

BLOQUE 3 TC

Def.: Máquina de Turing, consta de 7 elementos, $M = (Q \Sigma, \Gamma, s, B, F, \delta)$

Q es un conj. finito de estados.

Σ alfabeto de los símbolos de entrada.

Γ alfabeto de los símbolos de la cinta.

$s \in Q$ estado inicial

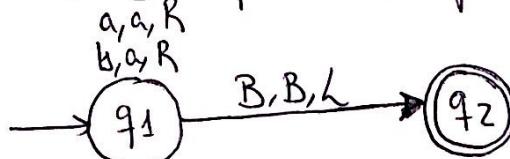
$B \in \Gamma$ símbolo en blanco

$F \subseteq Q$ conj. de estados finales

$\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ función de transición. L (izquierdo) y R (derecha).

Ejemplo: $Q = \{q_1, q_2\}$ $s = q_1$ $\delta(q_1, a) = (q_1, a, R)$
 $\Sigma = \{a, b\}$ $F = \{q_2\}$ $\delta(q_1, b) = (q_1, b, R)$
 $\Gamma = \{a, b, B\}$ $B = B$ $\delta(q_1, B) = (q_2, B, L)$

Al procesar a, b reemplaza los "b" por "a"

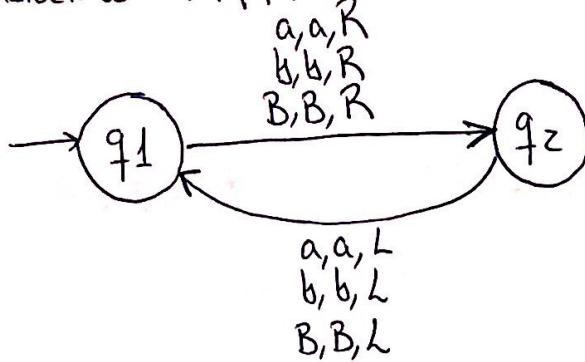


Si le pasamos abba:

$(q_1, \underline{abba}) \xrightarrow{} (q_1, ab\underline{ba}) \xrightarrow{} (q_1, aa\underline{ba}) \xrightarrow{} (q_1, aaa\underline{a}) \xrightarrow{} (q_1, aaaa\underline{B}) \xrightarrow{} (q_2, aaaa\underline{B})$

$\vdash (q_2, aaaa\underline{B})$

Considerando la MT:

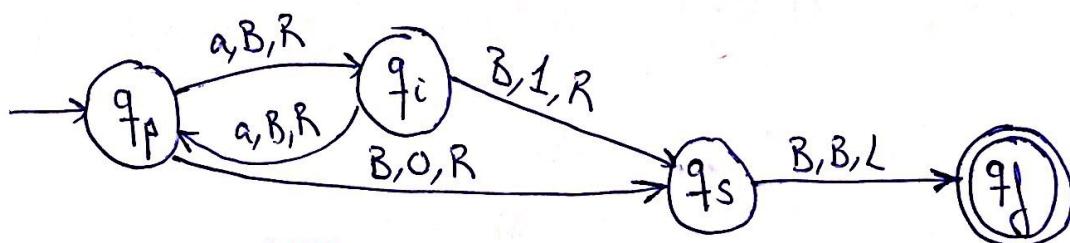


Es una MT que nunca parará.

$q_1 ab \dots t^* \infty$

Def. Un lenguaje aceptado por una MT se denomina lenguaje recursivamente enumerable, todos sus códigos pueden ser enumerados por una MT.

Ejercicio: Construir una MT tal que $f(n) = \begin{cases} 0, & \text{si } n \text{ es par} \\ 1, & \text{si } n \text{ es impar} \end{cases}$



Construcción de MT:

$R \rightarrow$ derecha

$L \rightarrow$ izquierda

$RB \rightarrow$ busca el 1^{er} blanco que hay a la derecha

$LB \rightarrow$ busca el 1^{er} blanco que hay a la izquierda

$R\bar{B} \rightarrow$ " " símbolo NO blanco a la derecha

$L\bar{B} \rightarrow$ " " a la izquierda

$S_R \rightarrow$ "shift right" mueve la cinta a la derecha de todo $\underline{B}wB \Rightarrow B\underline{B}w$

$S_L \rightarrow$ "shift left" hace suprimir del primer elemento y se mueve hacia la pos de ese elemento

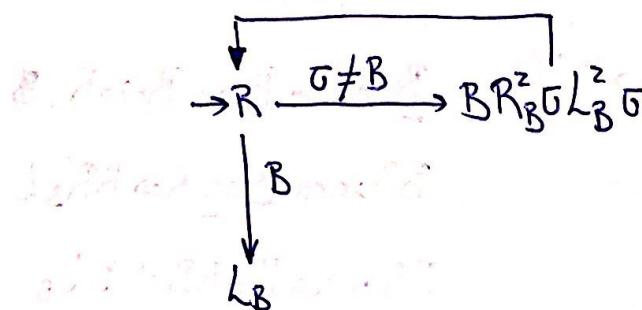
Suponiendo $\underline{\sigma_1 \sigma_2 w}B \Rightarrow \underline{\sigma_2} wBB$

Ejercicio: Efectos de SR sobre BaaBbbB y S^2 sobre BaaBbbBccB?

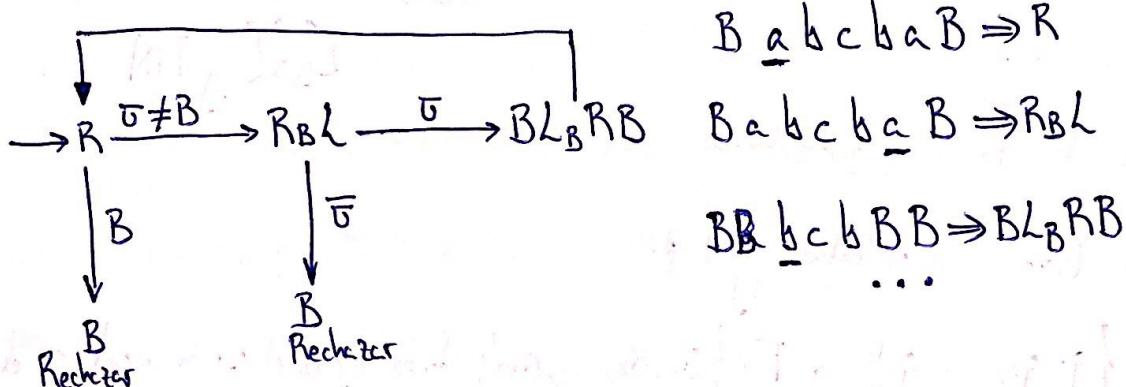
1º BBaaBbbB

2º BBaaBbbBccB y después BBBaaBbbccB

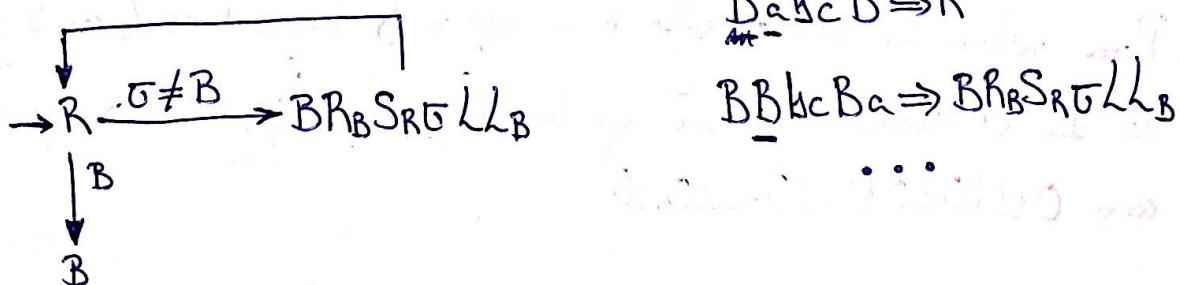
Ejercicio: Máquina de copia. BwB en BwBwB



Ejercicio: MT que acepte $\{w \mid w = w'\}$ situación inicial es BwB



Ejercicio: MT que transforme BwB en Bw' B



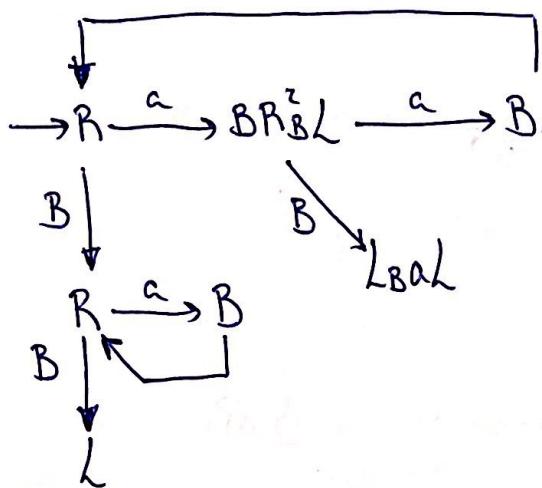
BabcB ⇒ R

BBbcBa ⇒ BRBSRG[L_B]

Ejercicio:

$$n \div m = \begin{cases} n-m, & \text{si } n \geq m \\ 0, & \text{si } n < m \end{cases}$$

MT que transforme $\underline{B}a^nB a^mB$ en $\underbrace{\underline{B}a^{n-m}B}_{n \geq m} \circ \underline{BB}$ si $n < m$



5-2

BacacaBaaB $\Rightarrow R, B$

BBacacaBaaB $\Rightarrow BR_B^z L$

BBacacaBaBB $\Rightarrow BL_B^z B$

BBBacacaBaBB $\Rightarrow BR_B^z L$

BBBBacacaBBBB $\Rightarrow BL_B^z B$

LBaL y FIN

Máquina de Turing Universal

- MT tiene un único estado final p.
- $Q = \{q_1, q_2, \dots, q_n\}$ y $T = \{\overline{0}_1, \overline{0}_2, \dots, \overline{0}_m\}$ donde q_1 es el estado inicial, q_2 el único estado final y $\overline{0}_1$ es el blanco.
- Para codificar una MT es suficiente con codificar δ . Representamos L con 1 y R con 11. Utilizaremos el 0 como separador. $\delta(q_3, \overline{0}_1) = (q_4, \overline{0}_3, L)$ se representa como: 01110101110111010

LENGUAJES RECURSIVAMENTE ENUMERABLES

$M = (Q, \Sigma, S, F, \delta)$ un AFD. Podemos construir $M' = (Q', \Sigma', \Gamma, S', B, F', \delta')$

una MT tal que $L(M) = L(M')$ como sigue:

$$Q' = Q \cup \{q'\} \quad \Sigma' = \Sigma \quad \Gamma = \sum \cup \{B\} \quad S' = S \quad F' = \{q'\}$$

$$\delta'(q, \sigma) = (\delta(q, \sigma), \sigma, R), \forall q \in Q, \forall \sigma \in \Sigma'$$

$$\delta'(q, B) = (q', B, S), \forall q \in F.$$

Teorema: Si L es regular, entonces L es también un lenguaje recursivo.

Teorema: Si L es independiente del contexto $\Rightarrow L$ es también un lenguaje recursivo.

Teorema: Si L_1 y L_2 son recursivos $\Rightarrow L_1 \cap L_2$ también lo es.

Ejercicio: L_1 y L_2 son lenguajes recursivos $\Rightarrow L_1 \cup L_2$ también.

Dadas M_1 y M_2 dos MT's y tales que $L(M_1) = L_1$ y $L(M_2) = L_2$, construiremos una nueva MT M de 2 cintas, de forma que:

- w se copia en ambas

- M simula a M_1 con w en la 1^{ta} cinta hasta que M_1 pere

- M_1 pere en estado final $\Rightarrow M$ lo mismo

- M_1 pere en no final $\Rightarrow M$ simula a M_2 con w sobre la segunda cinta.

- M_2 pere en estado final $\Rightarrow M$ hace lo mismo.

- M_2 pere en un no final $\Rightarrow M$ también.

$$w \in L(M) \iff w \in L(M_1) \circ \text{ bien } w \in L(M_2)$$

$$L(M) = L(M_1) \cup L(M_2) \Rightarrow L_1 \cup L_2 \text{ es recursivo!}$$

Teorema: Si L recursivo, $\sum^* - L$ también lo es.

Teorema: \exists un L recursivamente enumerable L para el cual $\sum^* - L$ no es recursivamente enumerable.

Teorema: Si L es recursivamente enumerable para el cual $\underbrace{\sum^* - L}_{\text{complementario}}$ también lo es $\Rightarrow L$ es recursivo. y su complementario también.

Teorema: L es recursivamente enumerable $\iff L$ es enumerado por alguno M.T.

Lenguajes sensibles al contexto y la jerarquía de Chomsky

Def.- Una gramática sensible al contexto es aquella en la que todas las reglas son de la forma $\alpha \rightarrow \beta$, donde $\alpha, \beta \in (N \cup \Sigma)^*$ y $|\alpha| \leq |\beta|$

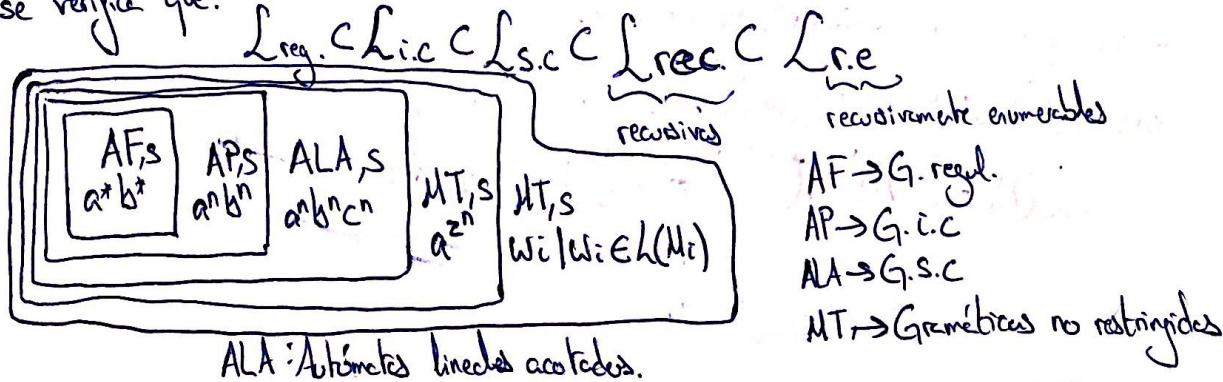
En 1959 Chomsky clasificó las gramáticas en 4 tipos:

- Tipo 0: No restringidas (r.e)
- Tipo 1: Sensibles al contexto (s.c)
- Tipo 2: Independientes del contexto (i.c)
- Tipo 3: Regulares (reg.)

Todo lenguaje L es también de tipo $i-1$, siendo cada uno de estos inclusiones una inclusión propia. Esto es lo que se conoce como la jerarquía de Chomsky.

Teorema de la Jerarquía: Siendo L_x la representación del conjunto de lenguajes de tipo x ,

se verifica que:



El problema de la parada

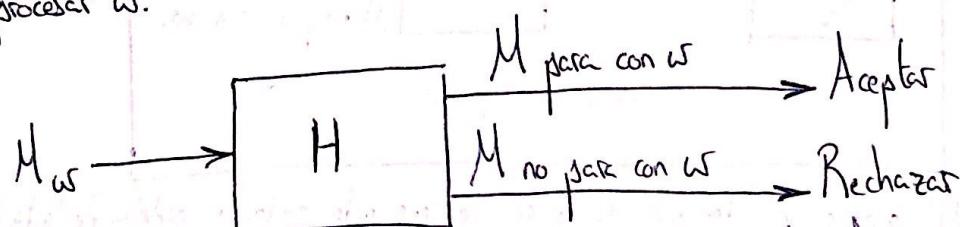
Una función f es Turing Computable si existe una MT que computa $f(w)$ para toda cadena w perteneciente al dominio de f .

Def. - Un problema de decisión es resoluble o decidable si \exists un algoritmo capaz de responder sí o no a cada uno de los casos de dicho problema. Si dicho algoritmo \nexists es irresoluble.

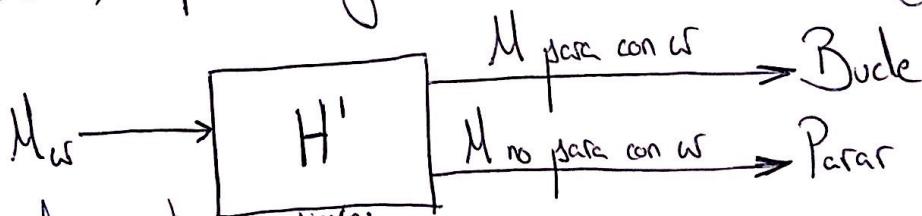
Problema de la Parada: Sea M una máquina de Turing arbitraria con alfabeto de entrada Σ y sea w una cadena de Σ^* . ¿Pasa M con w como cadena de entrada?

Teorema: Es irresoluble, demostración:

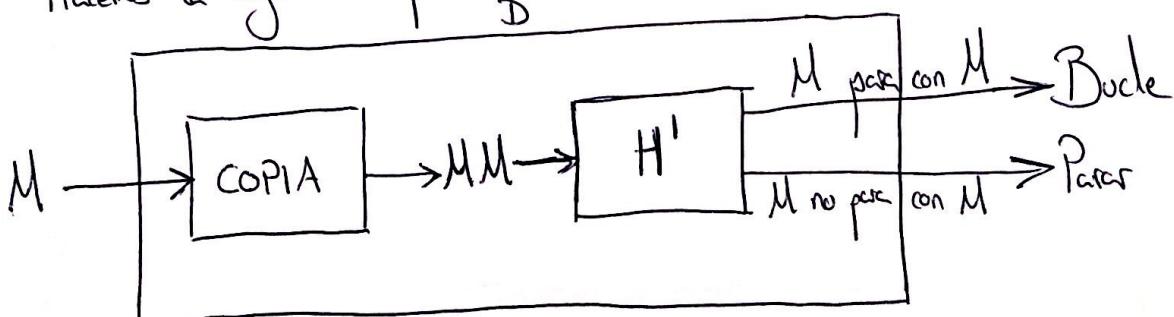
- Supongamos que es resoluble
- \exists una MT H (Halting), similar a la M_H , que recibe como entrada la codificación de una MT M y de una cadena w , y que es capaz de determinar si M pasa al procesar w .



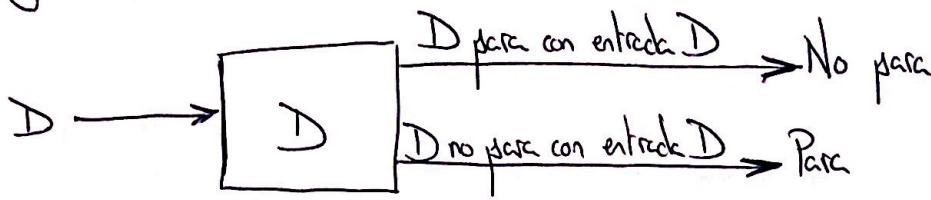
Entonces, podemos modificar H para crear otra MT H' de la siguiente forma:



Haremos la siguiente composición:



y llegamos a que:

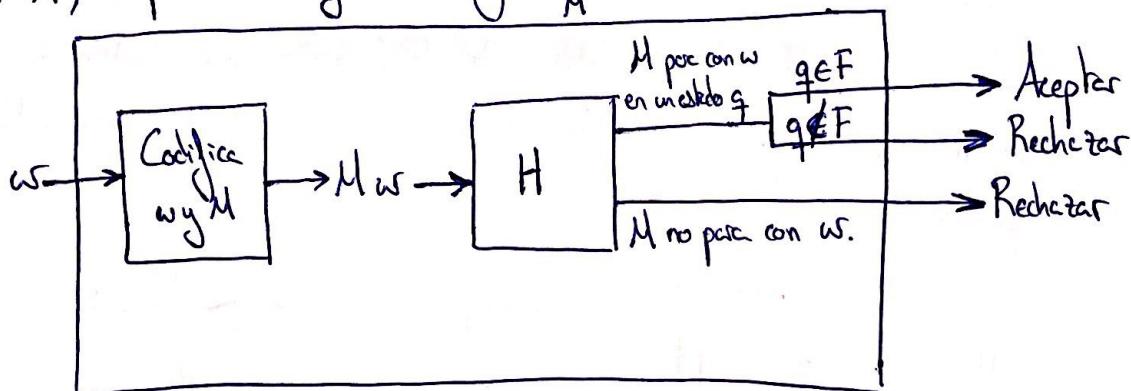


Lo cual es una contradicción, viene del hecho de haber supuesto que era soluble el Problema de la Pascua.

Ejercicio: Demuestre que si el problema de la pascua se pudiere resolver, entonces todo lenguaje recursivamente enumerable sería recursivo.

Sea L un lenguaje recursivamente enumerable, sea M una MT; $L(M) = L$ y sea H

la MT que supuestamente resuelve el problema de la pascua. Entonces construimos una nueva MT M' , a partir de M y H , como sigue:



La MT M' acepta el lenguaje L y se detiene ante cualquier cadena de entrada w . Por tanto, L sería recursivo.

Problema de correspondencia de Post o PCP

Def.- 3 elementos:

- Σ alfabeto

- 2 conjuntos A y B de cadenas Σ^+ , donde ambos conjuntos tienen el mismo cardinal, es decir, $A = \{u_1, u_2, \dots, u_k\}$ y $B = \{v_1, v_2, \dots, v_k\}$

Una sol. es una secuencia de índices i_1, i_2, \dots, i_n tal que

$$u_{i_1}u_{i_2}\dots u_{i_n} = v_{i_1}v_{i_2}\dots v_{i_n}$$

Si $\Sigma = \{a, b\}$, $A = \{a, aba, aab\}$ y $B = \{aaa, ab, b\}$, la solución al SCP viene dada mediante la secuencia de índices 2, 1, 1, 3, ya que

$$u_2u_1u_1u_3 = v_2v_1v_1v_3 = abaaa.aaaab.$$

Representación en "fiches de dominó":

u _i	⇒	a	aba	ab
v _i		aaa	ab	b
		1	2	3

Con los índices 2, 1, 1, 3:

aba	a	a	ab
ab	aaa	aaa	b

Sin embargo, el SCP siguiente:

ab	bac	aba
aba	aa	bac
1	2	3

Solo podemos empezar por la ficha 1, le podríamos seguir la 3, pero...

ab	aba
aba	bac
1	3

La siguiente ficha tiene que ser la 3 \Rightarrow por

ab	aba	aba
aba	bac	bac
1	3	3

lo que nunca acabaría y el razonamiento sería infinito. El SCP no tiene solución.

Def.- El PCP consiste en determinar si el SCP arbitrario tiene solución o no.

Problemas no decidibles en LIC's

Sea $C = \{(u_1, v_1), (u_2, v_2), \dots, (u_n, v_n)\}$. Podemos construir 2 GIC's: G_u y G_v

$$G_u: N_u = \{S_u\}$$

$$\sum_u = \{\sum_c u_i | i=1, 2, \dots, n\}$$

$$P_u = \{S_u \rightarrow u_i | S_u \rightarrow u_i | i=1, 2, \dots, n\}$$

$$G_v: N_v = \{S_v\}$$

$$\sum_v = \sum_c v_i | i=1, 2, \dots, n$$

$$P_v = \{S_v \rightarrow v_i | S_v \rightarrow v_i | i=1, 2, \dots, n\}$$

El SCP C tiene solución si \exists una secuencia $i_1i_2\dots i_k$ tal que:

$$U_{i_1}U_{i_2}\dots U_{i_k} = V_{i_1}V_{i_2}\dots V_{i_k}$$

En este caso, G_U y G_V realizan las siguientes derivaciones:

$$S_U \xrightarrow{*} U_{i_1}U_{i_2}\dots U_{i_k}i_k\dots i_1$$

$$S_V \xrightarrow{*} V_{i_1}V_{i_2}\dots V_{i_k}i_k\dots i_1$$

donde $U_{i_1}U_{i_2}\dots U_{i_k}i_k\dots i_1 = V_{i_1}V_{i_2}\dots V_{i_k}i_k\dots i_1$. $L(G_U) \cap L(G_V)$ es no vacía.

Y de manera inversa, si $w \in L(G_U) \cap L(G_V) \Rightarrow w$ consta de una cadena $w' \in \Sigma^*$ seguida de una secuencia $i_k\dots i_1$. Y la cadena $w' = U_{i_1}U_{i_2}\dots U_{i_k} = V_{i_1}V_{i_2}\dots V_{i_k}$ es solución para C.

Ejercicio:

SCP:

a	abaaa	ab
aaa	ab	b
1	2	3

Solución:

abaaa	a	a	ab
ab	aaa	aaa	b

$$G_U: S_U \xrightarrow{1} a S_{U1} \mid a \xrightarrow{2} | abaaa S_{U2} \mid abaaa \xrightarrow{3} a S_{U3} \mid ab \xrightarrow{4} 3$$

$$G_V: S_V \xrightarrow{1} a a a S_{V1} \mid a a a \xrightarrow{2} | ab S_{V2} \mid ab \xrightarrow{3} b S_{V3} \mid b \xrightarrow{4} 3$$

$$\text{Solución: } S_U \xrightarrow{3} abaaa S_{U2} \xrightarrow{1} abaaa S_{U1} \xrightarrow{2} aba a a a S_{U11} \xrightarrow{6} \underline{aba a a a b} 3 1 1 2,$$

$$S_V \xrightarrow{3} ab S_{V2} \xrightarrow{1} ab a a a S_{V1} \xrightarrow{2} ab a a a a a S_{V11} \xrightarrow{6} \underline{ab a a a a a b} 3 1 1 2,$$

Nota: Gramáticas NO ambiguas. (ej cadena "abaaa" en G_U).

$$S_U \xrightarrow{4} \underline{abaaa}, 2$$

$$S_U \xrightarrow{5} ab S_{U3} \xrightarrow{1} ab a S_{U13} \xrightarrow{2} ab a a a S_{U113} \xrightarrow{3} \underline{ab a a a}, 1 1 1 3,$$

Teorema: No existe ningún algoritmo que determine si los lenguajes generados por 2 GIC's son disjuntos.

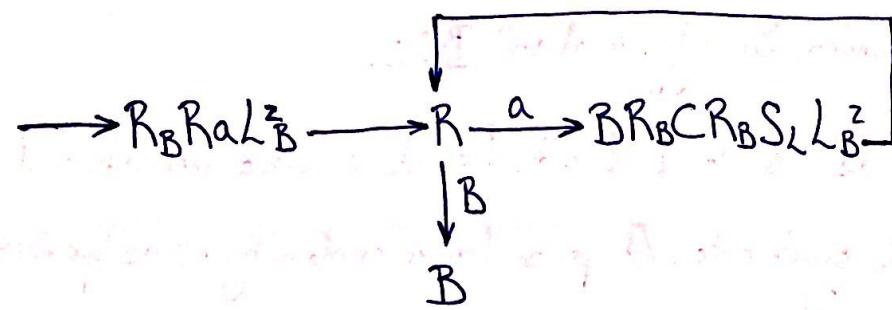
Teorema: \nexists ningún algoritmo que determine si el lenguaje generado por una GIC.

$G = (N, \Sigma, P, S)$ es Σ^* .

Teorema: \nexists ningún algoritmo que determine si una GIC arbitraria es ambigua.

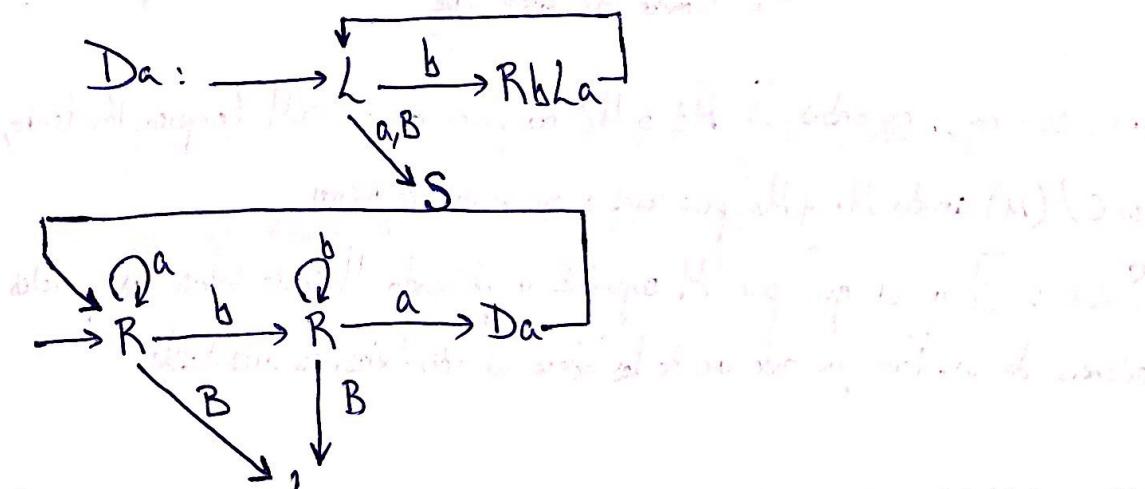
BOLETÍN 4 TC

62.- MT n aes genera una nueva cadena 2^n aes aceptada por B.

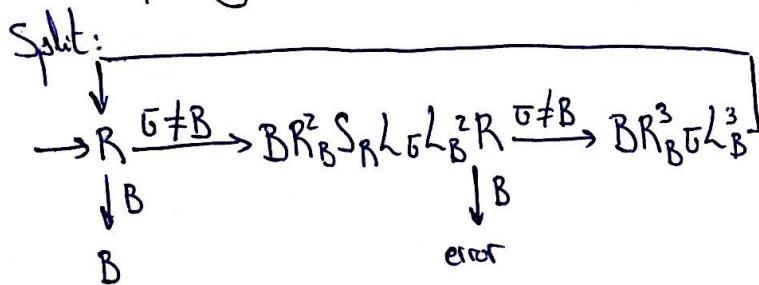


BaaaB \Rightarrow BaaaBa \Rightarrow BaaaBaB \Rightarrow BaBaaBaB \Rightarrow
 \Rightarrow BaBaaaB \Rightarrow BaaaBaaB \Rightarrow BaaaaaaaB

63.- MT BbbaaB se transforme en BaabbbB



64.- Split y Combine $\overset{L_B}{\Rightarrow}$ BacdefB a BaceBbdfB y BaceBbdfB a BacdefB



65.- BaaaaaaaaccccccB genere como salida B1101

La máquina tendrá 2 cintas:

- La 1^a almacenará la cadena de entrada
- La 2^a el nº binario. Su config. inicial será B0B.

M marcará con B los acs de la 1^a cinta. Por cada acs a marca incrementa en 1 el nº binario de la segunda cinta. El proceso termina cuando no hay más acs que marcar.

69.- M_1 y M_2 dos MT, ¿ $\exists w \in \Sigma^*$; M_1 como M_2 paren al procesar w?

Construimos M y; $L(M) = L(M_1) \cap L(M_2)$

M con 2 cintas $\begin{cases} \xrightarrow{1^a} \text{simular } M_1 \text{ sobre ella} \\ \xrightarrow{2^a} \text{simular } M_2 \text{ sobre ella} \end{cases}$

w se copia en ambas. Si M_1 o M_2 no paren con w $\Rightarrow M$ tampoco. Por tanto, $w \in L(M)$ cuando M_1 y M_2 paren ambas con w en aceptación.

Saber si \exists un w que pare M supondría ir probando M sobre todas las posibles cadenas de w. Pero por cada una de las cadenas w individuales, es irresoluble.

70.- PCP sobre alfabetos de un solo símbolo es resoluble.

$$\Sigma = \{a\}$$

3 fichas:

- Positives $\frac{aaaa}{aa} \equiv 3$

- Negatives $\frac{a}{aa} \equiv -2$

- Cero $\frac{aa}{aa} \equiv 0$

Las fichas "cero" ya tienen una solución.

Todos positivos, no hay solución

" " negativas, "

Si hay al menos una ficha positiva $\frac{aaaa}{aa} \equiv 3$ y una negativa $\frac{a}{aa} \equiv -2$, entonces hay soluciones, vienen dadas por la ecuación:

$$3x - 2y = 0$$

Cogiendo 2 veces la pieza 3 y 3 veces la pieza 2, obtendremos el mismo nº de aes (~~que de las~~) arriba y abajo de las piezas

$$\frac{a^{13}}{a^{13}} \equiv 0$$