash_M (q_1, ababaab) \vdash_M (q_3, babaab)$

$\vdash_M (q_4, abaab) \vdash_M^* (q_4, \lambda)$; $q_4 \in F$

luego $w \in L(M)$

Ejercicio

Sea $M = (K, \Sigma, \Delta, s, F)$ un AFN

Demostrar:

$$(q, x) \vdash^{*}_{M} (p, \lambda) \Rightarrow (q, xy) \vdash^{*}_{M} (p, y)$$

$p, q \in K, \quad x, y \in \Sigma^{*}$

Propiedades de los lenguajes aceptados por automatas finitos.

Teo

1. Union

2. Concatenación

3. Estrella de Kleen

4. Complementación

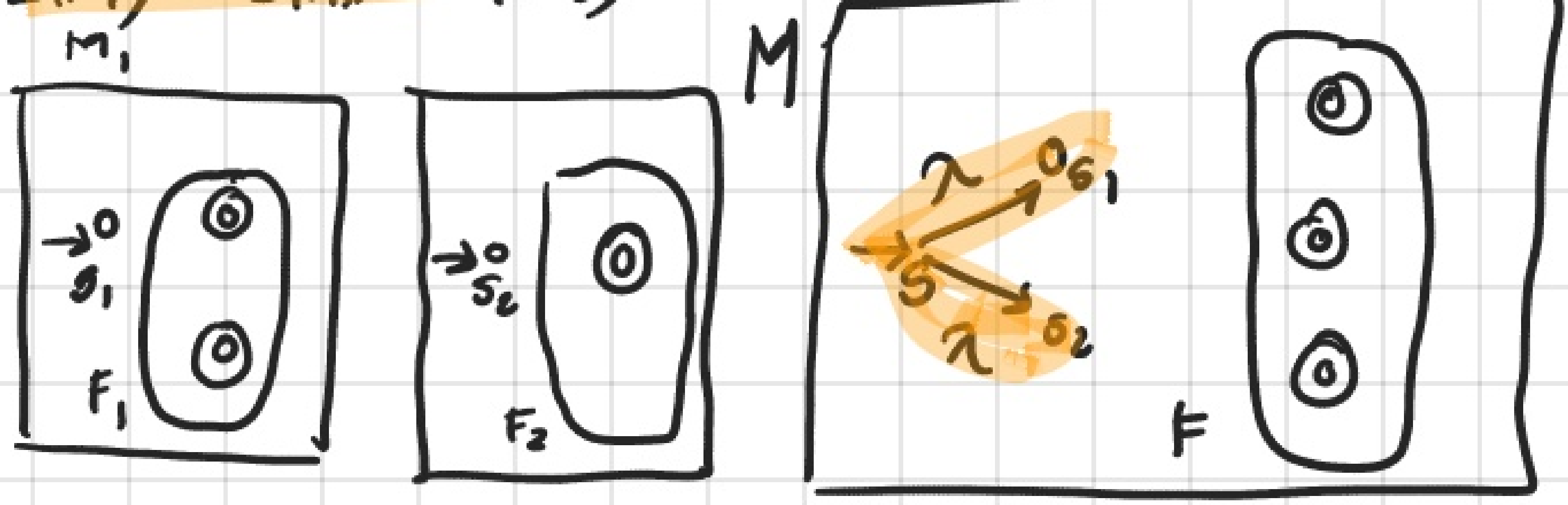
5. Intersección

Prueba

Sea $M_1(K_1, \Sigma, \Delta_1, s_1, F_1)$ y $M_2(K_2, \Sigma, \Delta_2, s_2, F_2)$ AF's que aceptan $L(M_1)$ y $L(M_2)$ respectivamente

Construimos M tal que

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



$M = (K, \Sigma, \Delta, s, F)$

s es un nuevo elemento de K (estado inicial)

$K = K_1 \cup K_2 \cup \{s\}$; $K_1 \cap K_2 = \emptyset$

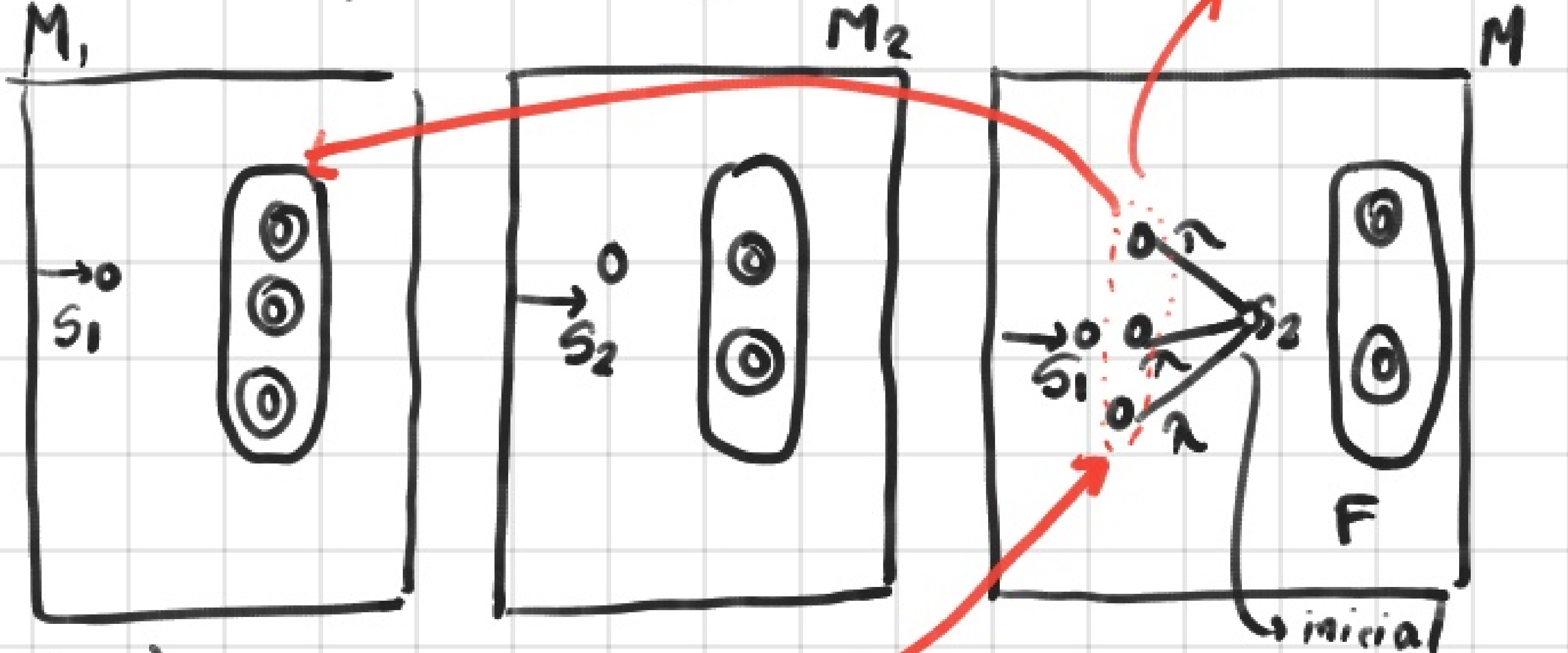
$F = F_1 \cup F_2$

$\Delta = \Delta_1 \cup \Delta_2 \cup \{(s, \lambda, s_1), (s, \lambda_2)\}$

b) Concatenación

Construimos M / $L(M) = L(M_1)L(M_2)$

dejan de ser estados finales



$M = (K, \Sigma, \Delta, s, F)$

$K = K_1 \cup K_2$

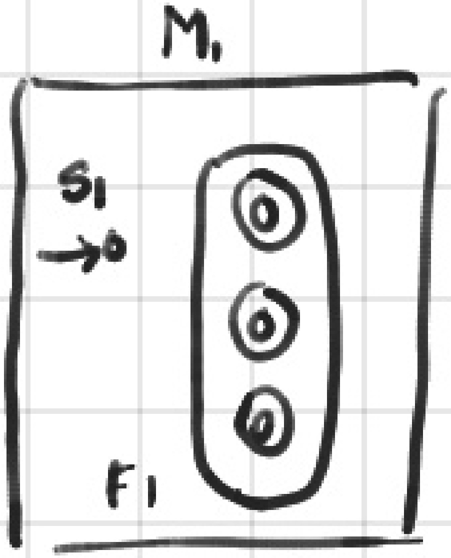
$F = F_2$

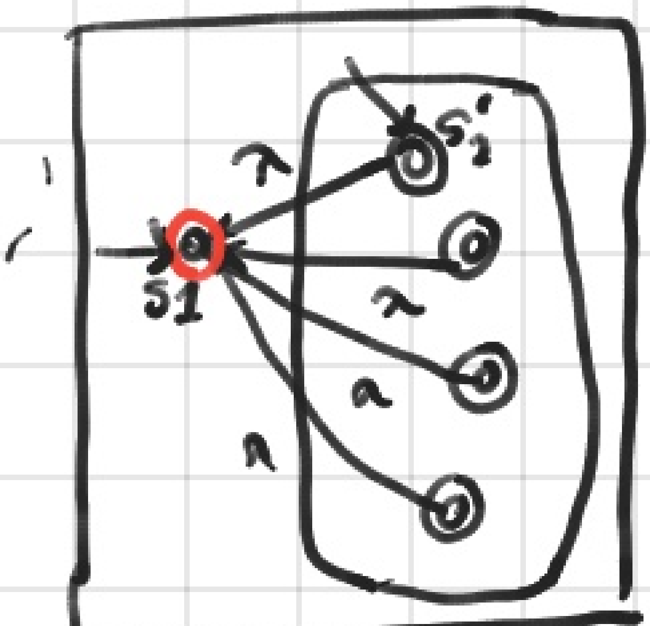
$s = s_1$

$\Delta = \Delta_1 \cup \Delta_2 \cup (F_1 \times \{\lambda\} \times \{s_2\})$

d) Estrella de Kleene

Construimos M tal que $L(M) = L(M_1)^*$





$M = (K, \Sigma, \Delta, s, F)$

s_1 = es un nuevo estado (inicial y final a la vez)

$F = F_1 \cup \{s_1'\}$

$K = K_1 \cup \{s_1'\}$

$\Delta = \Delta_1 \cup (F \times \{\lambda\} \times \{s_1\})$

d) Complementación

Sea $M = \{K, \Sigma, \delta, s, F\}$ un AFD

$\Sigma^* - L(M)$ es aceptado por el automata.

$M = \{K, \Sigma, \delta, s, K - \Sigma\}$ Los F dejan de ser finales y los que eran se vuelven finales.

e) Intersección

Sean L_1 y L_2 2 lenguajes aceptados por automatas finitos, M_1 y M_2

$$L_1 \cap L_2 = \Sigma^* - \left[(\Sigma^* - L_1) \cup (\Sigma^* - L_2) \right]$$

* Sea $\alpha = ba^*b$

Justificando cada paso construir el AFD tal que $L(M) = L(\alpha)$

$\varnothing$

Lenguaje No regulares

Pumping Sea L un lenguaje regular infinito
Entonces existe palabras x, y, z tales que
$y \neq \lambda$ y $xy^nz \in L$ para cada $n \geq 0$

Demostrar:
$L = \{a^n b^n : n \geq 0\}$
No es regular
Prueba RAA
Sea L un lenguaje regular infinito
// aaabbb

$\underbrace{aaa}_{x}\underbrace{aaaaa}_{y}\underbrace{bbbbbbbb}_{z} \in L$

Caso 1: Si y consta exclusivamente de a's

$x = a^p$, $y = a^q$, $z = a^r b^s$

$p, r \geq 0$, $q, s > 0$

$xy^nz \in L, \forall n \geq 0$

$a^p (a^q)^n a^r b^s \in L$, $\forall n \geq 0$

$a^{p+nq+r} b^s \in L$, $\forall n \geq 0$

$a^{p+nq+r} b^s \in L$, Solo para $n = 1$

n=0 aaaaabbbbbbbb
n=1 aaaaaaaabbbbbbbb
n=2 aaaaaaaaaaaaabbbbbbbb
n=3
En el único caso, cuando $n = 1$

Caso 2: Si y consta solo de b's

$x = a^p b^q$, $y = b^r$, $z = b^s$ // aaaaaaaa bbbbbbbb

$p, r > 0$ $q, s \geq 0$

$xy^nz \in L \quad , \forall n \geq 0$
$a^p b^q (b^r)^n b^s \quad , \forall n \geq 0$
$a^p b^{q+rn+s} \quad , \forall n \geq 0$

$n=0$ aaaaaaaabbbbb ✗ No pertenece
$n=1$ aaaaaaaabbbbbbbb ✓ pertenece
$n=2$
$n=3$

Caso 3: y consta de a's y de b's // $\underbrace{aaaaaa}_{x}\underbrace{aabbbb}_{y}\underbrace{bbb}_{z}$

$x=a^p, \ y=a^q b^r, \ z=b^s$
$p, s \geq 0 \quad q, r > 0$
$xy^nz \in L \quad , \forall n \geq 0$
$a^p (a^q b^r)^n b^s, \forall n \geq 0$
$a^p a^{qn} b^{rn} b^s, \forall n \geq 0$
$a^{p+qn} b^{rn+s}, \forall n \geq 0$

$n=0$
$n=1$
$n=2$ aaa $(aabbb)^2$ bb
aaa(aabbb)(aabbb)bb $\notin L$

∴ como supuse que L es un lenguaje regular infinito y no se cumplió :

L es no regular //

E9 $L = \{ a^n : n \text{ es primo} \}$

Demostrar que L es no regular

$a, aa, aaa, aaaaa, \underbrace{a}_{x}\underbrace{aaaa}_{y}\underbrace{aa}_{z}$

Prueba: (R.A.A)

Sea L un lenguaje regular infinito

// si es regular infinito cumple el teorema, $\exists x, y, z$, $xy^nz \in L, \forall n \geq 0$

$x = a^p, \; y = a^q, \; z = a^r$

$p, r \geq 0 \; , \; q > 0$

$xy^nz \in L, \forall n \geq 0$

$a^p (a^q)^n a^r, \forall n \geq 0$

$a^p a^{qn} a^r \in L, \forall n \geq 0$

$a^{p+qn+r} \in L, \forall n \geq 0$

Si $n = p + 2q + r + 2$?

$a^{p + q(p+2q+r+2)+r}$

$a^{p + pq + 2q^2 + rq + 2q + r}$

$a^{p(1+q) + 2q(q+1) + r(q+1)}$

$a^{p(1+q) + (q+1)(2q+r)}$

$a^{\underbrace{(q+1)}(p+2q+r)}$

es otra factorizacion distinta de el mismo o 1.

Significa que q es 0 o mayor y rompe con $n \geq 0$

Pues no cumple cuando $n = p + 2q + r + 2$

$\therefore$ <u>L no es regular</u>

Gramaticas Regulares

Una gramatica libre de contexto

$G = (V, \Sigma, R, S)$ es una gramática ssi

$$R \subseteq (V-\Sigma) \times \Sigma^*\left((V-\Sigma) \cup \{\lambda\}\right)$$

$\Sigma^*(\{\lambda\}) \to ($, $\to$ solo minusculas)

$\Sigma^*(V-\Sigma) \to ($, $\to$ minusculas terminando con una mayuscula.

② $G = (V, \Sigma, R, S)$ donde $V = \{S, A, B, a, b\}$

$\Sigma = \{a, b\}$

$R = \{S \to bA,\ S \to aB,\ A \to abaS,\ B \to babS,\ S \to \lambda\}$

Teorema: // lenguajes regulares generados por gramáticas regulares.

Un lenguaje es regular ssi es generado por una gramática regular.

<u>Prueba.</u> // Demostración condicional

$P \Rightarrow Q$ // si es regular existe un automata que lo acepta

Sea $M=(K,\Sigma,\delta,s,F)$ un AFD construimos

$G=(V,\Sigma,R,S)$ donde:

$V=\Sigma \cup K$

$S=s$

$R=\{q\to ap : \delta(q,a)=p\} \cup \{q\to \lambda : q\in F\}$

Sea G una GR construimos $M=(K,\Sigma,\Delta,s,F)$ donde:

$K=(V-\Sigma)\cup\{f\}$ donde f es un nuevo elemento

$s=S$

$F=\{f\}$

$\Delta=\{(A,\omega,B): A\to\omega B,\ A,B\in(V-\Sigma),\ \omega\in\Sigma^*\}\cup$

$\{(A,\omega,f): A\to\omega,\ A\in(V-\Sigma),\ \omega\in\Sigma^*\}$

Dada la gramatica

$G=(V,\Sigma,R,S)$

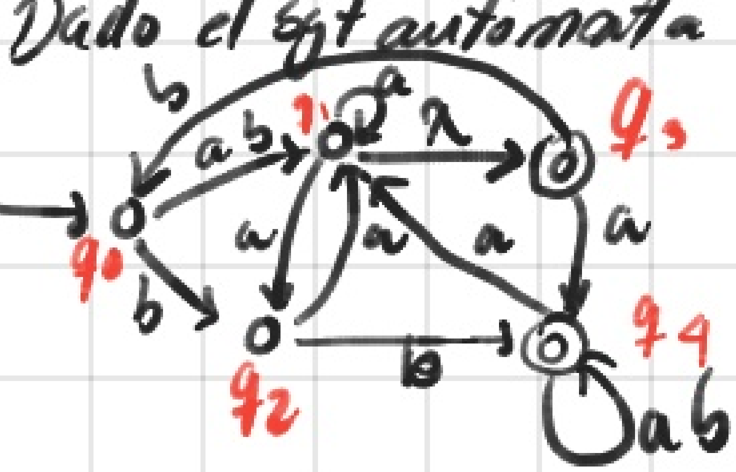
$V=\{S,A,B,a,b\}$

$\Sigma=\{a,b\}$

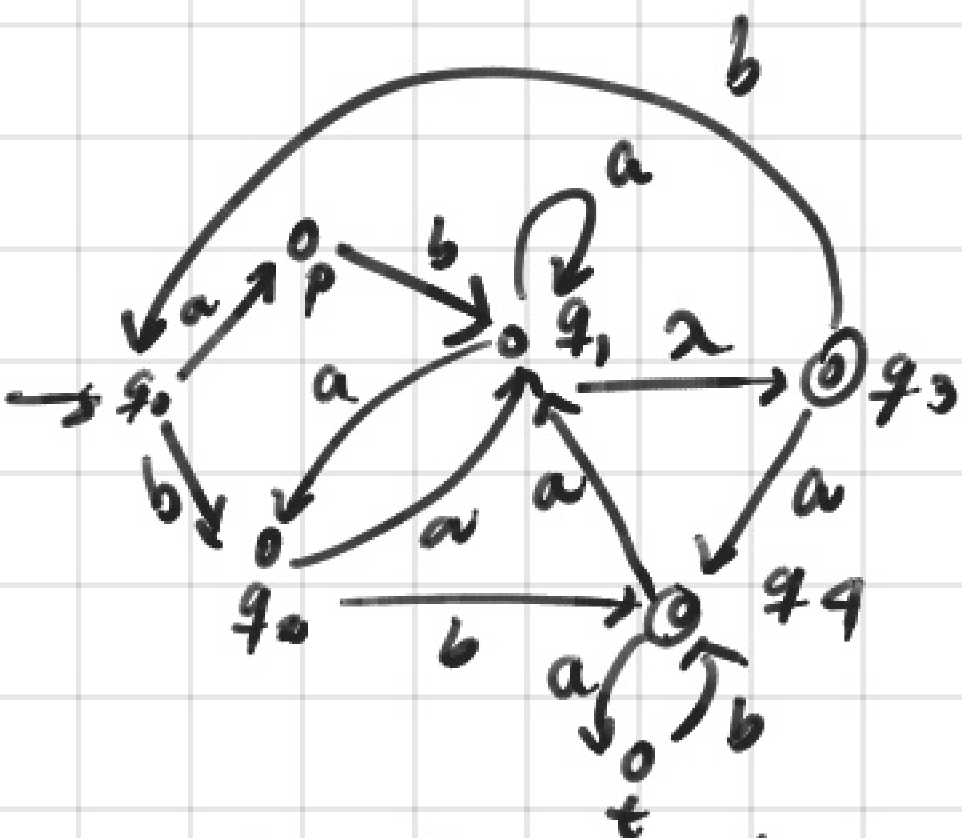
$R=\{S\to bA,\ S\to aB,\ A\to abaS,\ B\to babS,\ S\to\lambda\}$

construir el AFN tal que $L(M)=L(G)$

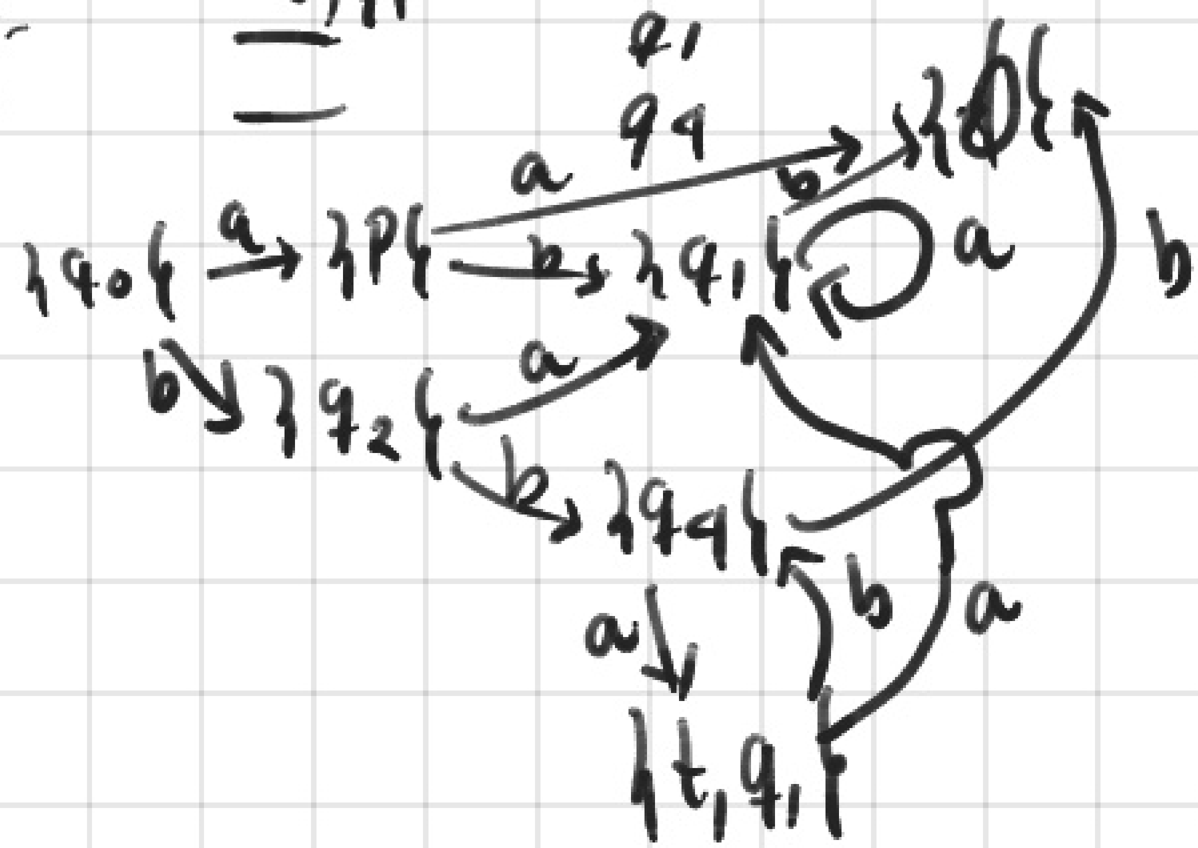
(2) Dado el sgt autómata



Construir la gramática G.



	λ	a	b
q_0	q_0	p	q_2
q_1		q_1	—
q_2		q_1	q_4
q_3		q_4	q_0
q_4		t, q_1	—
p		—	q_1
t		—	q_4



Automatas con Pila

Teo. L =

sea $G = (V, \Sigma, R, S)$ una gramática L.C

construimos $M = (\{p, q\}, \Sigma, V, \Delta, p, \{q\})$

1. $(p, \lambda, \lambda), (q, S)$
2. $((q, \lambda, A), (q, x)) \; \forall A \to x \in R$
3. $((q, a, a), (q, \lambda)) \; \forall a \in \Sigma$

Ej $G = (V, \Sigma, R, S)$

$V = \{S, a, b, c\}, \Sigma = \{a, b, c\}$

$R = \{S \to aSa, S \to bSb, S \to c\}$

a) Construir M

b) Procesar abbcbba

a) $\Sigma = \{a, b, c\}$ $V = \{S, a, b, c\}$

$$R = \left\{ \begin{array}{l} ((p, \lambda, \lambda), (q, S)), ((q, \lambda, S), (q, aSa)) \\ ((q, \lambda, S), (q, bSb)), ((q, \lambda, S), (q, c)), ((q, a, a), (q, \lambda)) \\ ((q, b, b), (q, \lambda)), ((q, c, c), (q, \lambda)) \end{array} \right\}$$

Propiedades de los lenguajes libres de Contexto

Teo: Los LLC son cerrados bajo la Union, concatenación y estrella de Kleene.

Prueba:

Sea $G_1 = \{V_1, \Sigma_1, R_1, S_1\}$

$G_2 = \{V_2, \Sigma_2, R_2, S_2\}$ 2 GLC y sin perdida de operacidad asumimos que $V_1 - \Sigma_1$ y $V_2 - \Sigma_2$ son disjuntos

UNION

$$G = \{V_1 \cup V_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, R_1 \cup R_2 \cup \{S \to S_1, S \to S_2\}, S\}$$

CONCATENACION

$$G = \{V_1 \cup V_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, R_1 \cup R_2 \cup \{S \to S_1 S_2\}, S\}$$

ESTRELLA DE KLEENE

$$G = \{V_1, \Sigma_1, R_1 \cup \{S_1 \to \lambda, S_1 \to S_1 S_1\}, S_1\}$$

La intersección de un LLC y un LR es un LLC

Sea G una GLC entonces existe un K que depende de G tal que cualquier palabra w en L (G) de longitud mayor que K se puede se escribir $w = uvxyz$ de tal manera ya sea v o y es no vacía. y $uv^n x y^n z \in L(G)$ $\forall n \geq 0$

Demostrar que:

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

No es LC.

Los LLC no son cerrados bajo la intersección o complementación

Sean (q_1, w_1, a, u_1) y (q_2, w_2, a_2, u_2) 2 config.

$(q_1, w, a_1, u_1) \vdash_M (q_2, w_2, a_2, u_2)$ ssi :

Para algun $b \in \Sigma \cup \{L, R\}$

$\delta(q_1, a_1) = (q_2, b)$ y si

1) $b \in \Sigma$, $w_1 = w_2$, $u_1 = u_2$ y $a_2 = b$

2) $b = L$, $w_1 = w_2 a_2$ y

a) $u_2 = a_1 u_1$; si $a_1 \neq \#$ ó $u_1 \neq \lambda$ ó

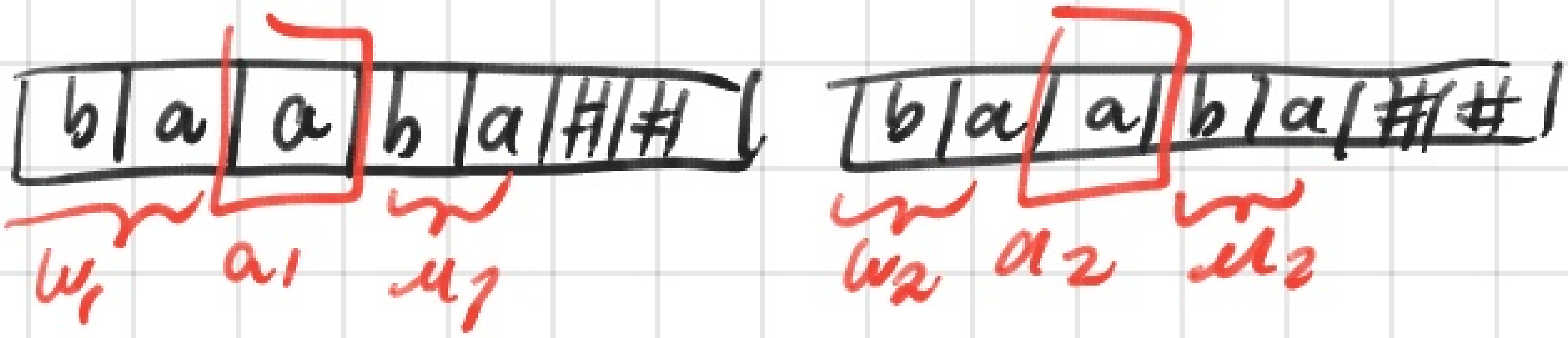
b) $u_2 = \lambda$; si $a_1 = \#$ y $u_1 = \lambda$

3) $b = R$, $w_2 = w_1 a_1$ y

a) $u_1 = a_2 u_2$

b) $u_1 = u_2 = \lambda$ y $a_2 = \#$

Demostrar por $\vdash^*$ ala clausura reflexiva transitiva de $\vdash_M$



- En el caso 1 solo reescribe a_1 / $a_2 = b$
- En el caso 2 recorre hacia la izquierda
- En el caso 3 recorre hacia la derecha.

CÁLCULOS CON MT

Convenio:

- Entrada rodeada de # (blancos)
- Cabeza lectora al inicio en el blanco de la derecha
- Inicia: estado inicial

Def: Sean Σ_0, Σ_1 alfabetos que no contienen el símbolo # y sea f una función de Σ_0^* en Σ_1^*. Una MT $M=(K,\Sigma,\delta,s)$ se dice que computa f si $\Sigma_0 \Sigma_1 \subseteq \Sigma$ y $\forall w \in \Sigma_0^*$ si $f(w) = u \Rightarrow$ $(s, \# w \underline{\#}) \vdash_M^* (h, \# u \underline{\#})$ si tal máquina existe se dice que f es Turing computable

(Ej) $\Sigma_0 = \Sigma_1 = \{a, b\}$ $f: \Sigma_0^* \to \Sigma_1^*$
tal que $\forall w \in \Sigma_0^*$ $f(w) = \overline{w}$ (cambia as por bs y vice versa)

es F turing computable?

$M = (K, \Sigma, \delta, q_0)$; $K = \{q_0, q_1, q_2\}$

$\Sigma = \{a, b, \#\}$

	a	b	#
q_0	(q_1, L)	(q_1, L)	(q_1, L)
q_1	(q_0, b)	(q_0, a)	(q_2, R)
q_2	(q_2, R)	(q_2, R)	$(h, \#)$

procesar:
i) bba
ii) λ

sol. $(q_0, \#bba\underline{\#}) \longmapsto (q_1, \#bb\underline{a})$
$\vdash_M (q_0, \#bb\underline{b}) \longmapsto (q_1, \#b\underline{b})$
$\vdash (q_0, \#b\underline{a}b) \vdash (q_1, \#\underline{b}ab)$
$\vdash (q_0, \#\underline{a}ab) \vdash (q_1, \underline{\#}aab)$
$\vdash (q_2, \#\underline{a}ab) \vdash (q_2, \#a\underline{a}b)$
$\vdash (q_2, \#aa\underline{b}) \vdash (q_2, \#aab\underline{\#})$
$\vdash (h, \#aab\underline{\#})$

$(q_0, \#bba\underline{\#}) \vdash_M^* (h, \#aab\underline{\#})$

Completar ii) $\Rightarrow$ $\therefore$ f es turing computable

En general:

$f(\Sigma_0^*)^K \longrightarrow \Sigma_1^*$ tal que

$f(w_1, w_2, \ldots, w_K) = u$ entonces existe

una MT tal que

$(s, \#w_1\#w_2\# \ldots \#w_K\underline{\#}) \vdash (h, \#u\underline{\#})$

entonces f es turing computable

Para funciones $f: \mathbb{N} \to \mathbb{N}$ // $[I^n = n]$

$f: \mathbb{N} \to \mathbb{N}$ es turing computable si M computa la función $f: \{I\}^* \to \{I\}^*$ donde $f'(I^n) = I^{f(n)}$ $\forall n \in \mathbb{N}$

- En general $f: \mathbb{N}^K \to \mathbb{N}$ es turing computable si M computa la función $f'(\{I\}^*)^K \to \{I\}^*$ donde $f'(I^{n_1}, I^{n_2}, I^{n_3} \ldots, I^{n_K}) = I^{f(n_1, n_2, n_3, \ldots n_K)}$

$\forall n_1, n_2, n_3 \in \mathbb{N}$,

Ej. $f: \mathbb{N} \to \mathbb{N}$ tal que

$f(n) = n+1$

	I	#
q_0	(I, R)	(q_0, I)

Procesar

a) III

b) λ

¿Es Turing computable?

Extensión de Máquinas de Turing:

Los sgt modelos de MT son igualmente poderosos:

1) La máquina de Turing estandar
2) " " " de varias cabezas
3) La máquina " " de varias pistas
4) " " " " " de varias cintas
5) " " " bidireccional
6) Automata con pila con 2 pilas
7) IBM Mainframe