

Lenguajes Formales y Computabilidad | FAMAF - UNC

# Combos de Definiciones, Convenciones Notacionales y Teoremas

Ramiro Lugo Viola

2025

# Contenido

<b>DEFINICIONES Y CONVENCIONES NOTACIONALES</b>	<b>1</b>
Combo 1 .....	1
Combo 2 .....	2
Combo 3 .....	3
Combo 4 .....	4
Combo 5 .....	4
Combo 6 .....	4
Combo 7 .....	5
Combo 8 .....	5
Combo 9 .....	6
Combo 10 .....	7
Combo 11 .....	9
Combo 12 .....	10
Combo 13 .....	11
Combo 14 .....	13
Combo 15 .....	14
Combo 16 .....	15
Combo 17 .....	16
Definiciones Auxiliares .....	16
Procedimiento efectivo .....	16
 <b>TEOREMAS</b>	 <b>17</b>
Combo 1 .....	17
Combo 2 .....	19
Combo 3 .....	21
Combo 4 .....	22
Combo 5 .....	24
Combo 6 .....	26
Combo 7 .....	29
Combo 8 .....	32
Combo 9 .....	34
Resultados Muy Usados en las Demostraciones .....	36

# DEFINICIONES Y CONVENCIONES NOTACIONALES

## Combo 1

1. Defina cuando un **conjunto**  $S \subseteq w^n \times \Sigma^{*m}$  es llamado  **$\Sigma$ -recursivo**.

*Nota:* no hace falta que defina “función  $\Sigma$ -recursiva”.

↳

*Definición:*

Un conjunto  $S \subseteq w^n \times \Sigma^{*m}$  es llamado  **$\Sigma$ -recursivo** cuando la función  $\chi_S^{w^n \times \Sigma^{*m}}$  sea  **$\Sigma$ -recursiva**.

2. Defina  $\langle s_1, s_2, \dots \rangle$ .

↳

*Definición:*

Dada una infinitupla  $(s_1, s_2, \dots) \in \omega^{[\mathbb{N}]}$ , usamos  $\langle s_1, s_2, \dots \rangle$  para denotar al número  $\prod_{i=1}^{\infty} \text{pr}(i)^{s_i}$ <sup>1</sup>

3. Defina “ $f$  es una función  $\Sigma$ -mixta”.

↳

*Definición:*

Sea  $\Sigma$  un alfabeto finito. Dada una función  $f$ , diremos que  **$f$  es una función  $\Sigma$ -mixta** si cumple las siguientes propiedades:

- Existen  $n, m \geq 0$  tales que  $D_f \subseteq w^n \times \Sigma^{*m}$
- Además  $I_f \subseteq \omega$  o  $I_f \subseteq \Sigma^*$

4. Defina “familia  $\Sigma$ -indexada de funciones”.

*Definición:*

Dado un alfabeto  $\Sigma$ , una **familia  $\Sigma$ -indexada de funciones** será una función  $\mathcal{G}$  tal que  $D_{\mathcal{G}} = \Sigma$  y para cada  $a \in D_{\mathcal{G}}$  se tiene que  $\mathcal{G}(a)$  es una función.

Notar que para cada  $a \in \Sigma$  escribimos  $\mathcal{G}_a$  para denotar a la función  $\mathcal{G}(a)$ .

5. Defina  $R(f, \mathcal{G})$ .

*Nota:* haga el caso de valores numéricos.

↳

*Definición:*

Supongamos  $\Sigma$  un alfabeto finito. Sea

$$f : S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \omega$$

con  $S_1, \dots, S_n \subseteq \omega$  y  $L_1, \dots, L_m \subseteq \Sigma^*$  conjuntos no vacíos y  $\mathcal{G}$  una familia  $\Sigma$ -indexada de funciones tal que

$$\mathcal{G}_a : \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \times \Sigma^* \rightarrow \omega \text{ para cada } a \in \Sigma$$

Definamos  **$R(f, \mathcal{G})$** :  $S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \times \Sigma^* \rightarrow \omega$  de la siguiente manera

- $R(f, \mathcal{G})(\vec{x}, \vec{\alpha}, \varepsilon) = f(\vec{x}, \vec{\alpha})$
- $R(f, \mathcal{G})(\vec{x}, \vec{\alpha}, \alpha a) = \mathcal{G}_a(R(f, \mathcal{G})(\vec{x}, \vec{\alpha}, \alpha), \vec{x}, \vec{\alpha}, \alpha)$

Diremos que  $R(f, \mathcal{G})$  es obtenida por *recursión primitiva* a partir de  $f$  y  $\mathcal{G}$ .

<sup>1</sup>Donde  $\text{pr}(i)$  es la función que retorna el  $i$ -ésimo primo.

## Combo 2

### 1. Defina $d \vdash^n d'$ y $d \vdash^* d'$

*Nota:* no hace falta que defina  $\vdash$

↳

*Definición:*

Para  $d, d' \in \mathbf{Des}$  y  $n \geq 0$ , escribimos  $d \vdash^n d'$  si existen  $d_1, \dots, d_{n+1} \in \mathbf{Des}$  tales que

$$d = d_1$$

$$d' = d_{n+1}$$

$$d_i \vdash d_{i+1}, \text{ para } i = 1, \dots, n$$

Luego definimos  $d \vdash^* d'$  si y solo si  $(\exists n \in \omega) d \vdash^n d'$ .

### 2. Defina $L(M)$

↳

*Definición:*

Sea  $M$  una máquina de Turing.

Diremos que una palabra  $\alpha \in \Sigma^*$  es aceptada por  $M$  por alcance de estado final cuando

$$[q_0 B \alpha] \vdash^* d, \text{ con } d \text{ tal que } \text{St}(d) \in F.$$

El **lenguaje aceptado por  $M$  por alcance de estado final** se define de la siguiente manera:

$$L(M) = \{\alpha \in \Sigma^* : \alpha \text{ es aceptada por } M \text{ por alcance de estado final}\}$$

### 3. Defina “ $f$ es una función de tipo $(n, m, s)$ ”.

↳

*Definición:*

Dada una función  $\Sigma$ -mixta  $f$  con  $n, m \in \omega$  tales que  $D_f \subseteq w^n \times \Sigma^{*m}$  y además  $I_f \subseteq \omega$ , entonces diremos que  **$f$  es una función de tipo  $(n, m, \#)$** .

Si en cambio  $I_f \subseteq \Sigma^*$ , entonces diremos que  **$f$  es una función de tipo  $(n, m, *)$** .

### 4. Defina $(x)$

↳

*Definición:*

Dado  $x \in \mathbb{N}$ , usaremos  $(x)$  para denotar a la única infinitupla  $(s_1, s_2, \dots) \in \omega^{[\mathbb{N}]}$  tal que

$$x = \langle s_1, s_2, \dots \rangle = \prod_{i=1}^{\infty} \text{pr}(i)^{s_i} \quad \text{donde } \text{pr}(i) \text{ es el } i\text{-ésimo primo.}$$

### 5. Defina $(x)_i$

↳

*Definición:*

Para  $i \in \mathbb{N}$ , usaremos  $(x)_i$  para denotar al  $i$ -ésimo elemento de la infinitupla  $(x)$ , es decir, al  $s_i$  de la definición anterior.

- $(x)_i$  es el exponente de  $\text{pr}(i)$  en la (única posible) factorización de  $x$  como producto de primos.<sup>2</sup>

<sup>2</sup>Se le suele llamar la “bajada  $i$ -ésima de  $x$ ” al número  $(x)_i$ .

(Ya que representa bajar al exponente de  $\text{pr}(i)$  en la única factorización de  $x$  como producto de primos)

## Combo 3

1. Defina cuando un conjunto  $S \subseteq w^n \times \Sigma^{*m}$  es llamado  **$\Sigma$ -recursivamente enumerable**.

*Nota:* no hace falta que defina “función  $\Sigma$ -recursiva”.

↳

*Definición:*

Un conjunto  $S \subseteq w^n \times \Sigma^{*m}$  será llamado  **$\Sigma$ -recursivamente enumerable** cuando sea vacío o haya una función  $F : \omega \rightarrow w^n \times \Sigma^{*m}$  tal que  $I_F = S$  y  $F_{(i)}$  sea  $\Sigma$ -recursiva, para cada  $i \in \{1, \dots, n+m\}$ .<sup>3</sup>

2. Defina  $s^{\leq}$

↳

*Definición:*

Sea  $\Sigma = \{a_1, \dots, a_n\}$  un alfabeto no vacío, con  $\leq$  un orden total sobre  $\Sigma$  dado por  $a_1 < a_2 < \dots < a_n$ . Definimos la función  $s^{\leq} : \Sigma^* \rightarrow \Sigma^*$  de la siguiente manera

$$\begin{aligned} s^{\leq}((a_n)^m) &= (a_1)^{m+1} \quad \text{para cada } m \geq 0 \\ s^{\leq}(\alpha a_i (a_n)^m) &= \alpha a_{i+1} (a_1)^m \quad \text{cada vez que } \alpha \in \Sigma^*, \text{ con } 1 \leq i < n \text{ y } m \geq 0 \end{aligned}$$

3. Defina  $*^{\leq}$

↳

*Definición:*

Sea  $\Sigma = \{a_1, \dots, a_n\}$  un alfabeto no vacío, con  $\leq$  un orden total sobre  $\Sigma$  dado por  $a_1 < a_2 < \dots < a_n$ . Definimos la función  $*^{\leq} : \omega \rightarrow \Sigma^*$  de la siguiente manera

$$\begin{aligned} *^{\leq}(0) &= \varepsilon \\ *^{\leq}(i+1) &= s^{\leq}(*^{\leq}(i)) \quad (\text{donde } s^{\leq} \text{ es la función definida justo antes}) \end{aligned}$$

4. Defina  $\#^{\leq}$

↳

*Definición:*

*Lema:* Sea  $\Sigma = \{a_1, \dots, a_n\}$  un alfabeto no vacío, con  $\leq$  un orden total sobre  $\Sigma$  dado por  $a_1 < a_2 < \dots < a_n$ . Entonces para cada  $\alpha \in \Sigma^* - \{\varepsilon\}$  hay únicos  $k \in \omega$  y  $i_0, i_1, \dots, i_k \in \{1, \dots, n\}$  tales que  $\alpha = a_{i_k} \dots a_{i_0}$ .

Sea  $\Sigma = \{a_1, \dots, a_n\}$  un alfabeto no vacío, con  $\leq$  un orden total sobre  $\Sigma$  dado por  $a_1 < a_2 < \dots < a_n$ .

Gracias al *Lema* anterior podemos definir la función  $\#^{\leq} : \Sigma^* \rightarrow \omega$  de la siguiente manera

$$\begin{aligned} \#^{\leq}(\varepsilon) &= 0 \\ \#^{\leq}(a_{i_k} \dots a_{i_0}) &= i_k n^k + \dots + i_0 n^0 \quad \text{para } i_0, i_1, \dots, i_k \in \{1, \dots, n\} \\ \#^{\leq}(\alpha) &= i_k n^k + \dots + i_0 n^0 \text{ con } \alpha = a_{i_k} \dots a_{i_0} \text{ tal que } k \in \omega \text{ y } i_0, i_1, \dots, i_k \in \{1, \dots, n\} \end{aligned}$$

<sup>3</sup>

Dados  $k, l, n, m \in \omega$  y  $F : D_F \subseteq \omega^k \times \Sigma^{*l} \rightarrow w^n \times \Sigma^{*m}$  con  $n+m \geq 1$ . Entonces denotaremos con  $F_{(i)}$  a la función  $p_i^{n,m} \circ F$ .

- $F_{(i)} : D_F \subseteq \omega^k \times \Sigma^{*l} \rightarrow \omega$  para  $i = 1, \dots, n$ .
- $F_{(i)} : D_F \subseteq \omega^k \times \Sigma^{*l} \rightarrow \Sigma^*$  para  $i = n+1, \dots, n+m$ .

por lo tanto son  $\Sigma$ -mixtas y  $F = [F_{(1)}, \dots, F_{(n+m)}]$ .

## Combo 4

1. Defina cuando una función  $f : D_f \subseteq w^n \times \Sigma^{*m} \rightarrow \omega$  es llamada  $\Sigma$ -efectivamente computable y defina “el procedimiento  $\mathbb{P}$  computa a la función  $f$ ”.  $\hookrightarrow$

*Definición:*

Una función  $f : D_f \subseteq w^n \times \Sigma^{*m} \rightarrow \omega$  es llamada  **$\Sigma$ -efectivamente computable** si hay un procedimiento efectivo  $\mathbb{P}$  tal que

- (1) El conjunto de datos de entrada de  $\mathbb{P}$  es  $w^n \times \Sigma^{*m}$ .
- (2) El conjunto de datos de salida está contenido en  $\omega$ .
- (3) Si  $(\vec{x}, \vec{\alpha}) \in D_f$ , entonces  $\mathbb{P}$  se detiene partiendo de  $(\vec{x}, \vec{\alpha})$ , dando como dato de salida  $f(\vec{x}, \vec{\alpha})$ .
- (4) Si  $(\vec{x}, \vec{\alpha}) \in (w^n \times \Sigma^{*m}) - D_f$ , entonces  $\mathbb{P}$  no se detiene partiendo de  $(\vec{x}, \vec{\alpha})$ .

Cuando  $\mathbb{P}$  cumpla los items anteriores diremos que “**el procedimiento  $\mathbb{P}$  computa a la función  $f$** ”.

## Combo 5

1. Defina cuando un conjunto  $S \subseteq w^n \times \Sigma^{*m}$  es llamado  $\Sigma$ -efectivamente computable y defina “el procedimiento efectivo  $\mathbb{P}$  decide la pertenencia a  $S$ ”.  $\hookrightarrow$

*Definición:*

Un conjunto  $S \subseteq w^n \times \Sigma^{*m}$  será llamado  **$\Sigma$ -efectivamente computable** cuando la función característica  $\chi_S^{w^n \times \Sigma^{*m}}$  sea  $\Sigma$ -efectivamente computable.

Si  $\mathbb{P}$  es un procedimiento efectivo el cual computa a la función  $\chi_S^{w^n \times \Sigma^{*m}}$ , diremos que “**el procedimiento efectivo  $\mathbb{P}$  decide la pertenencia a  $S$** ”, con respecto al conjunto  $w^n \times \Sigma^{*m}$ . Es decir

- El conjunto de datos de entrada de  $\mathbb{P}$  es  $w^n \times \Sigma^{*m}$ , siempre termina y da como dato de salida un elemento de  $\{0, 1\}$ .
- Dado  $(\vec{x}, \vec{\alpha}) \in w^n \times \Sigma^{*m}$ ,  $\mathbb{P}$  da como salida el número 1 si  $(\vec{x}, \vec{\alpha}) \in S$  y al número 0 si  $(\vec{x}, \vec{\alpha}) \notin S$ .

## Combo 6

1. Defina cuando un conjunto  $S \subseteq w^n \times \Sigma^{*m}$  es llamado  $\Sigma$ -efectivamente enumerable y defina “el procedimiento efectivo  $\mathbb{P}$  enumera a  $S$ ”.  $\hookrightarrow$

*Definición:*

Un conjunto  $S \subseteq w^n \times \Sigma^{*m}$  será llamado  **$\Sigma$ -efectivamente enumerable** cuando sea vacío o haya una función  $F : \omega \rightarrow w^n \times \Sigma^{*m}$  tal que  $I_F = S$  y  $F_{(i)}$  sea  $\Sigma$ -efectivamente computable, para cada  $i \in \{1, \dots, n + m\}$ .

Diremos que “**el procedimiento efectivo  $\mathbb{P}$  enumera a  $S$** ” cuando

- (1) El conjunto de datos de entrada de  $\mathbb{P}$  es  $\omega$ .
- (2)  $\mathbb{P}$  se detiene para cada  $x \in \omega$ .
- (3) El conjunto de datos de salida de  $\mathbb{P}$  es igual a  $S$ .  
(Es decir, siempre que  $\mathbb{P}$  se detiene, da como salida un elemento de  $S$ , y para cada elemento  $(\vec{x}, \vec{\alpha}) \in S$ , hay un  $x \in \omega$  tal que  $\mathbb{P}$  da como salida a  $(\vec{x}, \vec{\alpha})$  cuando lo corremos con  $x$  como dato de entrada).

En los combos 4, 5 y 6 usamos la **definición de procedimiento efectivo** que está en definiciones auxiliares.

## Combo 7

1. Defina cuando una función  $f : D_f \subseteq w^n \times \Sigma^{*m} \rightarrow \omega$  es llamada  $\Sigma$ -Turing computable y defina “la máquina de Turing  $M$  computa a la función  $f$ ”.  $\hookrightarrow$

*Definición:*

Diremos que una función  $f : D_f \subseteq w^n \times \Sigma^{*m} \rightarrow \omega$  es  **$\Sigma$ -Turing computable** si existe una máquina de Turing con unit,  $M = (Q, \Sigma, \Gamma, \delta, q_0, B, \mathbf{I}, F)$  tal que

- (1) Si  $(\vec{x}, \vec{\alpha}) \in D_f$ , entonces hay un  $p \in Q$  tal que

$$[q_0 B \mathbf{I}^{x_1} B \dots B \mathbf{I}^{x_n} B \alpha_1 B \dots B \alpha_m] \vdash^* [p B \mathbf{I}^{f(\vec{x}, \vec{\alpha})}]$$

y  $[p B \mathbf{I}^{f(\vec{x}, \vec{\alpha})}] \not\vdash d$ , para cada  $d \in \text{Des}$ .

- (2) Si  $(\vec{x}, \vec{\alpha}) \in w^n \times \Sigma^{*m} - D_f$ , entonces  $M$  **no** se detiene partiendo de

$$[q_0 B \mathbf{I}^{x_1} B \dots B \mathbf{I}^{x_n} B \alpha_1 B \dots B \alpha_m]$$

Si  $M$  y  $f$  cumplen los ítems (1) y (2), diremos que “**la máquina de Turing  $M$  computa a la función  $f$** ”.

## Combo 8

1. Defina  $M(P)$   $\hookrightarrow$

*Definición:*

Sea  $\Sigma$  un alfabeto finito y sea  $P : D_P \subseteq \omega \times w^n \times \Sigma^{*m} \rightarrow \omega$  un predicado. Dado  $(\vec{x}, \vec{\alpha}) \in w^n \times \Sigma^{*m}$ , cuando exista al menos un  $t \in \omega$  tal que  $P(t, \vec{x}, \vec{\alpha}) = 1$ , usaremos  $\min_t P(t, \vec{x}, \vec{\alpha})$  para denotar al menor de tales  $t$ 's. Esta expresión está definida solo para aquellas  $(n + m)$ -uplas  $(\vec{x}, \vec{\alpha})$  para las cuales existe al menos un  $t \in \omega$  tal que se da  $P(t, \vec{x}, \vec{\alpha}) = 1$  (obviamente también  $(t, \vec{x}, \vec{\alpha}) \in D_P$ ). Ahora sí, definamos

$$M(P) = \lambda \vec{x} \vec{\alpha} \left[ \min_t P(t, \vec{x}, \vec{\alpha}) \right]$$

Es decir que

$$D_{M(P)} = \{(\vec{x}, \vec{\alpha}) \in w^n \times \Sigma^{*m} : (\exists t \in \omega) P(t, \vec{x}, \vec{\alpha}) = 1\}$$

$$M(P)(\vec{x}, \vec{\alpha}) = \min_t P(t, \vec{x}, \vec{\alpha}), \text{ para cada } (\vec{x}, \vec{\alpha}) \in D_{M(P)}$$

2. Defina  $Lt$   $\hookrightarrow$

*Definición:*

La función  $Lt : \mathbb{N} \rightarrow \omega$  se define de la siguiente manera  $Lt(x) = \begin{cases} \max_i (x)_i \neq 0 & \text{si } x \neq 1 \\ 0 & \text{si } x = 1 \end{cases}$

3. Defina Conjunto rectangular  $\hookrightarrow$

*Definición:*

Un conjunto  $\Sigma$ -mixto  $S$  será llamado **rectangular** si es de la forma  $S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m$  con  $S_1, \dots, S_n \subseteq \omega$  y  $L_1, \dots, L_m \subseteq \Sigma^*$ .

4. Defina “ $S$  es un conjunto de tipo  $(n, m)$ ”  $\hookrightarrow$

*Definición:*

Dado un conjunto  $\Sigma$ -mixto  $S$ , si  $n, m \in \omega$  son tales que  $S \subseteq w^n \times \Sigma^{*m}$ , entonces diremos que  **$S$  es un conjunto de tipo  $(n, m)$** .<sup>4</sup>

<sup>4</sup>Si  $S \neq \emptyset$ , entonces hay un único par  $(n, m)$  con esa propiedad. Pero si  $S = \emptyset$ , entonces hay varios pares  $(n, m)$  con esa propiedad. Por ello para hablar de **EL TIPO DE  $S$**  tiene que ser  $S \neq \emptyset$ .

## Combo 9

### 1. Defina “ $I$ es una instrucción de $S^\Sigma$ ”

↳

*Definición:*

Una **instrucción de  $S^\Sigma$**  es ya sea una instrucción básica de  $S^\Sigma$ , o una instrucción de la forma  $\alpha I$ , donde  $\alpha \in \{L\bar{n} : n \in \mathbb{N}\}$  e  $I$  es una instrucción básica de  $S^\Sigma$ .

(Usamos  $\text{Ins}^\Sigma$  para denotar al conjunto de todas las instrucciones de  $S^\Sigma$ ).

### 2. Defina “ $\mathcal{P}$ es programa de $S^\Sigma$ ”

↳

*Definición:*

Un **programa de  $S^\Sigma$**  es una palabra de la forma

$$I_1 I_2 \dots I_n$$

donde  $n \geq 1$ ,  $I_1, \dots, I_n \in \text{Ins}^\Sigma$  y además cumple la siguiente propiedad llamada *ley de los GOTO*

*Para cada  $i \in \{1, \dots, n\}$ , si  $\text{GOTO}L\bar{m}$  es tramo final de  $I_i$ ,  
entonces existe  $j \in \{1, \dots, n\}$  tal que  $I_j$  tiene label  $L\bar{m}$ .*

(Usamos  $\text{Pro}^\Sigma$  para denotar al conjunto de todos los programas de  $S^\Sigma$ ).

### 3 y 4. Defina $I_i^\mathcal{P}$ y $n(\mathcal{P})$

↳

*Definición:*

*Lema :*

- (a) Si  $I_1, \dots, I_n = J_1, \dots, J_m$  con  $I_1, \dots, I_n, J_1, \dots, J_m \in \text{Ins}^\Sigma$ , entonces  $n = m$  y  $I_i = J_i$  para cada  $i \geq 1$ .
- (b) Si  $\mathcal{P} \in \text{Pro}^\Sigma$ , entonces existe una única sucesión de instrucciones  $I_1, \dots, I_n$  tal que  $\mathcal{P} = I_1 I_2 \dots I_n$ .

Gracias al *Lema* anterior y dados  $i \in \omega$  y  $\mathcal{P} \in \text{Pro}^\Sigma$ , tenemos unicamente determinados  $n(\mathcal{P}) \in \mathbb{N}$  e  $I_1^\mathcal{P}, \dots, I_{n(\mathcal{P})}^\mathcal{P} \in \text{Ins}^\Sigma$  tales que  $\mathcal{P} = I_1^\mathcal{P} \dots I_{n(\mathcal{P})}^\mathcal{P}$  y definimos  $I_i^\mathcal{P} = \varepsilon$  cuando  $i = 0$  o  $i > n(\mathcal{P})$ .

### 5. Defina la función $\text{Bas}$

↳

*Definición:*

La función  $\text{Bas} : \text{Ins}^\Sigma \rightarrow (\Sigma \cup \Sigma_p)^*$  se define de la siguiente manera

$$\text{Bas}(I) = \begin{cases} J & \text{si } I \text{ es de la forma } L\bar{k}J, \text{ con } k \in \mathbb{N} \text{ y } J \in \text{Ins}^\Sigma \\ I & \text{caso contrario} \end{cases}$$

## Combo 10

### 1. Defina relativo al lenguaje $S^\Sigma$ , “estado”

↳

*Definición:*

Un **estado** es un par  $(\vec{s}, \vec{\sigma}) = ((s_1, s_2, \dots), (\sigma_1, \sigma_2, \dots)) \in \omega^{[\mathbb{N}]} \times \Sigma^{*[\mathbb{N}]}$

Si  $i \geq 1$ , entonces diremos que  $s_i$  es el contenido o valor de la variable  $N\bar{i}$  en el estado  $(\vec{s}, \vec{\sigma})$  y  $\sigma_i$  es el contenido o valor de la variable  $P\bar{i}$  en el estado  $(\vec{s}, \vec{\sigma})$ .

### 2. Defina relativo al lenguaje $S^\Sigma$ , “descripción instantánea”

↳

*Definición:*

Una **descripción instantánea** es una terna  $(i, \vec{s}, \vec{\sigma})$  tal que  $(\vec{s}, \vec{\sigma})$  es un estado e  $i \in \omega$ .

(Es decir que el  $\omega \times \omega^{[\mathbb{N}]} \times \Sigma^{*[\mathbb{N}]}$  es el conjunto formado por todas las descripciones instantáneas)

### 3. Defina relativo al lenguaje $S^\Sigma$ la función $S_{\mathcal{P}}$

↳

*Definición:*

Dado un programa  $\mathcal{P} \in \text{Pro}^\Sigma$ , definimos  $S_{\mathcal{P}} : \omega \times \omega^{[\mathbb{N}]} \times \Sigma^{*[\mathbb{N}]} \rightarrow \omega \times \omega^{[\mathbb{N}]} \times \Sigma^{*[\mathbb{N}]}$  tal que

$$S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = \text{descripción instantánea que resulta luego de realizarp}^5 I_i^{\mathcal{P}}, \text{ estando en el estado } (\vec{s}, \vec{\sigma})$$

Para definirla formalmente damos los siguientes casos

Cuando  $i \in \{1, \dots, n(\mathcal{P})\}$  entonces

- Caso Bas( $I_i^{\mathcal{P}}$ ) =  $N\bar{k} \leftarrow N\bar{k} - 1$ . Entonces  $S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (i + 1, (s_1, \dots, s_{k-1}, s_k - 1, s_{k+1}, \dots), \vec{\sigma})$
- Caso Bas( $I_i^{\mathcal{P}}$ ) =  $N\bar{k} \leftarrow N\bar{k} + 1$ . Entonces  $S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (i + 1, (s_1, \dots, s_{k-1}, s_k + 1, s_{k+1}, \dots), \vec{\sigma})$
- Caso Bas( $I_i^{\mathcal{P}}$ ) =  $N\bar{k} \leftarrow N\bar{n}$ . Entonces  $S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (i + 1, (s_1, \dots, s_{k-1}, s_n, s_{k+1}, \dots), \vec{\sigma})$
- Caso Bas( $I_i^{\mathcal{P}}$ ) =  $N\bar{k} \leftarrow 0$ . Entonces  $S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (i + 1, (s_1, \dots, s_{k-1}, 0, s_{k+1}, \dots), \vec{\sigma})$
- Caso Bas( $I_i^{\mathcal{P}}$ ) = IF  $N\bar{k} \neq 0$  GOTO  $L\bar{m}$ . Sea  $s_k$  el valor de  $N\bar{k}$  en  $(\vec{s}, \vec{\sigma})$ , entonces

$$S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = \begin{cases} (\min\{l : I_l^{\mathcal{P}} \text{ tiene label } L\bar{m}\}, \vec{s}, \vec{\sigma}) & \text{si } s_k \neq 0 \\ (i + 1, \vec{s}, \vec{\sigma}) & \text{si } s_k = 0 \end{cases}$$

- Caso Bas( $I_i^{\mathcal{P}}$ ) =  $P\bar{k} \leftarrow \sim P\bar{k}$ . Entonces  $S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (i + 1, \vec{s}, (\sigma_1, \dots, \sigma_{k-1}, \sim \sigma_k, \sigma_{k+1}, \dots))$
- Caso Bas( $I_i^{\mathcal{P}}$ ) =  $P\bar{k} \leftarrow P\bar{k}.a$ . Entonces  $S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (i + 1, \vec{s}, (\sigma_1, \dots, \sigma_{k-1}, \sigma_k a, \sigma_{k+1}, \dots))$
- Caso Bas( $I_i^{\mathcal{P}}$ ) =  $P\bar{k} \leftarrow P\bar{n}$ . Entonces  $S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (i + 1, \vec{s}, (\sigma_1, \dots, \sigma_{k-1}, \sigma_n, \sigma_{k+1}, \dots))$
- Caso Bas( $I_i^{\mathcal{P}}$ ) =  $P\bar{k} \leftarrow \varepsilon$ . Entonces  $S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (i + 1, \vec{s}, (\sigma_1, \dots, \sigma_{k-1}, \varepsilon, \sigma_{k+1}, \dots))$
- Caso Bas( $I_i^{\mathcal{P}}$ ) = IF  $P\bar{k}$  BEGINS a GOTO  $L\bar{m}$ . Sea  $\sigma_k$  el valor de  $P\bar{k}$  en  $(\vec{s}, \vec{\sigma})$ , entonces

$$S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = \begin{cases} (\min\{l : I_l^{\mathcal{P}} \text{ tiene label } L\bar{m}\}, \vec{s}, \vec{\sigma}) & \text{si } \sigma_k \text{ comienza con a} \\ (i + 1, \vec{s}, \vec{\sigma}) & \text{si } \sigma_k \text{ no comienza con a} \end{cases}$$

- Caso Bas( $I_i^{\mathcal{P}}$ ) = GOTO  $L\bar{m}$ . Entonces  $S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (\min\{l : I_l^{\mathcal{P}} \text{ tiene label } L\bar{m}\}, \vec{s}, \vec{\sigma})$
- Caso Bas( $I_i^{\mathcal{P}}$ ) = SKIP. Entonces  $S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (i + 1, \vec{s}, \vec{\sigma})$

Pero cuando  $i \notin \{1, \dots, n(\mathcal{P})\}$  simplemente,  $S_{\mathcal{P}}(i, \vec{s}, \vec{\sigma}) = (i, \vec{s}, \vec{\sigma})$ .

<sup>5</sup>El verbo “realizarp” una actividad es realizarla si se puede.

4. Defina relativo al lenguaje  $S^\Sigma$ , “estado obtenido luego de  $t$  pasos, partiendo del estado  $(\vec{s}, \vec{\sigma})$ ”  $\hookrightarrow$

*Definición:*

Diremos que

$$\overbrace{S_{\mathcal{P}}(\dots S_{\mathcal{P}}(S_{\mathcal{P}}(1, \vec{s}, \vec{\sigma}))\dots)}^{t \text{ veces}} = (j, \vec{u}, \vec{\eta})$$

es la descripción instantánea obtenida luego de  $t$  pasos, partiendo del estado  $(\vec{s}, \vec{\sigma})$  y  $(\vec{u}, \vec{\eta})$  **es el estado obtenido luego de  $t$  pasos, partiendo del estado  $(\vec{s}, \vec{\sigma})$ .**

5. Defina relativo al lenguaje  $S^\Sigma$ , “ $\mathcal{P}$  se detiene (luego de  $t$  pasos), partiendo del estado  $(\vec{s}, \vec{\sigma})$ ”  $\hookrightarrow$

*Definición:*

Cuando la primera coordenada de

$$\overbrace{S_{\mathcal{P}}(\dots S_{\mathcal{P}}(S_{\mathcal{P}}(1, \vec{s}, \vec{\sigma}))\dots)}^{t \text{ veces}}$$

sea igual a  $n(\mathcal{P}) + 1$  diremos que  **$\mathcal{P}$  se detiene (luego de  $t$  pasos), partiendo del estado  $(\vec{s}, \vec{\sigma})$ .**

## Combo 11

### 1. Defina $\Psi_{\mathcal{P}}^{n,m,\#}$

↳

*Definición:*

Dados  $x_1, \dots, x_n \in \omega$  y  $\sigma_1, \dots, \sigma_m \in \Sigma^*$ , usaremos  $\|x_1, \dots, x_n, \sigma_1, \dots, \sigma_m\|$  para denotar el estado  $((x_1, \dots, x_n, 0, \dots), (\sigma_1, \dots, \sigma_m, \varepsilon, \dots))$ .

Ahora sí, dado  $\mathcal{P} \in \text{Pro}^\Sigma$ , definamos para cada par  $n, m \in \omega$ , la función  $\Psi_{\mathcal{P}}^{n,m,\#}$  de la siguiente manera

$$D_{\Psi_{\mathcal{P}}^{n,m,\#}} = \{(\vec{x}, \vec{\alpha}) \in w^n \times \Sigma^{*m} : \mathcal{P} \text{ termina,} \\ \text{partiendo del estado } \|x_1, \dots, x_n, \sigma_1, \dots, \sigma_m\|\}$$

$$\Psi_{\mathcal{P}}^{n,m,\#}(\vec{x}, \vec{\alpha}) = \text{valor de N1 en el estado obtenido cuando } \mathcal{P} \text{ termina,} \\ \text{partiendo de } \|x_1, \dots, x_n, \sigma_1, \dots, \sigma_m\|$$

### 2. Defina “ $f$ es $\Sigma$ -computable”

↳

*Definición:*

Una función  $\Sigma$ -mixta  $f : D_f \subseteq w^n \times \Sigma^{*m} \rightarrow \omega$  es llamada  **$\Sigma$ -computable** si hay un programa  $\mathcal{P}$  de  $S^\Sigma$  tal que  $f = \Psi_{\mathcal{P}}^{n,m,\#}$ .

Análogamente una función  $\Sigma$ -mixta  $f : D_f \subseteq w^n \times \Sigma^{*m} \rightarrow \Sigma^*$  es llamada  **$\Sigma$ -computable** si hay un programa  $\mathcal{P}$  de  $S^\Sigma$  tal que  $f = \Psi_{\mathcal{P}}^{n,m,*}$ .

### 3. Defina “ $\mathcal{P}$ computa a $f$ ”

↳

*Definición:*

Sea  $\mathcal{P}$  un programa de  $S^\Sigma$ .

Dada una función  $\Sigma$ -mixta  $f : D_f \subseteq w^n \times \Sigma^{*m} \rightarrow \omega$  diremos que  **$\mathcal{P}$  computa  $f$**  si  $f = \Psi_{\mathcal{P}}^{n,m,\#}$ .

Análogamente, si  $f : D_f \subseteq w^n \times \Sigma^{*m} \rightarrow \Sigma^*$ , también diremos que  **$\mathcal{P}$  computa a  $f$**  si  $f = \Psi_{\mathcal{P}}^{n,m,*}$ .

### 4. Defina $M^{\leq}(P)$

↳

*Definición:*

Sea  $\Sigma$  un alfabeto no vacío,  $\leq$  un orden total sobre  $\Sigma^6$  y  $P : D_P \subseteq w^n \times \Sigma^{*m} \times \Sigma^* \rightarrow \omega$  un predicado.

Dado  $(\vec{x}, \vec{\alpha}) \in w^n \times \Sigma^{*m}$ , cuando exista al menos un  $\alpha \in \Sigma^*$  tal que  $P(\vec{x}, \vec{\alpha}, \alpha) = 1$ , usaremos

$$\min_{\alpha} P(\vec{x}, \vec{\alpha}, \alpha)$$

para denotar al menor de tales  $\alpha'$ s. Esta expresión está definida solo para aquellas  $(n+m)$ -uplas  $(\vec{x}, \vec{\alpha})$  para las cuales existe al menos un  $\alpha \in \Sigma^*$  tal que se da  $P(\vec{x}, \vec{\alpha}, \alpha) = 1$  (obviamente también  $(\vec{x}, \vec{\alpha}, \alpha) \in D_P$ ).

Ahora sí, definamos

$$M^{\leq}(P) = \lambda \vec{x} \vec{\alpha} [\min_{\alpha} P(\vec{x}, \vec{\alpha}, \alpha)]$$

Es decir que

$$D_{M^{\leq}(P)} = \{(\vec{x}, \vec{\alpha}) \in w^n \times \Sigma^{*m} : (\exists \alpha \in \Sigma^*) P(\vec{x}, \vec{\alpha}, \alpha) = 1\}$$

$$M^{\leq}(P)(\vec{x}, \vec{\alpha}) = \min_{\alpha} P(\vec{x}, \vec{\alpha}, \alpha), \text{ para cada } (\vec{x}, \vec{\alpha}) \in D_{M^{\leq}(P)}$$

<sup>6</sup>Recordar que  $\leq$  puede ser naturalmente extendido a un orden total sobre  $\Sigma^*$  ↳

## Combo 12

1. Defina cuando un conjunto  $S \subseteq w^n \times \Sigma^{*m}$  es llamado  $\Sigma$ -computable.  $\hookrightarrow$

*Definición:*

Un conjunto  $S \subseteq w^n \times \Sigma^{*m}$  será llamado  **$\Sigma$ -computable** cuando la función  $\chi_S^{w^n \times \Sigma^{*m}}$  sea  $\Sigma$ -computable.

2. Defina cuando un conjunto  $S \subseteq w^n \times \Sigma^{*m}$  es llamado  $\Sigma$ -enumerable.  $\hookrightarrow$

*Definición:*

Un conjunto  $S \subseteq w^n \times \Sigma^{*m}$  será llamado  **$\Sigma$ -enumerable** cuando sea vacío o haya una función  $F : \omega \rightarrow w^n \times \Sigma^{*m}$  tal que  $I_F = S$  y  $F_{(i)}$  sea  $\Sigma$ -computable, para cada  $i \in \{1, \dots, n+m\}$ .

3. Defina “el programa  $\mathcal{P}$  que enumera a  $S$ ”.  $\hookrightarrow$

*Definición:*

Diremos que **el programa  $\mathcal{P} \in S^\Sigma$  enumera a  $S$**  cuando cumple lo siguiente

- Para cada  $x \in \omega$ , tenemos que  $\mathcal{P}$  se detiene partiendo del estado  $\|x\|$  y llega a un estado  $((x_1, \dots, x_n, y_1, \dots), (\sigma_1, \dots, \sigma_m, \beta_1, \dots))$ , donde  $(x_1, \dots, x_n, \sigma_1, \dots, \sigma_m) \in S$ .
- Para cada  $(x_1, \dots, x_n, \sigma_1, \dots, \sigma_m) \in S$ , hay un  $x \in \omega$  tal que  $\mathcal{P}$  se detiene partiendo del estado  $\|x\|$  y llega a un estado  $((x_1, \dots, x_n, y_1, \dots), (\sigma_1, \dots, \sigma_m, \beta_1, \dots))$ .

(Notar que en los estados se agregan  $y_1, \dots$  y  $\beta_1, \dots$  para representar “datos extra” que el programa puede haber producido además de los datos de salida relevantes  $(x_1, \dots, x_n, \sigma_1, \dots, \sigma_m)$  )

## Combo 13

**1, 2 y 3.** Defina las funciones  $i^{n,m}$ ,  $E_{\#}^{n,m}$  y  $E_{*}^{n,m}$

↳

*Definiciones:*

Sean  $n, m \in \omega$  fijos. Definimos

$$\begin{aligned} i^{n,m} &: \omega \times w^n \times \Sigma^{*m} \times \text{Pro}^{\Sigma} \rightarrow \omega \\ E_{\#}^{n,m} &: \omega \times w^n \times \Sigma^{*m} \times \text{Pro}^{\Sigma} \rightarrow \omega^{[\mathbb{N}]} \\ E_{*}^{n,m} &: \omega \times w^n \times \Sigma^{*m} \times \text{Pro}^{\Sigma} \rightarrow \Sigma^{*[\mathbb{N}]} \end{aligned}$$

de la siguiente manera

$$\begin{aligned} (i^{n,m}(0, \vec{x}, \vec{\alpha}, \mathcal{P}), E_{\#}^{n,m}(0, \vec{x}, \vec{\alpha}, \mathcal{P}), E_{*}^{n,m}(0, \vec{x}, \vec{\alpha}, \mathcal{P})) &= (1, (x_1, \dots, x_n, 0, \dots), (\alpha_1, \dots, \alpha_m, \varepsilon, \dots)) \\ (i^{n,m}(t+1, \vec{x}, \vec{\alpha}, \mathcal{P}), E_{\#}^{n,m}(t+1, \vec{x}, \vec{\alpha}, \mathcal{P}), E_{*}^{n,m}(t+1, \vec{x}, \vec{\alpha}, \mathcal{P})) &= \\ S_{\mathcal{P}}(i^{n,m}(t, \vec{x}, \vec{\alpha}, \mathcal{P}), E_{\#}^{n,m}(t, \vec{x}, \vec{\alpha}, \mathcal{P}), E_{*}^{n,m}(t, \vec{x}, \vec{\alpha}, \mathcal{P})) \end{aligned}$$

Notar que

$$(i^{n,m}(t, \vec{x}, \vec{\alpha}, \mathcal{P}), E_{\#}^{n,m}(t, \vec{x}, \vec{\alpha}, \mathcal{P}), E_{*}^{n,m}(t, \vec{x}, \vec{\alpha}, \mathcal{P}))$$

es la descripción instantánea luego de correr  $\mathcal{P}$  una cantidad  $t$  de pasos, partiendo de  $\|x_1, \dots, x_n, \alpha_1, \dots, \alpha_m\|$ .<sup>7</sup>

Además  $i^{n,m}$  es  $(\Sigma \cup \Sigma_p)$ -mixta pero  $E_{\#}^{n,m}$  y  $E_{*}^{n,m}$  no.

**4 y 5.** Defina las funciones  $E_{\#j}^{n,m}$  y  $E_{*j}^{n,m}$

↳

*Definiciones:*

Definimos para cada  $j \in \mathbb{N}$ , las funciones

$$\begin{aligned} E_{\#j}^{n,m} &: \omega \times w^n \times \Sigma^{*m} \times \text{Pro}^{\Sigma} \rightarrow \omega \\ E_{*j}^{n,m} &: \omega \times w^n \times \Sigma^{*m} \times \text{Pro}^{\Sigma} \rightarrow \Sigma^{*} \end{aligned}$$

de la siguiente manera

$$\begin{aligned} E_{\#j}^{n,m}(t, \vec{x}, \vec{\alpha}, \mathcal{P}) &= j\text{-ésima coordenada de } E_{\#}^{n,m}(t, \vec{x}, \vec{\alpha}, \mathcal{P}) \\ E_{*j}^{n,m}(t, \vec{x}, \vec{\alpha}, \mathcal{P}) &= j\text{-ésima coordenada de } E_{*}^{n,m}(t, \vec{x}, \vec{\alpha}, \mathcal{P}) \end{aligned}$$

y es claro que estas funciones son  $(\Sigma \cup \Sigma_p)$ -mixtas.

<sup>7</sup>Osea que para el paso  $t$  podemos pensar que  $i^{n,m}$  representa “número de instrucción”,  $E_{\#}^{n,m}$  representa “los valores numéricos” y  $E_{*}^{n,m}$  representa “los valores alfabéticos”.

6. Defina  $Halt^{n,m}$ 

*Definiciones:*

Dados  $n, m \in \omega$ , definimos

$$Halt^{n,m} = \lambda t \vec{x} \vec{\alpha} \mathcal{P} [i^{n,m}(t, \vec{x}, \vec{\alpha}) = n(\mathcal{P}) + 1]$$

Es decir que dado  $(t, \vec{x}, \vec{\alpha}, \mathcal{P}) \in \omega \times w^n \times \Sigma^{*m} \times \text{Pro}^\Sigma$ , tenemos que

$$Halt^{n,m}(t, \vec{x}, \vec{\alpha}, \mathcal{P}) = 1 \quad \text{sii} \quad \mathcal{P} \text{ se detiene luego de } t \text{ pasos, partiendo del estado } \|x_1, \dots, x_n, \alpha_1, \dots, \alpha_m\|.$$

Ojo, la notación lambda es respecto al alfabeto  $(\Sigma \cup \Sigma_p)$ .

7. Defina  $T^{n,m}$ 

*Definiciones:*

Dados  $n, m \in \omega$  y como  $Halt^{n,m}$  es un predicado podemos definir

$$T^{n,m} = M(Halt^{n,m})$$

Es decir que para  $(\vec{x}, \vec{\alpha}, \mathcal{P}) \in D_{T^{n,m}}$ :

$$T^{n,m}(\vec{x}, \vec{\alpha}, \mathcal{P}) = \text{cantidad de pasos necesarios para que } \mathcal{P} \text{ se detenga partiendo de } \|x_1, \dots, x_n, \alpha_1, \dots, \alpha_m\|$$

Notar que  $D_{T^{n,m}} = \{(\vec{x}, \vec{\alpha}, \mathcal{P}) : \mathcal{P} \text{ se detiene partiendo de } \|x_1, \dots, x_n, \alpha_1, \dots, \alpha_m\|\}$ .

8. Defina  $AutoHalt^\Sigma$ 

*Definición:*

Cuando  $\Sigma_p \subseteq \Sigma$ , podemos definir

$$AutoHalt^\Sigma = \lambda \mathcal{P} [(\exists t \in \omega) Halt^{0,1}(t, \mathcal{P}, \mathcal{P})]$$

Pensándolo de otra manera, podemos decir que para cada  $\mathcal{P} \in \text{Pro}^\Sigma$  tenemos que

$$AutoHalt^\Sigma(\mathcal{P}) = 1 \text{ si y solo si } \mathcal{P} \text{ se detiene partiendo del estado } \|\mathcal{P}\| \text{ (i.e de sí mismo)}$$

9. Defina los conjuntos  $A$  y  $N$ 

*Definición:*

Dado  $\Sigma_p \subseteq \Sigma$ , definimos

$$A = \{\mathcal{P} \in \text{Pro}^\Sigma : AutoHalt^\Sigma(\mathcal{P}) = 1\}$$

$$N = \{\mathcal{P} \in \text{Pro}^\Sigma : AutoHalt^\Sigma(\mathcal{P}) = 0\}$$

## Combo 14

### 1. Explique en forma detallada la notación lambda.



*Definición:*

Definamos primero cuándo una **expresión**  $E$  será llamada **lambdificable con respecto a  $\Sigma$** . Dado que no es un concepto matemáticamente preciso, daremos características que se deben cumplir

(1) Puede involucrar variables:

(i) Numéricas, valuadas en  $\omega$  y seleccionadas de  $x, y, z, n, m, k, \dots$

$x_1, x_2, \dots$

$y_1, y_2, \dots$

*etc*

(ii) Alfabéticas, valuadas en  $\Sigma^*$  y seleccionadas de  $\alpha, \beta, \gamma, \eta, \dots$

$\alpha_1, \alpha_2, \dots$

$\beta_1, \beta_2, \dots$

*etc*

(2) Puede no involucrar variables, por lo tanto produce un valor constante.

(3) Para ciertas valuaciones de sus variables  $E$  puede no estar definida. (Por ejemplo  $\text{Pred}(x)$  con  $x = 0$ )

(4) Cuando  $E$  esté definida al valuar sus variables numéricas en  $\omega$  y alfabéticas en  $\Sigma^*$  deberá producir siempre un elemento de  $\omega$  o de  $\Sigma^*$ . (Es decir no puede tomar valores mixtos)

(5) Se pueden usar expresiones del lenguaje coloquial castellano. (Por ejemplo “*es x un número primo*”)

(6) Las expresiones booleanas toman valores en  $\{0, 1\} \subseteq \omega$ . (“1 = verdad”, “0 = falso”)

Ahora definamos la **notación lambda**.

Sea un alfabeto fijo y finito  $\Sigma$ ,  $E$  una expresión que sea lambdificable con respecto a  $\Sigma$ . Sean  $x_1, \dots, x_n, \alpha_1, \dots, \alpha_m$  variables distintas tales que, las variables numéricas que ocurren en  $E$  están en  $\{x_1, \dots, x_n\}$  y las variables alfabéticas que ocurren en  $E$  están en  $\{\alpha_1, \dots, \alpha_m\}$ . Entonces

$$\lambda x_1 \dots x_n \alpha_1 \dots \alpha_m [E]$$

denota la función definida por:

- $D_{\lambda x_1 \dots x_n \alpha_1 \dots \alpha_m [E]} = \{(k_1, \dots, k_n, \beta_1, \dots, \beta_m) \in w^n \times \Sigma^{*m} : E \text{ está definida cuando le asignamos a cada } x_i \text{ el valor } k_i \text{ y a cada } \alpha_i \text{ el valor } \beta_i\}$
- $\lambda x_1 \dots x_n \alpha_1 \dots \alpha_m [E](k_1, \dots, k_n, \beta_1, \dots, \beta_m) = \text{valor que asume o representa } E \text{ cuando le asignamos a cada } x_i \text{ el valor } k_i \text{ y a cada } \alpha_i \text{ el valor } \beta_i.$

Notar que por (4)  $\lambda x_1 \dots x_n \alpha_1 \dots \alpha_m [E]$  es  $\Sigma$ -mixta de tipo  $(n, m, s)$  con  $s \in \{\#, *\}$  según corresponda.

## Combo 15

1. Dada una función  $f : D_f \subseteq \omega \times \Sigma^* \rightarrow \omega$ , describa qué tipo de objeto es y que propiedades debe tener el macro  $[V2 \leftarrow f(V1, W1)]$   $\hookrightarrow$

*Definición:*

Dada una función  $f : D_f \subseteq \omega \times \Sigma^* \rightarrow \omega$ , usaremos

$$[V2 \leftarrow f(V1, W1)]$$

para denotar el macro  $M$ , el cual es de tipo palabra y cumple las siguientes propiedades

- (1) Las variables oficiales de  $M$  son  $V1$ ,  $V2$  y  $W1$ .
- (2)  $M$  No tiene labels oficiales.
- (3) Si reemplazamos
  - (a) Las variables oficiales de  $M$  (i.e.  $V1$ ,  $V2$ ,  $W1$ ) por las variables concretas  $\overline{Nk_1}, \overline{Nk_2}$  y  $\overline{Pj_1}$  con  $k_1, k_2, j_1 \in \mathbb{N}$ .
  - (b) Las variables auxiliares de  $M$  por variables concretas (distintas de a dos) y distintas de  $\overline{Nk_1}, \overline{Nk_2}, \overline{Pj_1}$ .
  - (c) Los labels auxiliares de  $M$  por labels concretos (distintos de a dos).

Entonces la palabra así obtenida es un programa de  $S^\Sigma$  que denotaremos con

$$[\overline{Nk_2} \leftarrow f(\overline{Nk_1}, \overline{Pj_1})]$$

el cual debe tener la siguiente propiedad:

Si lo hacemos correr partiendo de un estado  $e$  que le asigne a las variables  $\overline{Nk_1}$  y  $\overline{Pj_1}$  valores  $x_1$  y  $\alpha_1$ , entonces independientemente de los valores que les asigne  $e$  al resto de las variables se dará que

- (i) Si  $(x_1, \alpha_1) \notin D_f$  entonces  $[\overline{Nk_2} \leftarrow f(\overline{Nk_1}, \overline{Pj_1})]$  **no se detiene**.
- (ii) Si  $(x_1, \alpha_1) \in D_f$ , entonces  $[\overline{Nk_2} \leftarrow f(\overline{Nk_1}, \overline{Pj_1})]$  se detiene (i.e. intenta realizar la siguiente a su última instrucción) y llega a un estado  $e'$  el cual cumple
  - (a)  $e'$  le asigna a  $\overline{Nk_2}$  el valor  $f(x_1, \alpha_1)$ .
  - (b)  $e'$  solo puede diferir de  $e$  en los valores que le asigna a  $\overline{Nk_2}$  o a las variables que fueran a reemplazar a las variables auxiliares de  $M$ . Al resto de las variables, no las modifica.

Finalmente el programa  $[\overline{Nk_2} \leftarrow f(\overline{Nk_1}, \overline{Pj_1})]$  es llamado la **expansión del macro**  $[V2 \leftarrow f(V1, W1)]$  con respecto a la elección de variables y labels realizada.

## Combo 16

1. Dado un predicado  $P : D_f \subseteq \omega \times \Sigma^* \rightarrow \omega$ , describa qué tipo de objeto es y qué propiedades debe tener el macro  $[\text{IF } P(V1, W1) \text{ GOTO } A1]$   $\hookrightarrow$

*Definición:*

Dado un predicado  $P : D_f \subseteq \omega \times \Sigma^* \rightarrow \omega$ , usaremos

$$[\text{IF } P(V1, W1) \text{ GOTO } A1]$$

para denotar el macro  $M$ , el cual es de tipo palabra y cumple las siguientes propiedades

- (1) Las variables oficiales de  $M$  son  $V1$  y  $W1$ .
- (2)  $A1$  es el único label oficial de  $M$ .
- (3) Si reemplazamos
  - (a) Las variables oficiales de  $M$  (i.e.  $V1, W1$ ) por las variables concretas  $\overline{Nk_1}$  y  $\overline{Pj_1}$  con  $k_1, j_1 \in \mathbb{N}$ .
  - (b) El label oficial  $A1$  por el label concreto  $\overline{Lk}$  con  $k \in \mathbb{N}$ .
  - (c) Las variables auxiliares de  $M$  por variables concretas (distintas de a dos) y distintas de  $\overline{Nk_1}, \overline{Pj_1}$ .
  - (d) Los labels auxiliares de  $M$  por labels concretos (distintos de a dos) y todos distintos de  $\overline{Lk}$ .

Entonces la palabra así obtenida es un programa de  $S^\Sigma$ , salvo por la ley de los GOTO respecto de  $\overline{Lk}$ . Este programa lo denotaremos con

$$[\text{IF } P(\overline{Nk_1}, \overline{Pj_1}) \text{ GOTO } \overline{Lk}]$$

el cual debe tener la siguiente propiedad:

Si lo hacemos correr partiendo de un estado  $e$  que le asigne a las variables  $\overline{Nk_1}$  y  $\overline{Pj_1}$  valores  $x_1$  y  $\alpha_1$ , entonces independientemente de los valores que le asigne  $e$  al resto de las variables se dará que

- (i) Si  $(x_1, \alpha_1) \notin D_P$  entonces  $[\text{IF } P(\overline{Nk_1}, \overline{Pj_1}) \text{ GOTO } \overline{Lk}]$  **no se detiene**.
- (ii) Si  $(x_1, \alpha_1) \in D_P$  entonces luego de una cantidad finita de pasos de  $[\text{IF } P(\overline{Nk_1}, \overline{Pj_1}) \text{ GOTO } \overline{Lk}]$ 
  - (a) Si  $P(x_1, \alpha_1) = 1$ , **direcciona al label  $\overline{Lk}$** .
  - (b) Si  $P(x_1, \alpha_1) = 0$ , **se detiene**. (i.e. intenta realizar la siguiente a su última instrucción)

En ambos casos quedándose en un estado  $e'$  el cual solo puede diferir de  $e$  en los valores que le asigna a las variables que fueran a reemplazar a las variables auxiliares de  $M$ . Al resto de las variables, no las modifica.

8

Finalmente, el programa  $[\text{IF } P(\overline{Nk_1}, \overline{Pj_1}) \text{ GOTO } \overline{Lk}]$  es llamado la **expansión del macro**  $[\text{IF } P(V1, W1) \text{ GOTO } A1]$  con respecto a la elección de variables y labels realizada.

<sup>8</sup>El punto (ii) en el apunte está así

- (ii) Si  $(x_1, \alpha_1) \in D_P$  y  $P(x_1, \alpha_1) = 1$ , entonces luego de una cantidad finita de pasos  $[\text{IF } P(\overline{Nk_1}, \overline{Pj_1}) \text{ GOTO } \overline{Lk}]$  **direcciona al label  $\overline{Lk}$**  quedándose en un estado  $e'$  el cual solo puede diferir de  $e$  en los valores que le asigna a las variables que fueran a reemplazar a las variables auxiliares, al resto de las variables, no las modifica.
- (iii) Si  $(x_1, \alpha_1) \in D_P$  y  $P(x_1, \alpha_1) = 0$ , entonces luego de una cantidad finita de pasos  $[\text{IF } P(\overline{Nk_1}, \overline{Pj_1}) \text{ GOTO } \overline{Lk}]$  **se detiene** quedando en un estado  $e'$  el cual solo puede diferir de  $e$  en los valores que le asigna a las variables que fueran a reemplazar a las variables auxiliares, al resto de las variables, no las modifica.

## Combo 17

1. Defina el concepto de función y desarrolle las tres Convenciones Notacionales asociadas a dicho concepto.

*Nota:* de la Guía 1



*Definición:*

Una **función** es un conjunto  $f$  de pares ordenados con la siguiente propiedad

$$\text{Si } (x, y) \in f \text{ y } (x, z) \in f, \text{ entonces } y = z$$

Además, dada una función  $f$  definimos

$$D_f = \text{dominio de } f = \{x : (x, y) \in f \text{ para algún } y\}$$

$$I_f = \text{imagen de } f = \{y : (x, y) \in f \text{ para algún } x\}$$

A veces escribimos  $\text{Dom}(f)$  y  $\text{Im}(f)$  en lugar de  $D_f$  e  $I_f$ , respectivamente.

Las **convenciones notacionales** son

- (1) Dado  $x \in D_f$  usaremos  $f(x)$  para denotar el único  $y \in I_f$  tal que  $(x, y) \in f$ .
- (2) Escribimos  $f : S \subseteq A \rightarrow B$  para expresar que  $f$  es una función tal que  $D_f = S \subseteq A$  y  $I_f \subseteq B$ .  
Escribimos  $f : A \rightarrow B$  para expresar que  $f$  es una función tal que  $D_f = A$  y  $I_f \subseteq B$ .  
En ese contexto llamaremos a  $B$  *conjunto de llegada* ( $B$  no está determinado por  $f$ , ya que  $I_f \subseteq B$ ).
- (3) Muchas veces, para definir una función  $f$ , lo que haremos es dar su dominio y su regla de asignación. Básicamente daremos precisamente el conjunto que es  $D_f$  y quién es  $f(x)$  para cada  $x \in D_f$ . Esto determina por completo a  $f$ , ya que  $f = \{(x, f(x)) : x \in D_f\}$ . Algunos ejemplos son

Básico	Con <i>conjunto de llegada</i> y flechas	Con flechas y por casos
$D_f = \omega$	$f : \omega \rightarrow \omega$	$f : \mathbb{N} \rightarrow \omega$
$f(x) = 23 \cdot x$	$x \rightarrow 23 \cdot x$	$x \rightarrow \begin{cases} x + 1 & \text{si } x \text{ es par} \\ x + 2 & \text{si } x \text{ es impar} \end{cases}$

## Definiciones Auxiliares

**Aux.** Procedimiento efectivo  $\mathbb{P}$ .



*Definición:*

Llamaremos **procedimientos efectivos**  $\mathbb{P}$  a aquellos que posean las siguientes características

- (1) El ejecutante de  $\mathbb{P}$  es una persona que trabajará con papel y lápiz disponibles en forma ilimitada.
- (2) Cada paso o tarea de  $\mathbb{P}$  debe ser simple y fácil de realizar en forma efectiva por cualquier persona.
- (3) El procedimiento  $\mathbb{P}$  comienza con cierto dato de entrada y sigue uno de dos posibles comportamientos
  - (a)  $\mathbb{P}$  luego de cierta cantidad de pasos realizados, se detiene y da cierto dato de salida.
  - (b)  $\mathbb{P}$  nunca se detiene, generando tareas sucesivamente sin fin.

Diremos que  $\mathbb{P}$  *se detiene* (caso a) o *no se detiene* (caso b) partiendo del dato de entrada en cuestión.

- (4) Hay  $n, m \in \omega$  y un alfabeto  $\Sigma$  tales que el conjunto de datos de entrada de  $\mathbb{P}$  es  $w^n \times \Sigma^{*m}$ .

Para ciertas  $(n + m)$ -uplas de  $w^n \times \Sigma^{*m}$  el procedimiento  $\mathbb{P}$  se detendrá y para otras no.

Esta definición está con otras palabras. ([Definición original en el apunte](#))

# TEOREMAS

## Combo 1

### 1. Proposición (Caracterización de conjuntos $\Sigma$ -p.r.) .

Un conjunto  $S$  es  $\Sigma$ -p.r. sii  $S$  es el dominio de alguna función  $\Sigma$ -p.r.

Nota: en la inducción de la prueba hacer solo el caso de la composición. ↪

*Demostración:*

( $\Rightarrow$ ) Notar que  $S = D_{\text{Pred} \circ \chi_S^{w^n \times \Sigma^{*m}}}$  y  $\text{Pred} \circ \chi_S^{w^n \times \Sigma^{*m}}$  es claramente  $\Sigma$ -p.r. ya que  $S$  lo es.

( $\Leftarrow$ ) Probaremos por inducción sobre  $k$  que para cada  $F \in \text{PR}_k^\Sigma$ ,  $D_F$  es  $\Sigma$ -p.r.

El caso  $k = 0$ , es fácil ya que  $\text{PR}_0^\Sigma = \{\text{Suc}, \text{Pred}, C_0^{0,0}, C_\varepsilon^{0,0}\} \cup \{d_a : a \in \Sigma\} \cup \{p_j^{n,m} : 1 \leq j \leq n+m\}$  y  $D_{\text{Suc}} = \omega$ ,  $D_{\text{Pred}} = \mathbb{N}$ ,  $D_{C_0^{0,0}} = D_{C_\varepsilon^{0,0}} = \{\diamond\}$  y  $D_{p_j^{n,m}} = w^n \times \Sigma^{*m}$ , que claramente son todos  $\Sigma$ -p.r.

Por lo tanto supongamos que vale para un  $k$  fijo y veamos que se cumple también para  $F \in \text{PR}_{k+1}^\Sigma$ .

Hay varios casos. Veamos el caso que  $F = g \circ [g_1, \dots, g_r]$  con  $g, g_1, \dots, g_r \in \text{PR}_k^\Sigma$ .

Si  $F = \emptyset$ , entonces es claro que  $D_F$  es  $\Sigma$ -p.r.

Si  $F \neq \emptyset$ , tenemos entonces que  $r$  es de la forma  $n + m$  y

$$g : D_g \subseteq w^n \times \Sigma^{*m} \rightarrow O$$

$$g_i : D_{g_i} \subseteq \omega^k \times \Sigma^{*l} \rightarrow \omega, \text{ para cada } i = 1, \dots, n$$

$$g_i : D_{g_i} \subseteq \omega^k \times \Sigma^{*l} \rightarrow \Sigma^*, \text{ para cada } i = n+1, \dots, n+m$$

con  $O \in \{\omega, \Sigma^*\}$  y  $k, l \in \omega$ . Por el Lema (1), hay funciones  $\Sigma$ -p.r.  $\bar{g}_1, \dots, \bar{g}_{n+m}$  las cuales son  $\Sigma$ -totales y

$$g_i = \bar{g}_i \upharpoonright_{D_{g_i}} \text{ para } i = 1, \dots, n+m$$

Por hipótesis inductiva los conjuntos  $D_g, D_{g_i}$ , con  $i = 1, \dots, n+m$ , son  $\Sigma$ -p.r. y por lo tanto también

$$S = \bigcap_{i=1}^{n+m} D_{g_i}$$

Notar que  $\chi_{D_F}^{w^k \times \Sigma^{*l}} = \left( \chi_{D_g}^{w^n \times \Sigma^{*m}} \circ [\bar{g}_1, \dots, \bar{g}_{n+m}] \wedge \chi_S^{w^k \times \Sigma^{*l}} \right)$  lo cual nos dice que  $D_F$  es  $\Sigma$ -p.r. ■

*Lema (1):* Sea  $O = \{\omega, \Sigma^*\}$  y  $n, m \in \omega$ . Si  $f : D_f \subseteq w^n \times \Sigma^{*m} \rightarrow O$  es  $\Sigma$ -p.r., entonces existe una función  $\Sigma$ -p.r.  $\bar{f} : w^n \times \Sigma^{*m} \rightarrow O$  tal que  $f = \bar{f} \upharpoonright_{D_f}$ . (En la guía 5, es el lema 18 y en el [apunte el 4.17](#))

<sup>9</sup>El truco está en que  $D_{\text{Pred}} = \omega - \{0\}$  por lo tanto cada vez que  $(\vec{x}, \vec{\alpha}) \notin S$  (i.e.  $\chi_S^{w^n \times \Sigma^{*m}}(\vec{x}, \vec{\alpha}) = 0$ ),  $(\vec{x}, \vec{\alpha}) \notin D_{\text{Pred} \circ \chi_S^{w^n \times \Sigma^{*m}}}$

**2. Teorema** (Neumann vence a Gödel) . Si  $h$  es  $\Sigma$ -recursiva, entonces  $h$  es  $\Sigma$ -computable.

*Nota:* en la inducción de la prueba hacer solo el caso  $h = R(f, \mathcal{G})$ , con  $I_h \subseteq \omega$

$\hookrightarrow$

*Demostración:*

Esto será probado por inducción en  $k$  que si  $h \in R_k^\Sigma$ , entonces  $h$  es  $\Sigma$ -computable .

El caso  $k = 0$ , es fácil ya que  $R_0 = \text{PR}_0$ , entonces hay que hacer programas que computen

$\{\text{Suc}, \text{Pred}, C_0^{0,0}, C_\varepsilon^{0,0}\} \cup \{d_a : a \in \Sigma\} \cup \{p_j^{n,m} : 1 \leq j \leq n+m\}$ , los cuales son todos triviales

Suc	Pred	$C_0^{0,0}$	$C_\varepsilon^{0,0}$	$d_a$	$p_j^{n,m}$
$N1 \leftarrow N1 + 1$	IF $N1 \neq 0$ GOTO L2 L1 GOTO L1 L2 $N1 \leftarrow N1 - 1$	$N1 \leftarrow 0$	$P1 \leftarrow \varepsilon$	$P1 \leftarrow P1.a$	$N1 \leftarrow N\bar{j}$ $o$ $P1 \leftarrow P\bar{j}$

Supongamos que la propiedad se cumple para un  $k$  fijo y veamos que se cumple también para  $h \in R_{k+1}^\Sigma$ .

Hay varios casos. Veamos el caso que  $h = R(f, \mathcal{G})$  con  $f \in R_k^\Sigma$  y  $\mathcal{G} \in R_k^\Sigma$  que son

$$f : S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \omega$$

$$\mathcal{G}_a : \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \times \Sigma^* \rightarrow \omega, \quad a \in \Sigma$$

Sea  $\Sigma = \{a_1, \dots, a_r\}$ . Por hipótesis inductiva, las funciones  $f, \mathcal{G}_a$  con  $a \in \Sigma$ , son  $\Sigma$ -computables y por lo tanto tenemos las macros

$$[\overline{Vn+1} \leftarrow f(\overline{V1}, \dots, \overline{Vn}, \overline{W1}, \dots, \overline{Wm})]$$

$$[\overline{Vn+2} \leftarrow \mathcal{G}_{a_i}(\overline{V1}, \dots, \overline{Vn+1}, \overline{W1}, \dots, \overline{Wm+1})] \text{ con } i = 1, \dots, r$$

Podemos entonces hacer el siguiente programa (tener en cuenta que  $\overline{Pm+2} = \varepsilon$  al iniciar)

$$\begin{aligned}
 & [ \overline{Nn+1} \leftarrow f(\overline{N1}, \dots, \overline{Nn}, \overline{P1}, \dots, \overline{Pm}) ] \\
 \overline{Lr+1} \quad & \text{IF } \overline{Pm+1} \text{ BEGINS } a_1 \text{ GOTO L1} \\
 & \vdots \\
 & \text{IF } \overline{Pm+1} \text{ BEGINS } a_r \text{ GOTO } \overline{Lr} \\
 & \text{GOTO } \overline{Lr+2} \\
 \text{L1} \quad & \overline{Pm+1} \leftarrow \sim \overline{Pm+1} \\
 & [ \overline{Nn+1} \leftarrow \mathcal{G}_{a_1}(\overline{Nn+1}, \overline{N1}, \dots, \overline{Nn}, \overline{P1}, \dots, \overline{Pm}, \overline{Pm+2}) ] \\
 & \overline{Pm+2} \leftarrow \overline{Pm+2} . a_1 \\
 & \text{GOTO } \overline{Lr+1} \\
 & \vdots \\
 \overline{Lr} \quad & \overline{Pm+1} \leftarrow \sim \overline{Pm+1} \\
 & [ \overline{Nn+1} \leftarrow \mathcal{G}_{a_r}(\overline{Nn+1}, \overline{N1}, \dots, \overline{Nn}, \overline{P1}, \dots, \overline{Pm}, \overline{Pm+2}) ] \\
 & \overline{Pm+2} \leftarrow \overline{Pm+2} . a_r \\
 & \text{GOTO } \overline{Lr+1} \\
 \overline{Lr+2} \quad & \overline{N1} \leftarrow \overline{Nn+1}
 \end{aligned}$$

es fácil ver que el programa computa  $h^{10}$ , por lo tanto  **$h$  es  $\Sigma$ -computable.** (El resto de casos no los pide) ■

<sup>10</sup>Recordar que  $R(f, \mathcal{G})(\vec{x}, \vec{\alpha}, \alpha a) = \mathcal{G}_a(R(f, \mathcal{G})(\vec{x}, \vec{\alpha}, \alpha), \vec{x}, \vec{\alpha}, \alpha)$ . La idea es que, por ejemplo para  $\vec{x}, \vec{\alpha}$  y  $\alpha = a_1 a_2 a_3$  tenemos que  $\mathcal{G}_{a_3}(\mathcal{G}_{a_2}(\mathcal{G}_{a_1}(f(\vec{x}, \vec{\alpha}), \vec{x}, \vec{\alpha}, \varepsilon), \vec{x}, \vec{\alpha}, a_1), \vec{x}, \vec{\alpha}, a_1 a_2))$  por esto calculamos primero  $f$  y después en  $\overline{Pm+1}$  vamos llevando los  $\varepsilon, a_1 a_2, a_1 a_2 a_3$

## Combo 2

- 1. Lema** (Lema de división por casos para funciones  $\Sigma$ -p.r.) . Supongamos  $f_i : D_{f_i} \subseteq w^n \times \Sigma^{*m} \rightarrow \Sigma^*$ , con  $i = 1, \dots, k$ , son funciones  $\Sigma$ -p.r. tales que  $D_{f_i} \cap D_{f_j} = \emptyset$  para  $i \neq j$ . Entonces  $f_1 \cup \dots \cup f_k$  es  $\Sigma$ -p.r.  
 Nota: hacer el caso  $k = 2, n = 2$  y  $m = 1$   $\hookrightarrow$

*Demostración:*

Supongamos  $k = 2, n = 2$  y  $m = 1$ . Por lo tanto tenemos que

$$\bar{f}_i : \omega \times \omega \times \Sigma^* \rightarrow \Sigma^*, i = 1, 2$$

son funciones  $\Sigma$ -p.r. tales que  $f_i = \bar{f}_i \upharpoonright_{D_{f_i}}, i = 1, 2$  por el Lema (1). Luego por la Proposición (Caracterización de Conjuntos  $\Sigma$ -p.r.), los conjuntos  $D_{f_1}$  y  $D_{f_2}$  son  $\Sigma$ -p.r. y por lo tanto por el Lema (o.p de Conjuntos  $\Sigma$ -p.r.),  $D_{f_1} \cup D_{f_2}$  también. Finalmente dado que

$$f_1 \cup f_2 = \left( \lambda \alpha \beta [\alpha \beta] \circ \left[ \lambda x \alpha [\alpha^x] \circ \left[ \chi_{D_{f_1}}^{w^n \times \Sigma^{*m}}, \bar{f}_1 \right] \quad , \quad \lambda x \alpha [\alpha^x] \circ \left[ \chi_{D_{f_2}}^{w^n \times \Sigma^{*m}}, \bar{f}_2 \right] \right] \right) \upharpoonright_{D_{f_1} \cup D_{f_2}}$$

y el Lema (Restricción de Dominios  $\Sigma$ -p.r.), tenemos que  $f_1 \cup f_2$  es  $\Sigma$ -p.r.. ■

*Lema (1):* Sea  $O = \{\omega, \Sigma^*\}$  y  $n, m \in \omega$ . Si  $f : D_f \subseteq w^n \times \Sigma^{*m} \rightarrow O$  es  $\Sigma$ -p.r., entonces existe una función  $\Sigma$ -p.r.  $\bar{f} : w^n \times \Sigma^{*m} \rightarrow O$  tal que  $f = \bar{f} \upharpoonright_{D_f}$ . (En la guía 5, es el lema 18 y en el apunte el 4.17 )

## 2. Proposición (Caracterización básica de conjuntos $\Sigma$ -enumerables).

Sea  $S \subseteq \omega^n \times \Sigma^{*m}$  un conjunto no vacío. Entonces son equivalentes

- (1)  $S$  es  $\Sigma$ -enumerable.
- (2) Hay un programa  $\mathcal{P} \in \text{Pro}^\Sigma$  tal que
  - (a) Para cada  $x \in \omega$ , tenemos que  $\mathcal{P}$  se detiene partiendo desde el estado  $\|x\|$  y llega a un estado de la forma  $((x_1, \dots, x_n, y_1, \dots), (\alpha_1, \dots, \alpha_m, \beta_1, \dots))$  donde  $(x_1, \dots, x_n, \alpha_1, \dots, \alpha_m) \in S$ .
  - (b) Para cada  $(x_1, \dots, x_n, \alpha_1, \dots, \alpha_m) \in S$  hay un  $x \in \omega$  tal que  $\mathcal{P}$  se detiene partiendo desde el estado  $\|x\|$  y llega a un estado de la forma  $((x_1, \dots, x_n, y_1, \dots), (\alpha_1, \dots, \alpha_m, \beta_1, \dots))$ .

Nota: hacer el caso  $n = 2$  y  $m = 1$

↳

*Demostración:*

Haremos el caso  $n = 2$  y  $m = 1$ , es decir  $S \subseteq \omega \times \omega \times \Sigma^*$ .

(1)  $\implies$  (2)

Ya que  $S$  es no vacío, por definición hay una  $F : \omega \rightarrow \omega \times \omega \times \Sigma^*$  tal que  $I_F = S$  y  $F_{(i)}$  es  $\Sigma$ -computable, para cada  $i = 1, 2, 3$ . Entonces por el Primer Manantial existen los macros

$$[V2 \leftarrow F_{(1)}(V1)] \quad [V2 \leftarrow F_{(2)}(V1)] \quad [W1 \leftarrow F_{(3)}(V1)]$$

Y damos el programa  $\mathcal{P}$

$$\begin{aligned} [P1 \leftarrow F_{(3)}(N1)] \\ [N2 \leftarrow F_{(2)}(N1)] \\ [N1 \leftarrow F_{(1)}(N1)] \end{aligned}$$

Donde se supone que las expansiones de los macros usan variables auxiliares que no aparecen en la lista  $N1, N2, P1$  y tampoco repiten labels auxiliares.

Notar que para cada  $x \in \omega$ ,  $\mathcal{P}$  siempre se detiene partiendo de un estado  $\|x\|$  porque las  $F_{(i)}$  son  $\Sigma$ -totales, entonces sus macros siempre se detienen. Luego es fácil ver que **cumplen las condiciones**

- (a) Porque ya sabemos que para cada  $x \in \omega$ ,  $\mathcal{P}$  se detiene partiendo desde el estado  $\|x\|$  y además como  $F(x) = (F_{(1)}(x), F_{(2)}(x), F_{(3)}(x)) = (x_1, x_2, \alpha_1) \in S$  entonces  $\mathcal{P}$  se detiene con un estado de la forma  $((x_1, x_2, y_1, \dots), (\alpha_1, \beta_1, \dots))$  donde  $(x_1, x_2, \alpha_1) \in S$ .
- (b) Porque para cada  $(x_1, x_2, \alpha_1) \in S$ , sabemos por definición que existe un  $x \in \omega$  tal que  $F(x) = (F_{(1)}(x), F_{(2)}(x), F_{(3)}(x)) = (x_1, x_2, \alpha_1)$ . Y como  $\mathcal{P}$  siempre se detiene, incluso partiendo del estado  $\|x\|$ , llegará a un estado de la forma  $((x_1, x_2, y_1, \dots), (\alpha_1, \beta_1, \dots))$ .

(1)  $\Leftarrow$  (2)

Supongamos  $\mathcal{P} \in \text{Pro}^\Sigma$  que cumple (a) y (b) de (2).

Sean

$$\mathcal{P}_1 = \mathcal{P} \ N1 \leftarrow N1 \quad \mathcal{P}_2 = \mathcal{P} \ N1 \leftarrow N2 \quad \mathcal{P}_3 = \mathcal{P} \ P1 \leftarrow P1$$

definamos las funciones

$$F_1 = \Psi_{\mathcal{P}_1}^{1,0,\#} \quad F_2 = \Psi_{\mathcal{P}_2}^{1,0,\#} \quad F_3 = \Psi_{\mathcal{P}_3}^{1,0,*}$$

Notar que cada  $F_i$  es  $\Sigma$ -computable y tienen dominio igual a  $\omega$ . Sea  $F = [F_1, F_2, F_3]$ , por definición  $D_F = \omega$  y ya que  $F_i = F_{(i)}$ , para cada  $i = 1, 2, 3$  tenemos que cada  $F_{(i)}$  es  $\Sigma$ -computable. Resta ver que  $I_F = S$

$I_F \subseteq S$  Por (a) tenemos que  $\mathcal{P}$  para todo  $x \in \omega$  partiendo de  $\|x\|$  llega a un estado de la forma  $((x_1, x_2, y_1, \dots), (\alpha_1, \beta_1, \dots))$  donde  $(x_1, x_2, \alpha_1) \in S$ . Por lo tanto  $F(x) = (F_1(x), F_2(x), F_3(x)) = (x_1, x_2, \alpha_1) \in S$  entonces  $I_F \subseteq S$ .

$S \subseteq I_F$  Por (b) tenemos que para cada  $(x_1, x_2, \alpha_1) \in S$ , hay un  $x \in \omega$  tal que  $\mathcal{P}$  partiendo de  $\|x\|$  llega a un estado de la forma  $((x_1, x_2, y_1, \dots), (\alpha_1, \beta_1, \dots))$ . Por lo tanto  $F(x) = (F_1(x), F_2(x), F_3(x)) = (x_1, x_2, \alpha_1) \in I_F$  entonces  $S \subseteq I_F$ .

Entonces finalmente  $I_F = S$  y por lo tanto  $S$  es  $\Sigma$ -enumerable. ■

## Combo 3

### 1. Teorema (Gödel vence a Neumann).

Si  $f : D_f \subseteq w^n \times \Sigma^{*m} \rightarrow \Sigma^*$  es  $\Sigma$ -computable, entonces  $f$  es  $\Sigma$ -recursiva.  $\hookrightarrow$

*Demostración:*

Sea  $\mathcal{P}_0$  un programa que compute a  $f$ . Primero veamos que  $f$  es  $(\Sigma \cup \Sigma_p)$ -recursiva. Notar que<sup>11</sup>

$$f = E_{*1}^{n,m} \circ [T^{n,m} \circ [p_1^{n,m}, \dots, p_{n+m}^{n,m}, C_{\mathcal{P}_0}^{n,m}], p_1^{n,m}, \dots, p_{n+m}^{n,m}, C_{\mathcal{P}_0}^{n,m}]$$

donde  $p_1^{n,m}, \dots, p_{n+m}^{n,m}$  y  $C_{\mathcal{P}_0}^{n,m}$  son respecto al alfabeto  $\Sigma \cup \Sigma_p$ , es decir que tienen dominio  $\omega^n \times (\Sigma \cup \Sigma_p)^{*m}$ . Esto nos dice que  $f$  es  $(\Sigma \cup \Sigma_p)$ -recursiva. Osea que el Teorema (Independencia del Alfabeto) nos dice que  **$f$  es  $\Sigma$ -recursiva.** ■

### 2. Teorema (Caracterización de conjuntos $\Sigma$ -efectivamente computable).

Sea  $S \subseteq w^n \times \Sigma^{*m}$ . Son equivalentes

(a)  $S$  es  $\Sigma$ -efectivamente computable.

(b)  $S$  y  $(w^n \times \Sigma^{*m}) - S$  son  $\Sigma$ -efectivamente enumerables.

Nota: haga solo (b)  $\implies$  (a). La prueba está al final de la Guía 3  $\hookrightarrow$

*Demostración:*

(b)  $\implies$  (a)

Si  $S = \emptyset$  o  $S = w^n \times \Sigma^{*m}$  es claro que se cumple (a).

Así que supongamos que  $S \neq \emptyset$  y  $S \neq w^n \times \Sigma^{*m}$  por lo cual  $(w^n \times \Sigma^{*m}) - S \neq \emptyset$ .

Además sea  $\mathbb{P}_1$  un procedimiento efectivo que enumere a  $S$  y  $\mathbb{P}_2$  un procedimiento efectivo que enumere a  $(w^n \times \Sigma^{*m}) - S$ . Ahora sí es fácil ver que el siguiente procedimiento efectivo  $\mathbb{P}$  computa  $\chi_S^{w^n \times \Sigma^{*m}}$

Sea  $(\vec{x}, \vec{\alpha}) \in w^n \times \Sigma^{*m}$ .

Etapas 1

Darle a la variable  $T$  el valor 0.

Etapas 2

Realizar  $\mathbb{P}_1$  con el valor  $T$  como entrada para obtener de salida la ulpa  $(\vec{y}, \vec{\beta})$ .

Etapas 3

Realizar  $\mathbb{P}_2$  con el valor  $T$  como entrada para obtener de salida la ulpa  $(\vec{z}, \vec{\gamma})$ .

Etapas 4

Si  $(\vec{y}, \vec{\beta}) = (\vec{x}, \vec{\alpha})$ , entonces detenerse y dar como dato de salida el valor 1.

Si  $(\vec{z}, \vec{\gamma}) = (\vec{x}, \vec{\alpha})$ , entonces detenerse y dar como dato de salida el valor 0.

Si no sucede ninguna de las dos, aumentar  $T$  en 1 y volver a la Etapa 2.

ya que, los procedimientos  $\mathbb{P}_1$  y  $\mathbb{P}_2$  siempre terminan y además para todo  $(\vec{x}, \vec{\alpha}) \in w^n \times \Sigma^{*m}$  se cumple que  $(\vec{x}, \vec{\alpha}) \in S$  o  $(\vec{x}, \vec{\alpha}) \notin S$ . Osea que siempre existe algún  $t \in \omega$  tal que haga detenerse a  $\mathbb{P}$  dando como dato de salida 1 o 0 respectivamente.

Entonces  **$S$  es  $\Sigma$ -efectivamente computable.** ■

<sup>11</sup>

$$f = \underbrace{E_{*1}^{n,m}}_{\text{resultado de P1, importante (*)}} \circ \left[ \underbrace{T^{n,m} \circ [p_1^{n,m}, \dots, p_{n+m}^{n,m}, C_{\mathcal{P}_0}^{n,m}]}_{\text{cantidad de pasos para que termine}}, \underbrace{p_1^{n,m}, \dots, p_{n+m}^{n,m}}_{\text{input}}, \underbrace{C_{\mathcal{P}_0}^{n,m}}_{\text{programa}} \right]$$

## Combo 4

### 1. Proposición (misma que la del combo 2).

↳

### 2. Lema (Lema de la sumatoria).

Sea  $\Sigma$  un alfabeto finito. Si  $f : \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \omega$  es  $\Sigma$ -p.r., con  $S_1, \dots, S_n \subseteq \omega$  y  $L_1, \dots, L_m \subseteq \Sigma^*$  no vacíos, entonces la función  $\lambda xy\vec{x}\vec{\alpha} \left[ \sum_{t=x}^{t=y} f(t, \vec{x}, \vec{\alpha}) \right]$  es  $\Sigma$ -p.r. ↳

*Demostración:*

Sea  $G = \lambda tx\vec{x}\vec{\alpha} \left[ \sum_{i=x}^{i=t} f(i, \vec{x}, \vec{\alpha}) \right]$ . Ya que (es un simple cambio de lugar las variables)

$$\lambda xy\vec{x}\vec{\alpha} \left[ \sum_{t=x}^{t=y} f(t, \vec{x}, \vec{\alpha}) \right] = G \circ [p_2^{n+2,m}, p_1^{n+2,m}, p_3^{n+2,m}, \dots, p_{n+m+2}^{n+2,m}]$$

basta probar que  $G$  es  $\Sigma$ -p.r. Primero notar que

$$G(0, x, \vec{x}, \vec{\alpha}) = \begin{cases} 0 & \text{si } x > 0 \\ f(0, \vec{x}, \vec{\alpha}) & \text{si } x = 0 \end{cases}$$

$$G(t+1, x, \vec{x}, \vec{\alpha}) = \begin{cases} 0 & \text{si } x > t+1 \\ G(t, x, \vec{x}, \vec{\alpha}) + f(t+1, \vec{x}, \vec{\alpha}) & \text{si } x \leq t+1 \end{cases}$$

osea que si definimos

$$h : \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \omega$$

$$(x, \vec{x}, \vec{\alpha}) \rightarrow \begin{cases} 0 & \text{si } x > 0 \\ f(0, \vec{x}, \vec{\alpha}) & \text{si } x = 0 \end{cases}$$

$$g : \omega^3 \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \omega$$

$$(A, t, x, \vec{x}, \vec{\alpha}) \rightarrow \begin{cases} 0 & \text{si } x > t+1 \\ A + f(t+1, \vec{x}, \vec{\alpha}) & \text{si } x \leq t+1 \end{cases}$$

tenemos que  $G = R(h, g)$ . Es decir que sólo nos falta probar que  $h$  y  $g$  son  $\Sigma$ -p.r. Sean así

$$D_1 = \{(x, x_1, x_2, \alpha_1) \in \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m : x > 0\}$$

$$D_2 = \{(x, x_1, x_2, \alpha_1) \in \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m : x = 0\}$$

$$H_1 = \{(A, t, x, x_1, x_2, \alpha_1) \in \omega^3 \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m : x > t+1\}$$

$$H_2 = \{(A, t, x, x_1, x_2, \alpha_1) \in \omega^3 \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m : x \leq t+1\}$$

Notar que

$$h = C_0^{n+1,m} \upharpoonright_{D_1} \cup \lambda x\vec{x}\vec{\alpha} [f(0, \vec{x}, \vec{\alpha})] \upharpoonright_{D_2}$$

$$g = C_0^{n+3,m} \upharpoonright_{H_1} \cup \lambda Atx\vec{x}\vec{\alpha} [A + f(t+1, \vec{x}, \vec{\alpha})] \upharpoonright_{H_2}$$

Para probarlo, vamos a ver que todas las funciones y conjuntos que aparecen en  $h$  y  $g$  son  $\Sigma$ -p.r.

Trivialmente  $C_0^{n+1,m}$  y  $C_0^{n+3,m}$  son  $\Sigma$ -p.r. Luego como  $f$  es  $\Sigma$ -p.r. y

$$\lambda x \vec{x} \vec{\alpha} [f(0, \vec{x}, \vec{\alpha})] = f \circ [C_0^{n+1,m}, p_2^{n+1,m}, \dots, p_{n+1+m}^{n+1,m}]$$

$$\lambda A t x \vec{x} \vec{\alpha} [A + f(t+1, \vec{x}, \vec{\alpha})] = \lambda x y [x + y] \circ [p_1^{n+3,m}, f \circ [\text{Suc} \circ p_2^{n+3,m}, p_4^{n+3,m}, \dots, p_{n+3+m}^{n+3,m}]]$$

entonces tenemos que ambas funciones son  $\Sigma$ -p.r. Resta ver que los conjuntos  $D_1, D_2, H_1$  y  $H_2$  son  $\Sigma$ -p.r.

- $D_1$  y  $D_2$  : Como  $f$  es  $\Sigma$ -p.r., por la Proposición (Caracterización de Conjuntos  $\Sigma$ -p.r.), tengo que  $D_f = \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m$  también es  $\Sigma$ -p.r. Ahora como

$$\chi_{D_1}^{\omega^{n+1} \times \Sigma^{*m}} = \left( \chi_{D_f}^{\omega^{n+1} \times \Sigma^{*m}} \wedge \lambda x \vec{x} \vec{\alpha} [x > 0] \right)$$

$$\chi_{D_2}^{\omega^{n+1} \times \Sigma^{*m}} = \left( \chi_{D_f}^{\omega^{n+1} \times \Sigma^{*m}} \wedge \lambda x \vec{x} \vec{\alpha} [x = 0] \right)$$

por el Lema (o.p de Predicados  $\Sigma$ -p.r.), tenemos que  $D_1$  y  $D_2$  son  $\Sigma$ -p.r. .

- $H_1$  y  $H_2$  : Como  $f$  es  $\Sigma$ -p.r., por la Proposición (Caracterización de Conjuntos  $\Sigma$ -p.r.), tengo que  $D_f = \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m$  también es  $\Sigma$ -p.r..

Entonces no es difícil ver que  $R = \omega^3 \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m$  es  $\Sigma$ -p.r. usando el Lema (Caracterización de Conjuntos Rectangulares  $\Sigma$ -p.r.). Ahora como

$$\chi_{H_1}^{\omega^{n+3} \times \Sigma^{*m}} = \left( \chi_R^{\omega^{n+3} \times \Sigma^{*m}} \wedge \lambda A t x \vec{x} \vec{\alpha} [x > t + 1] \right)$$

$$\chi_{H_2}^{\omega^{n+3} \times \Sigma^{*m}} = \left( \chi_R^{\omega^{n+3} \times \Sigma^{*m}} \wedge \lambda A t x \vec{x} \vec{\alpha} [x \leq t + 1] \right)$$

por el Lema (o.p de Predicados  $\Sigma$ -p.r.), tenemos que  $H_1$  y  $H_2$  son  $\Sigma$ -p.r. .

Juntando todo por el Lema (Restricción de Dominios  $\Sigma$ -p.r.), todas las funciones usadas en  $h$  y  $g$  son  $\Sigma$ -p.r., pero notar que  $D_1 \cap D_2 = \emptyset$  y  $H_1 \cap H_2 = \emptyset$ , entonces por el Lema (División por Casos para funciones  $\Sigma$ -p.r.), tenemos que  $h$  y  $g$  son  $\Sigma$ -p.r. Por lo tanto  $G = R(h, g)$  es  $\Sigma$ -p.r. ■

## Combo 5

- 1. Lema.** Sea  $\Sigma = \{@, \%, !\}$ . Sea  $f : S_1 \times S_2 \times L_1 \times L_2 \rightarrow \omega$  con  $S_1, S_2 \subseteq \omega$  y  $L_1, L_2 \subseteq \Sigma^*$  no vacíos y sea  $\mathcal{G}$  una familia  $\Sigma$ -indexada de funciones tal que  $\mathcal{G}_a : \omega \times S_1 \times S_2 \times L_1 \times L_2 \times \Sigma^* \rightarrow \omega$  para cada  $a \in \Sigma$ . Si  $f$  y cada  $\mathcal{G}_a$  son  $\Sigma$ -efectivamente computables, **entonces**  $R(f, \mathcal{G})$  lo es. Nota: es un ej de la Guía 5.

*Demostración:*

Dado que  $f, \mathcal{G}_@, \mathcal{G}_\%$  y  $\mathcal{G}_!$  son  $\Sigma$ -efectivamente computables, entonces existen programas  $\mathbb{P}_f, \mathbb{P}_@, \mathbb{P}_\%$  y  $\mathbb{P}_!$  que las computan respectivamente. Entonces notar que el siguiente procedimiento efectivo  $\mathbb{P}$  computa  $R(f, \mathcal{G})$

Sea  $(x_1, x_2, \alpha_1, \alpha_2, \alpha) \in S_1 \times S_2 \times L_1 \times L_2 \times \Sigma^*$

Etapa 1

Hago las asignaciones  $I \leftarrow \varepsilon$  y  $J \leftarrow \alpha$ .

Realizar  $\mathbb{P}_f$  con la entrada  $(x_1, x_2, \alpha_1, \alpha_2)$  y guardar la salida en la variable  $A$ .

Etapa 2

Si  $J = \varepsilon$ , entonces detenerse y dar como dato de salida el valor  $A$ .

Si  $J \neq \varepsilon$ , realizar las asignaciones  $B \leftarrow [J]_1$  y  $J \leftarrow \sim J$ .

Ahora según el caso si  $B = @$  voy a la Etapa 3,

si  $B = \%$  voy a la Etapa 4,

si  $B = !$  voy a la Etapa 5.

Etapa 3

Corro el procedimiento  $\mathbb{P}_@$  con la entrada  $(A, x_1, x_2, \alpha_1, \alpha_2, I)$  y guardar la salida en la variable  $A$ .

Hago  $I \leftarrow I@$  y voy a la Etapa 2

Etapa 4

Corro el procedimiento  $\mathbb{P}_\%$  con la entrada  $(A, x_1, x_2, \alpha_1, \alpha_2, I)$  y guardar la salida en la variable  $A$ .

Hago  $I \leftarrow I\%$  y voy a la Etapa 2

Etapa 5

Corro el procedimiento  $\mathbb{P}_!$  con la entrada  $(A, x_1, x_2, \alpha_1, \alpha_2, I)$  y guardar la salida en la variable  $A$ .

Hago  $I \leftarrow I!$  y voy a la Etapa 2

y por ello  $R(f, \mathcal{G})$  es  $\Sigma$ -efectivamente computable. ■

**2. Lema** (Lema de cuantificación acotada) . Sea  $\Sigma$  un alfabeto finito. Sea  $P : S \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \omega$  un predicado  $\Sigma$ -p.r., con  $S, S_1, \dots, S_n \subseteq \omega$  y  $L_1, \dots, L_m \subseteq \Sigma^*$  no vacíos. Supongamos  $\bar{S} \subseteq S$  es  $\Sigma$ -p.r. Entonces  $\lambda x \vec{x} \vec{\alpha} \left[ \left( \forall t \in \bar{S} \right)_{(t \leq x)} P(t, \vec{x}, \vec{\alpha}) \right]$  es  $\Sigma$ -p.r.  $\hookrightarrow$

*Demostración:*

Sea

$$\bar{P} = P \upharpoonright_{\bar{S} \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m} \cup C_1^{1+n,m} \upharpoonright_{(\omega - \bar{S}) \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m}$$

veamos que es  $\Sigma$ -p.r.

Como  $\bar{S} \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \cap (\omega - \bar{S}) \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m = \emptyset$ ,  $P$  y  $C_1^{1+n,m}$  son  $\Sigma$ -p.r., por el Lema (División por Casos para funciones  $\Sigma$ -p.r.) y el Lema (Restricción de Dominios  $\Sigma$ -p.r.), alcanza con ver que los siguientes conjuntos son  $\Sigma$ -p.r.

$$\bar{S} \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \quad \text{y} \quad (\omega - \bar{S}) \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m$$

Por la Proposición (Caracterización de Conjuntos  $\Sigma$ -p.r.) sabemos que  $D_P = S \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m$  es  $\Sigma$ -p.r. , por lo tanto, por el Lema (Caracterización de Conjuntos Rectangulares  $\Sigma$ -p.r.) ,  $S_1, \dots, S_n, L_1, \dots, L_m$  son  $\Sigma$ -p.r. y además como  $\bar{S}$  es  $\Sigma$ -p.r., por el Lema (o.p de Conjuntos  $\Sigma$ -p.r.) ,  $(\omega - \bar{S})$  también. Entonces nuevamente por el Lema (Caracterización de Conjuntos Rectangulares  $\Sigma$ -p.r.) , ambos conjuntos son  $\Sigma$ -p.r. Así  $\bar{P}$  es  $\Sigma$ -p.r. Notar que  $D_{\bar{P}} = \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m$ . Además, como

$$\begin{aligned} \lambda x \vec{x} \vec{\alpha} \left[ \left( \forall t \in \bar{S} \right)_{(t \leq x)} P(t, \vec{x}, \vec{\alpha}) \right] &= \lambda x \vec{x} \vec{\alpha} \left[ \prod_{t=0}^{t=x} \bar{P}(t, \vec{x}, \vec{\alpha}) \right] \\ &= \lambda x y \vec{x} \vec{\alpha} \left[ \prod_{t=x}^{t=y} \bar{P}(t, \vec{x}, \vec{\alpha}) \right] \circ [C_0^{1+n,m}, p_1^{1+n,m}, \dots, p_{1+n+m}^{1+n,m}] \end{aligned}$$

el Lema de la Sumatoria implica que  $\lambda x \vec{x} \vec{\alpha} \left[ \left( \forall t \in \bar{S} \right)_{(t \leq x)} P(t, \vec{x}, \vec{\alpha}) \right]$  es  $\Sigma$ -p.r.  $\blacksquare$

## Combo 6

### 1. Lema ( $\Sigma$ -efectivamente computable implica $\Sigma$ -efectivamente enumerable).

Si  $S \subseteq w^n \times \Sigma^{*m}$  es  $\Sigma$ -efectivamente computable entonces  $S$  es  $\Sigma$ -efectivamente enumerable.  $\hookrightarrow$

*Demostración:*

Si  $S = \emptyset$ , por definición es  $\Sigma$ -efectivamente enumerable.

Supongamos entonces que  $S \neq \emptyset$  y fijamos  $(\vec{z}, \vec{\gamma}) \in S$ . Sea  $\mathbb{P}_S$  el procedimiento efectivo que compute a  $\chi_S^{w^n \times \Sigma^{*m}}$ . Sea  $\mathbb{P}_1$  un procedimiento efectivo que enumere a  $w^n \times \Sigma^{*m}$ , usando las bajadas y  $*^{\leq}$ , que ya sabemos que son  $\Sigma$ -efectivamente computable. Entonces  $\mathbb{P}_1$  sería

Sea  $x \in \omega$

Etapla 1

Si  $x = 0$ , entonces detenerse y dar como dato de salida  $(0, 0, 0, \dots, \varepsilon, \varepsilon, \varepsilon, \dots)$

Etapla 2

Detenerse y dar como dato de salida  $((x)_1, \dots, (x)_n, *^{\leq}((x)_{n+1}), \dots, *^{\leq}((x)_{(n+m)}))$

Entonces el siguiente procedimiento efectivo  $\mathbb{P}$  enumera a  $S$

Sea  $x \in \omega$

Etapla 1

Realizar  $\mathbb{P}_1$  con la entrada  $x$  para obtener como salida a un  $(\vec{x}, \vec{\alpha}) \in w^n \times \Sigma^{*m}$ .

Etapla 2

Realizar  $\mathbb{P}_S$  con la entrada  $(\vec{x}, \vec{\alpha})$  para obtener como salida un booleano  $e$ .

Etapla 3

Si  $e = 1$ , entonces detenerse y dar como dato de salida  $(\vec{x}, \vec{\alpha})$ .

Si  $e = 0$ , entonces detenerse y dar como dato de salida  $(\vec{z}, \vec{\gamma})$ .

y por ello  $S$  es  $\Sigma$ -efectivamente enumerable. ■

## 2. Teorema (Caracterización de conjuntos $\Sigma$ -recursivamente enumerable).

Dado  $S \subseteq w^n \times \Sigma^{*m}$ . Son equivalentes

(1)  $S$  es  $\Sigma$ -recursivamente enumerable.

(2)  $S = I_F$ , para alguna  $F : D_F \subseteq \omega^k \times \Sigma^{*l} \rightarrow w^n \times \Sigma^{*m}$  tal que cada  $F_{(i)}$  es  $\Sigma$ -recursiva.

(3)  $S = D_f$ , para alguna función  $\Sigma$ -recursiva  $f$ .

Nota: haga solo la prueba (2)  $\implies$  (3), caso  $k = l = 1$  y  $n = m = 2$

$\hookrightarrow$

*Demostración:*

(2)  $\implies$  (3)

Haremos el caso  $k = l = 1$  y  $n = m = 2$ , osea que  $S \subseteq \omega^2 \times \Sigma^{*2}$  y  $F : D_F \subseteq \omega \times \Sigma^* \rightarrow \omega^2 \times \Sigma^{*2}$  es tal que  $I_F = S$  y  $F_{(1)}, F_{(2)}, F_{(3)}$  y  $F_{(4)}$  son  $\Sigma$ -recursivas.

Gracias al Teorema (Neumann vence a Gödel), para cada  $i \in \{1, 2, 3, 4\}$ , las funciones  $F_{(i)}$  son  $\Sigma$ -computable, entonces por definición existen los programas  $\mathcal{P}_i$  que las computan. Sea  $\leq$  un orden total sobre  $\Sigma$ . Definimos

$$H_i = \lambda t x_1 \alpha_1 [\neg \text{Halt}^{1,1}(t, x_1, \alpha_1, \mathcal{P}_i)]$$

(te dice si el programa  $\mathcal{P}_i$  no se detiene partiendo de  $(x_1, \alpha_1)$  en  $t$  pasos)

Notar que  $D_{H_i} = \omega \times \omega \times \Sigma^*$  y  $H_i$  es  $\Sigma$ -mixta. Además sabemos que  $\text{Halt}^{1,1}$  es  $(\Sigma \cup \Sigma_p)$ -p.r, por lo tanto resulta fácil que  $H_i$  es  $(\Sigma \cup \Sigma_p)$ -p.r. Entonces por el Teorema (Independencia del Alfabeto),  $H_i$  es  $\Sigma$ -p.r., lo cual por el Segundo Manantial existen los macros

$$[\text{IF } \neg H_i(V2, V1, W1) \text{ GOTO } A1]$$

pero para usarlas de forma más intuitiva, las escribimos como

$$[\text{IF } \neg \text{Halt}^{1,1}(V2, V1, W1) \text{ GOTO } A1]$$

Luego para  $i = 1, 2$  definimos

$$E_i = \lambda x t x_1 \alpha_1 [x \neq E_{\#1}^{1,1}(t, x_1, \alpha_1, \mathcal{P}_i)]$$

(te dice si el programa  $\mathcal{P}_i$  en  $t$  pasos devuelve  $x$ )

Y para  $i = 3, 4$ , definimos

$$E_i = \lambda t x_1 \alpha_1 [\alpha \neq E_{*1}^{1,1}(t, x_1, \alpha_1, \mathcal{P}_i)]$$

(te dice si el programa  $\mathcal{P}_i$  en  $t$  pasos devuelve  $\alpha$ )

Notar que los predicados  $E_i$  son  $\Sigma$ -mixtos. Además sabemos que  $E_{\#1}^{1,1}$  y  $E_{*1}^{1,1}$  son  $(\Sigma \cup \Sigma_p)$ -p.r, por lo tanto resulta fácil que los  $E_i$  son  $(\Sigma \cup \Sigma_p)$ -p.r. Entonces por el Teorema (Independencia del Alfabeto), cada  $E_i$  es  $\Sigma$ -p.r., lo cual por el Segundo Manantial existen los macros

$$[\text{IF } E_i(V2, V3, V1, W1) \text{ GOTO } A1] \quad [\text{IF } E_i(V2, V1, W1, W2) \text{ GOTO } A1]$$

(para  $i = 1, 2$ ) (para  $i = 3, 4$ )

pero para usarlas de forma más intuitiva, las escribimos como

$$[\text{IF } V2 \neq E_{\#1}^{1,1}(V3, V1, W1, \mathcal{P}_i) \text{ GOTO } A1] \quad [\text{IF } W2 \neq E_{*1}^{1,1}(V2, V1, W1, \mathcal{P}_i) \text{ GOTO } A1]$$

(para  $i = 1, 2$ ) (para  $i = 3, 4$ )

Ahora ya que las funciones  $f_1 = \lambda x [(x)_1]$ ,  $f_2 = \lambda x [(x)_2]$  y  $f_3 = \circ \lambda x [*^\leq ((x)_3)]$  son  $\Sigma$ -p.r., por el Segundo Manantial existen los macros

$$[V2 \leftarrow f_1(V1)] \quad [V2 \leftarrow f_2(V1)] \quad [P1 \leftarrow f_3(V1)]$$

pero para usarlas de forma más intuitiva, las escribimos como

$$[V2 \leftarrow (V1)_1] \quad [V2 \leftarrow (V1)_2] \quad [P1 \leftarrow *^{\leq} (V1)_3]$$

Ahora sí, sea  $\mathcal{P}$  el siguiente programa de  $S^\Sigma$

```

L1  N20 ← N20 + 1
    [ N10 ← (N20)1 ]
    [ N3 ← (N20)2 ]
    [ P3 ← *≤ (N20)3 ]
    [ IF ¬Halt1,1(N10, N3, P3,  $\mathcal{P}_1$ ) GOTO L1 ]
    [ IF ¬Halt1,1(N10, N3, P3,  $\mathcal{P}_2$ ) GOTO L1 ]
    [ IF ¬Halt1,1(N10, N3, P3,  $\mathcal{P}_3$ ) GOTO L1 ]
    [ IF ¬Halt1,1(N10, N3, P3,  $\mathcal{P}_4$ ) GOTO L1 ]
    [ IF N1 ≠ E1,1#1(N10, N3, P3,  $\mathcal{P}_1$ ) GOTO L1 ]
    [ IF N2 ≠ E1,1#1(N10, N3, P3,  $\mathcal{P}_2$ ) GOTO L1 ]
    [ IF P1 ≠ E1,1*1(N10, N3, P3,  $\mathcal{P}_3$ ) GOTO L1 ]
    [ IF P2 ≠ E1,1*1(N10, N3, P3,  $\mathcal{P}_4$ ) GOTO L1 ]

```

es fácil entender el programa si lo ves por partes y teniendo en cuenta que toma como entrada  $(x_1, x_2, \alpha_1, \alpha_2)$

- La línea 2 genera un candidato  $t$  a cantidad de pasos.
- Las líneas 3 y 4 generan un candidato  $(y_1, \gamma_2)$  para la entrada de los  $F_{(i)}$ .
- Las líneas del 5 al 8 verifican si  $F_{(1)}, F_{(2)}, F_{(3)}$  y  $F_{(4)}$  se detienen en  $t$  pasos, con la entrada  $(y_1, \gamma_2)$ .
- Las líneas del 9 al 12 verifican si  $F_{(1)}, F_{(2)}, F_{(3)}$  y  $F_{(4)}$  devuelven  $x_1, x_2, \alpha_1$  y  $\alpha_2$  respectivamente.
- Si alguna verificación **no** es cierta, se vuelve a la línea 1 y repite el proceso con nuevos candidatos.

Finalmente, como  $F = [F_{(1)}, F_{(2)}, F_{(3)}, F_{(4)}]$  y  $I_F = S$ ,  $\mathcal{P}$  se detiene sólo cuando  $(x_1, x_2, \alpha_1, \alpha_2) \in S$ .

Sabiendo esto, es fácil ver que computa la función  $p_1^{2,2}|_S$ . Entonces, listo porque  $p_1^{2,2}|_S$  es  $\Sigma$ -computable, por lo cual es  $\Sigma$ -recursiva por el *Teorema (Gödel vence a Neumann)*, y trivialmente  $\text{Dom}(p_1^{2,2}|_S) = S$ . ■

## Combo 7

### 1. Lema (Lema de minimización acotada).

Sean  $n, m \geq 0$ . Sea  $P : D_P \subseteq \omega \times w^n \times \Sigma^{*m} \rightarrow \omega$  un predicado  $\Sigma$ -p.r. Entonces

(a)  $M(P)$  es  $\Sigma$ -recursiva.

(b) Si hay una función  $\Sigma$ -p.r.  $f : w^n \times \Sigma^{*m}$  tal que

$$M(P)(\vec{x}, \vec{\alpha}) = \min_t P(t, \vec{x}, \vec{\alpha}) \leq f(\vec{x}, \vec{\alpha}), \text{ para cada } (\vec{x}, \vec{\alpha}) \in D_{M(P)}$$

entonces  $M(P)$  es  $\Sigma$ -p.r.

↳

*Demostración:*

(a) (Idea básica, hacer que  $P$  sea  $\Sigma$ -total y que siga siendo  $\Sigma$ -p.r.)

Definimos el siguiente predicado, que es  $\Sigma$ -total y pone ceros donde  $P$  no estaba definida

$$\overline{P} = P \cup C_0^{1+n,m}|_{(\omega \times w^n \times \Sigma^{*m}) - D_P}$$

Dado que  $P$  es  $\Sigma$ -p.r., por la *Proposición (Caracterización de Conjuntos  $\Sigma$ -p.r.)*,  $D_P$  también.

Entonces por el *Lema (o.p de Conjuntos  $\Sigma$ -p.r.)* tengo que  $(\omega \times w^n \times \Sigma^{*m}) - D_P$  es  $\Sigma$ -p.r. y por lo tanto, por

el *Lema (Restricción de Dominios  $\Sigma$ -p.r.)*,  $C_0^{1+n,m}|_{(\omega \times w^n \times \Sigma^{*m}) - D_P}$  también. Como trivialmente

$D_P \cap ((\omega \times w^n \times \Sigma^{*m}) - D_P) = \emptyset$  por el *Lema (División por Casos para funciones  $\Sigma$ -p.r.)*  $\overline{P}$  es  $\Sigma$ -p.r.

Ahora es fácil ver que  $M(P) = M(\overline{P})$  ya que la minimización está definida cuando el predicado es 1. Osea

$$\{t \in \omega : P(t, \vec{x}, \vec{\alpha}) = 1\} = \{t \in \omega : \overline{P}(t, \vec{x}, \vec{\alpha}) = 1\}$$

Esto claramente dice que  $D_{M(P)} = D_{M(\overline{P})}$  y que  $M(P)(\vec{x}, \vec{\alpha}) = M(\overline{P})(\vec{x}, \vec{\alpha})$ , para cada  $(\vec{x}, \vec{\alpha}) \in D_{M(P)}$ , por lo cual  $M(P) = M(\overline{P})$ .

Ahora sí con ver que  $M(\overline{P})$  es  $\Sigma$ -recursiva, alcanza. Pero esto es fácil, porque si tomamos  $k$  tal que

$\overline{P} \in \text{PR}_k^\Sigma \subseteq R_k^\Sigma$  y como  $\overline{P}$  es  $\Sigma$ -total, tenemos que  $M(\overline{P}) \in R_{k+1}^\Sigma$  y por lo tanto  $M(\overline{P}) \in R^\Sigma$ .

(b)

Ya que  $M(P) = M(\overline{P})$ , basta probar que  $M(\overline{P})$  es  $\Sigma$ -p.r. (va a ser necesaria la “cota” que nos da  $f$ )

Primero veremos que  $D_{M(\overline{P})}$  es un conjunto  $\Sigma$ -p.r. Para ello notar que

$$\chi_{D_{M(\overline{P})}}^{w^n \times \Sigma^{*m}} = \lambda \vec{x} \vec{\alpha} [(\exists t \in \omega)_{t \leq f(\vec{x}, \vec{\alpha})} \overline{P}(t, \vec{x}, \vec{\alpha})]$$

lo cual nos dice que

$$\chi_{D_{M(\overline{P})}}^{w^n \times \Sigma^{*m}} = \lambda \vec{x} \vec{\alpha} [(\exists t \in \omega)_{t \leq x} \overline{P}(t, \vec{x}, \vec{\alpha})] \circ [f, p_1^{n,m}, \dots, p_{n+m}^{n,m}]$$

Pero dado que  $\overline{P}$  es  $\Sigma$ -p.r., por el *Lema de cuantificación acotada*, nos dice que  $\lambda \vec{x} \vec{\alpha} [(\exists t \in \omega)_{t \leq x} \overline{P}(t, \vec{x}, \vec{\alpha})]$  es  $\Sigma$ -p.r. y como  $f$  es  $\Sigma$ -p.r. tengo que  $\chi_{D_{M(\overline{P})}}^{w^n \times \Sigma^{*m}}$  también. Ahora definamos un predicado que será muy útil

$$P_1 = \lambda \vec{x} \vec{\alpha} [\overline{P}(t, \vec{x}, \vec{\alpha}) \wedge (\forall j \in \omega)_{j \leq t} j = t \vee \neg \overline{P}(j, \vec{x}, \vec{\alpha})]$$

(Te dice si para los  $< a t$  no se cumple  $\overline{P}$  y para  $t$  si se cumple)

Veamos además que es  $\Sigma$ -p.r. Si defino  $Q = \lambda j \vec{x} \vec{\alpha} [j = t \vee \neg \overline{P}(j, \vec{x}, \vec{\alpha})]$  que claramente es  $\Sigma$ -p.r. por el *Lema (o.p de Predicados  $\Sigma$ -p.r.)*. Por el *Lema de cuantificación acotada*, tengo que  $\lambda \vec{x} \vec{\alpha} [(\forall j \in \omega)_{j \leq t} Q(j, t, \vec{x}, \vec{\alpha})]$  es  $\Sigma$ -p.r. Pero notar que  $P_1 = \lambda \vec{x} \vec{\alpha} [\overline{P}(t, \vec{x}, \vec{\alpha}) \wedge (\forall j \in \omega)_{j \leq t} Q(j, t, \vec{x}, \vec{\alpha})]$  entonces nuevamente por el *Lema (o.p de Predicados  $\Sigma$ -p.r.)*,  $P_1$  es  $\Sigma$ -p.r.

Notar además que  $P_1$  es  $\Sigma$ -total y

$$P_1(t, \vec{x}, \vec{\alpha}) = 1 \quad \text{si y solo si} \quad (\vec{x}, \vec{\alpha}) \in D_{M(\overline{P})} \text{ y } t = M(\overline{P})(\vec{x}, \vec{\alpha})$$

Esto nos dice que

$$M(\overline{P}) = \left( \lambda \vec{x} \vec{\alpha} \left[ \prod_{t=0}^{f(\vec{x}, \vec{\alpha})} t^{P_1(t, \vec{x}, \vec{\alpha})} \right] \right) \upharpoonright_{D_{M(\overline{P})}}$$

(como  $t^0 = 1$ ,  $t^1 = t$  y un solo  $t$  va a cumplir  $P_1$  entonces queda  $1 \times \dots \times 1 \times t \times 1 \dots = t$ )

por lo cual para probar que  $M(\overline{P})$  es  $\Sigma$ -p.r., basta probar que

$$F = \lambda \vec{x} \vec{\alpha} \left[ \prod_{t=0}^{f(\vec{x}, \vec{\alpha})} t^{P_1(t, \vec{x}, \vec{\alpha})} \right]$$

lo es, pero

$$F = \lambda x y \vec{x} \vec{\alpha} \left[ \prod_{t=x}^y t^{P_1(t, \vec{x}, \vec{\alpha})} \right] \circ [C_0^{n,m}, f, p_1^{n,m}, \dots, p_{n+m}^{n,m}]$$

y por lo tanto el Lema de la Sumatoria nos dice que  $F$  es  $\Sigma$ -p.r., por lo cual  $M(\overline{P}) = M(P)$  es  $\Sigma$ -p.r. ■

*Lema (Lema de cuantificación acotada):*

Sea  $\Sigma$  un alfabeto finito. Sea  $P : S \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \omega$  un predicado  $\Sigma$ -p.r., con  $S, S_1, \dots, S_n \subseteq \omega$  y  $L_1, \dots, L_m \subseteq \Sigma^*$  no vacíos. Supongamos  $\overline{S} \subseteq S$  es  $\Sigma$ -p.r.

Entonces  $\lambda x \vec{x} \vec{\alpha} \left[ (\exists t \in \overline{S})_{(t \leq x)} P(t, \vec{x}, \vec{\alpha}) \right]$  es  $\Sigma$ -p.r. (Cambiar  $\forall$  por  $\exists$  y es el [Combo 5.2](#) o el Lema 23 de la guía 5)

**2. Lema .**

Supongamos  $f : D_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$  es  $\Sigma$ -recursiva,  $O = \{\omega, \Sigma^*\}$  y  $S \subseteq D_f$  es  $\Sigma$ -recursivamente enumerable entonces  $f|_S$  es  $\Sigma$ -recursiva.

Nota: haga solo el caso  $S$  no vacío,  $n = m = 1$  y  $O = \Sigma^*$

↳

*Demostración:*

Haremos el caso  $n = m = 1$  y  $O = \Sigma^*$ , osea que  $f : D_f \subseteq \omega \times \Sigma^* \rightarrow \Sigma^*$ .

Como  $S$  es  $\Sigma$ -recursivamente enumerable tenemos que hay una función  $F : \omega \rightarrow \omega \times \Sigma^*$  tal que  $I_F = S$  y  $F_{(1)}$  y  $F_{(2)}$  son  $\Sigma$ -recursivas. Además  $f$  también, entonces por el Segundo Manantial, existen las macros

$$[W2 \leftarrow f(V1, W1)] \quad [V2 \leftarrow F_{(1)}(V1)] \quad [W2 \leftarrow F_{(2)}(V1)]$$

Y como  $D = \lambda xy[x \neq y]$  y  $D' = \lambda \alpha \beta[\alpha \neq \beta]$  son  $\Sigma$ -p.r. Por el Segundo Manantial, existen las macros

$$[IF D(V1, V2) GOTO A1] \quad [IF D'(W1, W2) GOTO A1]$$

pero para usarlas de forma más intuitiva, las escribimos como

$$[IF V1 \neq V2 GOTO A1] \quad [IF W1 \neq W2 GOTO A1]$$

Ahora sí, sea  $\mathcal{P}$  el siguiente programa de  $S^\Sigma$  (donde  $N10 = 0$  al iniciar)

```

L1  [ N2 ← F(1)(N10) ]
    [ P2 ← F(2)(N10) ]
    [ IF N1 ≠ N2 GOTO L2 ]
    [ IF P1 ≠ P2 GOTO L2 ]
    [ P1 ← f(N1, P1) ]
    GOTO L3
L2  N10 ← N10 + 1
    GOTO L1
L3  SKIP

```

el cual es fácil ver que  $\mathcal{P}$  computa  $f|_S$ . Ya que, si analizamos por líneas

- Las **líneas 1** y **2** generan un elemento  $(y, \beta) \in S$ .
- Las líneas **3** y **4** comparan el input  $(x, \alpha)$  con  $(y, \beta)$ .
  - Si son iguales, esto implicaría que  $(x, \alpha) \in S$ . Por eso calculamos  $f(x, \alpha)$  y lo retornamos.
  - Si no son iguales, se incrementa la variable para generar y se vuelve a la **línea 1** para generar un nuevo  $(y, \beta) \in S$ .

Notar que si  $(x, \alpha) \notin S$ , entonces  $\mathcal{P}$  no se detiene porque la comparación **nunca** va a dar igual. Entonces así  $\mathcal{P}$  computa  $f|_S$ , por lo tanto es  $\Sigma$ -computable y por el Teorema (Gödel vence a Neumann), es  $\Sigma$ -recursiva. ■

## Combo 8

**1. Lema.** Supongamos que  $\Sigma \supseteq \Sigma_p$ . Entonces  $\text{AutoHalt}^\Sigma$  no es  $\Sigma$ -recursivo.  $\hookrightarrow$

*Demostración:*

Lo vamos a demostrar por el absurdo, entonces supongamos que  $\text{AutoHalt}^\Sigma$  **sí es  $\Sigma$ -recursivo**.

Por el Segundo Manantial tenemos que hay un macro

$$[ \text{IF } \text{AutoHalt}^\Sigma(W1) \text{ GOTO } A1 ]$$

Sea  $\mathcal{P}_0$  el siguiente programa de  $S^\Sigma$

$$L1 \quad [ \text{IF } \text{AutoHalt}^\Sigma(P1) \text{ GOTO } L1 ]$$

Notar que por definición de  $\text{AutoHalt}^\Sigma$  sabemos que para cada  $\mathcal{P} \in S^\Sigma$  se cumple que

$$\text{AutoHalt}^\Sigma(\mathcal{P}) = 1 \text{ sii } \mathcal{P} \text{ se detiene partiendo del estado } \|\mathcal{P}\|.$$

pero por otra parte

$$\text{AutoHalt}^\Sigma(\mathcal{P}_0) = 0 \text{ sii } \mathcal{P}_0 \text{ se detiene partiendo del estado } \|\mathcal{P}_0\|$$

Estas dos afirmaciones se contradicen y el absurdo viene de que supusimos que  $\text{AutoHalt}^\Sigma$  sí es  $\Sigma$ -recursivo. ■

Otra forma de decir lo mismo, es que si corremos  $\mathcal{P}_0$  partiendo de  $\|\mathcal{P}_0\|$ , tenemos dos posibilidades

- $\text{AutoHalt}^\Sigma(\mathcal{P}_0) = 0$ , osea que se sale del IF y  $\mathcal{P}_0$  **termina**, por lo tanto  $\text{AutoHalt}^\Sigma(\mathcal{P}_0) = 1$ . Absurdo.
- $\text{AutoHalt}^\Sigma(\mathcal{P}_0) = 1$ , osea entra en bucle y  $\mathcal{P}_0$  **no termina**, por lo tanto  $\text{AutoHalt}^\Sigma(\mathcal{P}_0) = 0$ . Absurdo.

**2. Teorema.** Supongamos que  $\Sigma \supseteq \Sigma_p$ . Entonces  $\text{AutoHalt}^\Sigma$  no es  $\Sigma$ -efectivamente computable. Es decir, no hay ningún procedimiento efectivo que decida si un programa de  $S^\Sigma$  termina partiendo de sí mismo.  $\hookrightarrow$

*Demostración:*

Si  $\text{AutoHalt}^\Sigma$  fuera  $\Sigma$ -efectivamente computable, la *Tesis de Church* nos diría que es  $\Sigma$ -recursivo, contradiciendo el Lema anterior. *Tesis de Church* : “Toda función  $\Sigma$ -efectivamente computable es  $\Sigma$ -recursiva.” ■

**3. Lema.** Supongamos que  $\Sigma \supseteq \Sigma_p$ .

Entonces  $A = \{\mathcal{P} \in \text{Pro}^\Sigma : \text{AutoHalt}^\Sigma(\mathcal{P}) = 1\}$  es  $\Sigma$ -recursivamente enumerable y no  $\Sigma$ -recursivo.

Más aún  $N = \{\mathcal{P} \in \text{Pro}^\Sigma : \text{AutoHalt}^\Sigma(\mathcal{P}) = 0\}$  no es  $\Sigma$ -recursivamente enumerable.  $\hookrightarrow$

*Demostración:*

Sea  $P = \lambda t \mathcal{P} [\text{Halt}^{0,1}(t, \mathcal{P}, \mathcal{P})]^{12}$ . Notar que  $P$  es  $\Sigma$ -p.r. (porque  $\Sigma \supseteq \Sigma_p$ ) y por lo tanto  $M(P)$  es  $\Sigma$ -recursiva. Además, recordar que se definió  $\text{AutoHalt}^\Sigma = \lambda \mathcal{P} [(\exists t \in \omega) \text{Halt}^{0,1}(t, \mathcal{P}, \mathcal{P}) = 1]$ , entonces

$$D_{M(P)} = \{\mathcal{P} \in \text{Pro}^\Sigma : (\exists t \in \omega) P(t, \mathcal{P}) = 1\} = \{\mathcal{P} \in \text{Pro}^\Sigma : \text{AutoHalt}^\Sigma(\mathcal{P}) = 1\} = A$$

Pero por la *Caracterización de conjuntos  $\Sigma$ -r.e.* (dada en el Combo 6.2) que entre otras cosas dice: un conjunto es  $\Sigma$ -r.e. sii es el dominio de alguna función  $\Sigma$ -r.e.. Entonces como  $D_{M(P)} = A$  tenemos que  **$A$  sí es  $\Sigma$ -r.e.** Supongamos ahora que  $N$  es  $\Sigma$ -r.e.. Entonces por el *Lema de restricción de dominios de funciones  $\Sigma$ -r.* (dado en el Combo 7.2) la función  $C_0^{0,1} \upharpoonright_N$  es  $\Sigma$ -recursiva ya que  $C_0^{0,1}$  lo es. Análogamente como  $A$  es  $\Sigma$ -r.e., también lo es  $C_1^{0,1} \upharpoonright_A$ .

<sup>12</sup>Recordar que  $\text{Halt}^{n,m} = \lambda t \vec{x} \vec{\alpha} \mathcal{P} [i^{n,m}(t, \vec{x}, \vec{\alpha}, \mathcal{P}) = n(\mathcal{P}) + 1]$ , osea te dice si  $\mathcal{P}$  con entrada  $\|\vec{x}, \vec{\alpha}\|$  luego de  $t$  pasos terminó.

Ahora sí, ya que

$$\text{AutoHalt}^\Sigma = C_1^{0,1} \mid_A \cup C_0^{0,1} \mid_N, \quad A \cup N = \text{Pro}^\Sigma \quad y \quad A \cap N = \emptyset$$

por el Lema (División por Casos para funciones  $\Sigma$ -r.) tenemos que  $\text{AutoHalt}^\Sigma$  es  $\Sigma$ -recursiva, lo cual contradice el Lema anterior. Esto prueba que  $N$  **no es  $\Sigma$ -r.e.**

Finalmente, supongamos que  $A$  es  $\Sigma$ -recursivo. Entonces el conjunto  $N = (\Sigma^* - A) \cap \text{Pro}^\Sigma$  debería serlo, lo cual es absurdo por lo visto anteriormente. Por lo tanto  $A$  **no es  $\Sigma$ -recursivo.** ■

**4. Teorema** (Neumann vence a Gödel) . Si una función  $h$  es  $\Sigma$ -recursiva, entonces  $h$  es  $\Sigma$ -computable.

*Nota:* en la inducción, hacer solo el caso  $h = M(P)$

↳

*Demostración:*

Esto será probado por inducción en  $k$ , que si  $h \in R_k^\Sigma$ , entonces  $h$  es  $\Sigma$ -computable .

El caso  $k = 0$  es fácil ya que  $R_0 = \text{PR}_0$ , entonces hay que hacer programas que computen

$\{\text{Suc}, \text{Pred}, C_0^{0,0}, C_\varepsilon^{0,0}\} \cup \{d_a : a \in \Sigma\} \cup \{p_j^{n,m} : 1 \leq j \leq n+m\}$ , los cuales son todos triviales

Suc	Pred	$C_0^{0,0}$	$C_\varepsilon^{0,0}$	$d_a$	$p_j^{n,m}$
$N1 \leftarrow N1 + 1$	IF $N1 \neq 0$ GOTO L2 L1 GOTO L1 L2 $N1 \leftarrow N1 - 1$	$N1 \leftarrow 0$	$P1 \leftarrow \varepsilon$	$P1 \leftarrow P1.a$	$N1 \leftarrow N\bar{j}$ $o$ $P1 \leftarrow P\bar{j}$

Supongamos que la propiedad se cumple para un  $k$  fijo y veamos que se cumple también para  $h \in R_{k+1}^\Sigma$ .

Hay varios casos. Veamos el caso donde  $h = M(P)$ , con  $P : \omega \times \omega^n \times \Sigma^{*m} \rightarrow \omega$ , un predicado en  $R_k^\Sigma$ .

Por hipótesis inductiva  $P$  es  $\Sigma$ -computable, osea que por el Primer Manantial existe el macro

$$[ \text{IF } P(V1, \dots, V\overline{n+1}, W1, \dots, W\overline{m}) \text{ GOTO } A1 ]$$

El cual nos permite realizar el siguiente programa  $\mathcal{P}$  de  $S^\Sigma$

$$\begin{aligned} \text{L1} \quad & [ \text{IF } P(\overline{Nn+1}, N1, \dots, N\overline{n}, P1, \dots, P\overline{m}) \text{ GOTO } \text{L2} ] \\ & \overline{Nn+1} \leftarrow \overline{Nn+1} + 1 \\ & \text{GOTO } \text{L1} \\ \text{L2} \quad & N1 \leftarrow \overline{Nn+1} \end{aligned}$$

que es fácil ver que computa  $M(P)$ . Supongamos que  $\mathcal{P}$  inicia de un estado  $\|\vec{x}, \vec{\alpha}\|$ , entonces hay dos casos

- Si  $(\vec{x}, \vec{\alpha}) \notin D_{M(P)}$ , entonces el predicado  $P$  nunca va a ser 1, incluso al incrementar  $\overline{Nn+1}$ , por lo tanto  $\mathcal{P}$  nunca se va a detener.
- Si  $(\vec{x}, \vec{\alpha}) \in D_{M(P)}$ , entonces eventualmente  $P$  va a ser 1. Como además  $\overline{Nn+1}$  inicia en 0 e incrementa de a 1, cuando  $P$  valga 1 entonces va a retornar el mínimo valor, osea  $M(P)(\vec{x}, \vec{\alpha})$ .

Por esto  $\mathcal{P}$  computa  $M(P)$  y por lo tanto  $M(P)$  **es  $\Sigma$ -computable.** ■

## Combo 9

### 1. Lema (Lema división por casos para funciones $\Sigma$ -recursivas).

Supongamos  $f_i : D_{f_i} \subseteq w^n \times \Sigma^{*m} \rightarrow O, i = 1, \dots, k$  son  $\Sigma$ -recursivas. Tales que  $D_{f_i} \cap D_{f_j} = \emptyset$  para cada  $i \neq j$ . Entonces la función  $f_1 \cup \dots \cup f_k$  es  $\Sigma$ -recursiva.

Nota: haga el caso  $k = 2, n = m = 1$  y  $O = \omega$

↳

*Demostración:*

Haremos el caso  $k = 2, n = m = 1$  y  $O = \omega$ . Osea que tenemos  $f_i : D_{f_i} \subseteq \omega \times \Sigma^* \rightarrow \omega$  con  $i = 1, 2$ .

Por el Teorema (Neumann vence a Gödel) las funciones son  $\Sigma$ -computables, osea que existen los programas  $\mathcal{P}_1$  y  $\mathcal{P}_2$  tales que las computan respectivamente. Entonces para  $i = 1, 2$  definamos

$$H_i = \lambda t x_1 \alpha_1 [\text{Halt}^{1,1}(t, x_1, \alpha_1, \mathcal{P}_i)]$$

(Te dice si en  $t$  pasos el programa  $\mathcal{P}_i$  con entrada  $\|x_1, \alpha_1\|$  termina)

notar que  $D_{H_i} = \omega \times \omega \times \Sigma^*$  y que  $H_i$  es  $\Sigma$ -mixta. Además sabemos que  $\text{Halt}^{1,1}$  es  $(\Sigma \cup \Sigma_p)$ -pr. Entonces por el Teorema (Independencia del Alfabeto),  $H_i$  es  $\Sigma$ -p.r. y por el Segundo Manantial existen las macros

$$[ \text{IF } H_i(V1, V2, W1) \text{ GOTO } A1 ]$$

pero para usarla de forma más intuitiva, la escribimos como

$$[ \text{IF } \text{Halt}^{1,1}(V1, V2, W1, \mathcal{P}_i) \text{ GOTO } A1 ]$$

Luego ya que cada  $f_i$  es  $\Sigma$ -computable, por el Primer Manantial existen las macros

$$[ V2 \leftarrow f_1(V1, W1) ] \quad [ V2 \leftarrow f_2(V1, W1) ]$$

Con todo esto definimos el siguiente programa  $\mathcal{P}$  de  $S^\Sigma$

```

L1  N10 ← N10 + 1
    [ IF Halt1,1(N10, N1, P1,  $\mathcal{P}_1$ ) GOTO L2 ]
    [ IF Halt1,1(N10, N1, P1,  $\mathcal{P}_2$ ) GOTO L3 ]
    GOTO L1
L2  [ N1 ←  $f_1$ (N1, P1) ]
    GOTO L4
L3  [ N1 ←  $f_2$ (N1, P1) ]
L4  SKIP

```

el cual claramente computa  $f_1 \cup f_2$ , ya que si corremos  $\mathcal{P}$  partiendo del estado  $\|x_1, \alpha_1\|$ , tenemos dos casos

- $(x_1, \alpha_1) \in D_{f_1 \cup f_2}$ , entonces en alguna cantidad de pasos se va cumplir alguno de los dos IF (nunca ambos ya que  $D_{f_1} \cap D_{f_2} = \emptyset$ ) y se va a detener retornando  $f_1(x_1, \alpha_1)$  o  $f_2(x_1, \alpha_1)$ , según corresponda.
- $(x_1, \alpha_1) \notin D_{f_1 \cup f_2}$ , entonces nunca se van a cumplir los IF, por lo tanto  $\mathcal{P}$  nunca va a detenerse.

por lo tanto  $f_1 \cup f_2$  es  $\Sigma$ -computable. ■

**2. Teorema** (Gödel vence a Neumann) .

Si  $f : D_f \subseteq w^n \times \Sigma^{*m} \rightarrow \omega$  es  $\Sigma$ -computable, entonces  $f$  es  $\Sigma$ -recursiva.

↳

*Demostración:*

Sea  $\mathcal{P}_0$  un programa que compute a  $f$ . Primero veamos que  $f$  es  $(\Sigma \cup \Sigma_p)$ -recursiva. Notar que<sup>13</sup>

$$f = E_{\#1}^{n,m} \circ [T^{n,m} \circ [p_1^{n,m}, \dots, p_{n+m}^{n,m}, C_{\mathcal{P}_0}^{n,m}], p_1^{n,m}, \dots, p_{n+m}^{n,m}, C_{\mathcal{P}_0}^{n,m}]$$

donde  $p_1^{n,m}, \dots, p_{n+m}^{n,m}$  y  $C_{\mathcal{P}_0}^{n,m}$  son respecto al alfabeto  $\Sigma \cup \Sigma_p$ , es decir, tienen dominio  $\omega^n \times (\Sigma \cup \Sigma_p)^{*m}$ . Esto nos dice que  $f$  es  $(\Sigma \cup \Sigma_p)$ -recursiva. Osea que el *Teorema (Independencia del Alfabeto)* nos dice que  **$f$  es  $\Sigma$ -recursiva.** ■

<sup>13</sup>

$$f = \underbrace{E_{\#1}^{n,m}}_{\text{resultado de N1, importante (\#)}} \circ \left[ \underbrace{T^{n,m} \circ [p_1^{n,m}, \dots, p_{n+m}^{n,m}, C_{\mathcal{P}_0}^{n,m}]}_{\text{cantidad de pasos para que termine}}, \underbrace{p_1^{n,m}, \dots, p_{n+m}^{n,m}}_{\text{input}}, \underbrace{C_{\mathcal{P}_0}^{n,m}}_{\text{programa}} \right]$$

## Resultados Muy Usados en las Demostraciones

Las (★) indican que son un combo y el [ **n** ] es la cantidad de veces que se mencionó el resultado (incluso en una misma demo).

### Lema (Operaciones con Predicados $\Sigma$ -p.r.) [ 4 ]

Si  $P : S \subseteq w^n \times \Sigma^{*m} \rightarrow \omega$  y  $Q : S \subseteq w^n \times \Sigma^{*m} \rightarrow \omega$  son predicados  $\Sigma$ -p.r., entonces  $P \wedge Q, P \vee Q$  y  $\neg P$  también.

### Lema (Operaciones con Conjuntos $\Sigma$ -p.r.) [ 3 ]

Si  $S_1, S_2 \subseteq w^n \times \Sigma^{*m}$  son  $\Sigma$ -p.r., entonces  $S_1 \cup S_2, S_1 \cap S_2$  y  $S_1 - S_2$  son  $\Sigma$ -p.r.

### Lema (Restricción de Dominios $\Sigma$ -p.r.) [ 4 ]

Supongamos  $f : D_f \subseteq w^n \times \Sigma^{*m} \rightarrow \omega$  es  $\Sigma$ -p.r. Si  $S \subseteq D_f$  es  $\Sigma$ -p.r., entonces  $f|_S$  es  $\Sigma$ -p.r.

### Lema (Caracterización de Conjuntos Rectangulares $\Sigma$ -p.r.) [ 3 ]

Supongamos  $S_1, \dots, S_n \subseteq \omega$  y  $L_1, \dots, L_m \subseteq \Sigma^*$  no vacíos. Entonces  $S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m$  es  $\Sigma$ -p.r. sii  $S_1, \dots, S_n, L_1, \dots, L_m$  son  $\Sigma$ -p.r.

### Lema (Lema de la Sumatoria) (Combo 4.2) [ 2 ]

(★)

Sea  $\Sigma$  un alfabeto finito. Si  $f : \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \omega$  es  $\Sigma$ -p.r., con  $S_1, \dots, S_n \subseteq \omega$  y  $L_1, \dots, L_m \subseteq \Sigma^*$  no vacíos. Entonces  $\lambda xy\vec{x}\vec{\alpha} \left[ \prod_{t=x}^{t=y} f(t, \vec{x}, \vec{\alpha}) \right]$  es  $\Sigma$ -p.r.

### Proposición (Caracterización de Conjuntos $\Sigma$ -p.r.) (Combo 1.1) [ 5 ]

(★)

Un conjunto  $S$  es  $\Sigma$ -p.r. sii  $S$  es el dominio de alguna función  $\Sigma$ -p.r.

### Proposición (Primer Manantial de Macros) [ 3 ]

Sea  $\Sigma$  un alfabeto finito. Si

$$f : D_f \subseteq w^n \times \Sigma^{*m} \rightarrow \omega$$

$$g : D_g \subseteq w^n \times \Sigma^{*m} \rightarrow \Sigma^*$$

$$P : D_P \subseteq w^n \times \Sigma^{*m} \rightarrow \{0, 1\}$$

son  **$\Sigma$ -computables**, entonces en  $S^\Sigma$  hay macros

$$[ \overline{Vn+1} \leftarrow f(V1, \dots, V\bar{n}, W1, \dots, W\bar{m}) ]$$

$$[ \overline{Wm+1} \leftarrow f(V1, \dots, V\bar{n}, W1, \dots, W\bar{m}) ]$$

$$[ \text{IF } P(V1, \dots, V\bar{n}, W1, \dots, W\bar{m}) \text{ GOTO } A1 ]$$

### Proposición (Segundo Manantial de Macros) [ 7 ]

Sea  $\Sigma$  un alfabeto finito. Si

$$f : D_f \subseteq w^n \times \Sigma^{*m} \rightarrow \omega$$

$$g : D_g \subseteq w^n \times \Sigma^{*m} \rightarrow \Sigma^*$$

$$P : D_P \subseteq w^n \times \Sigma^{*m} \rightarrow \{0, 1\}$$

son  **$\Sigma$ -recursivas**, entonces en  $S^\Sigma$  hay macros

$$[ \overline{Vn+1} \leftarrow f(V1, \dots, V\bar{n}, W1, \dots, W\bar{m}) ]$$

$$[ \overline{Wm+1} \leftarrow f(V1, \dots, V\bar{n}, W1, \dots, W\bar{m}) ]$$

$$[ \text{IF } P(V1, \dots, V\bar{n}, W1, \dots, W\bar{m}) \text{ GOTO } A1 ]$$

### Teorema (Independencia del Alfabeto) [ 5 ]

Sea  $\Sigma$  y  $\Gamma$  alfabetos cualquiera y  $f$  una función  $\Sigma$ -mixta y  $\Gamma$ -mixta, entonces  $f$  es  $\Sigma$ -recursiva sii  $f$  es  $\Gamma$ -recursiva.

### Teorema (Neumann vence a Gödel) (Combos 1.2 | 8.4) (★)

Si  $h$  es  $\Sigma$ -recursiva, entonces  $h$  es  $\Sigma$ -computable. [ 2 ]

### Teorema (Gödel vence a Neumann) (Combos 3.1 | 9.2) (★)

Si  $h$  es  $\Sigma$ -computable, entonces  $h$  es  $\Sigma$ -recursiva. [ 2 ]

### Lema (Lema de división por casos para funciones $\Sigma$ -p.r.) (Combo 2.1) [ 3 ]

(★)

Sea  $O = \{\omega, \Sigma^*\}$  y  $n, m, k \in \omega$ . Supongamos  $f_i : D_{f_i} \subseteq w^n \times \Sigma^{*m} \rightarrow O, i = 1, \dots, k$  son  **$\Sigma$ -p.r.**

Tales que  $D_{f_i} \cap D_{f_j} = \emptyset$  para cada  $i \neq j$ . Entonces la función  $f_1 \cup \dots \cup f_k$  es  **$\Sigma$ -p.r.**

### Lema (Lema de división por casos para funciones $\Sigma$ -recursivas) (Combo 9.1) [ 1 ]

(★)

Sea  $O = \{\omega, \Sigma^*\}$  y  $n, m, k \in \omega$ . Supongamos  $f_i : D_{f_i} \subseteq w^n \times \Sigma^{*m} \rightarrow O, i = 1, \dots, k$  son  **$\Sigma$ -recursivas**.

Tales que  $D_{f_i} \cap D_{f_j} = \emptyset$  para cada  $i \neq j$ . Entonces la función  $f_1 \cup \dots \cup f_k$  es  **$\Sigma$ -recursiva**.

## Referencias de los Resultados Anteriores (Están acá para no sobrecargar la página anterior)

1. *Lema (Operaciones con Predicados  $\Sigma$ -p.r.)* es el lema 14 de la Guía 5.
2. *Lema (Operaciones con Conjuntos  $\Sigma$ -p.r.)* es el lema 15 de la Guía 5.
3. *Lema (Restricción de Dominios  $\Sigma$ -p.r.)* es el lema 17 de la Guía 5.
4. *Lema (Caracterización de Conjuntos Rectangulares  $\Sigma$ -p.r.)* es el lema 16 de la Guía 5.
5. *Lema (Lema de la Sumatoria)* es el lema 22 de la Guía 5 y casi es el **Combo 4.2**.
6. *Proposición (Caracterización de Conjuntos  $\Sigma$ -p.r.)* es la prop 19 de la Guía 5, del apunte la 4.4 y el **Combo 1.1**.
7. *Proposición (Primer Manantial de Macros)* es la proposición 5 de la Guía 7.
8. *Proposición (Segundo Manantial de Macros)* es la proposición 2 de la Guía 8.
9. *Teorema (Independencia del Alfabeto)* es el teorema 4.2 del apunte.
10. *Teorema (Neumann vence a Gödel)* es el teorema 1 de la Guía 8, el **Combo 1.2** y el **Combo 8.4**.
11. *Teorema (Gödel vence a Neumann)* del apunte es el 4.3 y el **Combo 9.2**.
12. *Lema (Lema de división por casos para funciones  $\Sigma$ -p.r.)* es el lema 4.18 del apunte y el **Combo 2.1**.
13. *Lema (Lema de división por casos para funciones  $\Sigma$ -recursivas)* es el lema 4.56 del apunte y el **Combo 9.1**.