

Lenguajes Formales y Computabilidad | FAMAF - UNC

Combos de definiciones y convenciones notacionales y los Combos de teoremas

01.07.2025

Matias Viola

Contenido

1. Convenciones	1
2. Combos de definiciones y convenciones notacionales	1
2.1. Combo 1: Defina:	1
2.1.1. Cuando un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ es llamado Σ -recursivo	1
2.1.2. $\langle s_1, s_2, \dots \rangle$	1
2.1.3. « f es una función Σ -mixta»	1
2.1.4. «familia Σ -indexada de funciones»	1
2.1.5. $R(f, \varrho)$: Recursion primitiva sobre variable alfabética con valores numéricos..	1
2.2. Combo 2: Defina:	2
2.2.1. $d \vdash^n d'$ y $d \vdash^* d'$	2
2.2.2. $L(M)$	2
2.2.3. « f es una función de tipo (n, m, s) »	2
2.2.4. (x)	2
2.2.5. $(x)_i$	2
2.3. Combo 3: Defina:	2
2.3.1. Cuando un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ es llamado Σ -re	2
2.3.2. s^{\leq}	3
2.3.3. $*^{\leq}$	3
2.3.4. $\#^{\leq}$	3
2.4. Combo 4: Defina cuando una función $f : \text{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow \omega$ es llamada Σ -efectivamente computable y defina «el procedimiento P computa a la función f »	3
2.5. Combo 5: Defina cuando un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ es llamado Σ -efectivamente computable y defina: «el procedimiento efectivo P decide la pertenencia a S »	4
2.6. Combo 6: Defina cuando un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ es llamado Σ -efectivamente enumerable y defina: «el procedimiento efectivo P enumera a S »	4
2.7. Combo 7: Defina cuando una función $f : \text{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow \omega$ es llamada Σ -Turing computable y defina «la máquina de Turing M computa a la función f »	4
2.8. Combo 8: Defina:	4
2.8.1. $M(P)$ Minimización de variable numérica	4
2.8.2. Lt	5
2.8.3. Conjunto rectangular	5
2.8.4. « S es un conjunto de tipo (n, m) »	5
2.9. Combo 9	5
2.9.1. Conjunto rectangular	5
2.9.2. « I es una instrucción de S^Σ »	5
2.9.3. « P es un programa de S^Σ »	5
2.9.4. I_i^P	5
2.9.5. $n(P)$	6

2.9.6.	Bas	6
2.10.	Combo 10: Defina relativo al lenguaje S^Σ :	6
2.10.1.	«estado»	6
2.10.2.	«descripción instantánea»	6
2.10.3.	S_P	6
2.10.4.	«estado obtenido luego de t pasos, partiendo del estado $(\vec{x}, \vec{\alpha})$ »	6
2.10.5.	« P se detiene (luego de t pasos), partiendo desde el estado $(\vec{x}, \vec{\alpha})$ »	7
2.11.	Combo 11: Defina:	7
2.11.1.	$\Psi_P^{n,m,\#}$	7
2.11.2.	« f es Σ -computable» y « P computa a f »	7
2.11.3.	$M^\leq(P)$ Minimización de variable alfabética	7
2.12.	Combo 12: Defina cuando un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ es llamado Σ -computable, cuando es llamado Σ -enumerable y defina «el programa P enumera a S »	7
2.13.	Combo 13	8
2.13.1.	$i^{n,m}$	8
2.13.2.	$E_{\#}^{n,m}$	8
2.13.3.	$E_{\#}^{n,m} + E_{*}^{n,m}$	8
2.13.4.	$E_{\#j}^{n,m}$	8
2.13.5.	$E_{*j}^{n,m}$	8
2.13.6.	$\text{Halt}^{n,m}$	9
2.13.7.	$T^{n,m}$	9
2.13.8.	AutoHalt^Σ	9
2.13.9.	Los conjuntos A y N	9
2.14.	Combo 14: Explique en forma detallada la notación lambda	9
2.15.	Combo 15: Dada una función $f : \text{Dom}_f \subseteq \omega \times \Sigma^* \rightarrow \omega$, describa qué tipo de objeto es y qué propiedades debe tener el macro: $[V2 \leftarrow f(V1, W1)]$	10
2.16.	Combo 16: Dado un predicado $p : \text{Dom}_f \subseteq \omega \times \Sigma^* \rightarrow \omega$, describa qué tipo de objeto es y qué propiedades debe tener el macro: $[\text{IF } P(V1, W1) \text{ GOTO } A1]$	10
2.17.	Combo 17: Defina el concepto de función y desarrolle las tres Convenciones Notacionales asociadas a dicho concepto	11
3.	Combos de teoremas	11
3.1.	Combo 1	11
3.1.1.	Proposición: Caracterización de conjuntos Σ -pr	11
3.1.2.	Teorema: Neumann vence a Gödel	12
3.2.	Combo 2	13
3.2.1.	Lema 4.35: Lema de división por casos para funciones Σ -pr	13
3.2.2.	Proposición: Caracterización básica de conjuntos Σ -enumerables	14
3.3.	Combo 3	15
3.3.1.	Teorema: Gödel vence a Neumann	15
3.3.2.	Teorema: Caracterización de conjuntos Σ -efectivamente computables	15
3.4.	Combo 4	16

3.4.1.	Proposición: Caracterización básica de conjuntos Σ -enumerables	16
3.4.2.	Lema: Lema de la sumatoria	17
3.5.	Combo 5	17
3.5.1.	Lema, Ejercicio de la Guía 5	17
3.5.2.	Lema: Lema de cuantificación acotada	18
3.6.	Combo 6	18
3.6.1.	Lema:	18
3.6.2.	Teorema 4.12: Caracterización de conjuntos Σ -re	19
3.7.	Combo 7	20
3.7.1.	Lema 4.25: Lema de minimización acotada	20
3.7.2.	Lema	21
3.8.	Combo 8	21
3.8.1.	Lema 4.62	21
3.8.2.	Teorema 4.15.	22
3.8.3.	Lema 4.63.	22
3.8.4.	Teorema: Neumann vence a Gödel	22
3.9.	Combo 9	23
3.9.1.	Lema: Lema de división por casos para funciones Σ -r	23
3.9.2.	Teorema 4.5: Gödel vence a Neumann	23
4.	Utilidades	24
4.1.	Lema 14.	24
4.2.	Def Conjuntos Σ -pr	24
4.3.	Lema 4.31.	24
4.4.	Lema 4.32.	24
4.5.	Lema 4.33.	24
4.6.	Lema 4.34.	24
4.7.	Proposición 19.	24
4.8.	Lema 4.35: Lema de division por casos para funciones Σ -pr	24
4.9.	Lema 4.21.	24
4.10.	Proposición 4.8	25
4.11.	Definición de función Σ -computable	25
4.12.	Definición de conjuntos Σ -enumerables	25
4.13.	Teorema 4.2 (Independencia del alfabeto)	25
4.13.1.	Lema 4.22 (Lema de cuantificación acotada)	25
4.13.2.	Lema 4.30	26
4.13.3.	Lema 4.25: Lema de minimización acotada	26

TODO: ordenar los lemas comunes y sus citaciones. agregar los de los manantiales.

1. Convenciones

Si no se especifica lo contrario, usaremos las siguientes convenciones:

1. $x, y, z, u, v, w, n, m, k, \dots \in \omega$
2. $\alpha, \beta, \gamma, \delta, \varepsilon, \psi, \eta, \dots \in \Sigma^*$
3. $O, s \in \{(\omega, \#), (\Sigma^*, *)\}$
4. «tq» es «tal que»
5. Σ -pr es « Σ -primitivo recursivo»
6. Sea $f : \text{Dom}_f \rightarrow \{0, 1\}$, entonces f es un predicado.

2. Combos de definiciones y convenciones notacionales

2.1. Combo 1: Defina:

2.1.1. Cuando un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ es llamado Σ -recursivo

Un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ sera llamado Σ -recursivo cuando la función $\chi_S^{\omega^n \times \Sigma^{*m}}$ sea Σ -r.

2.1.2. $\langle s_1, s_2, \dots \rangle$

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

2.1.3. « f es una función Σ -mixta»

Sea Σ un alfabeto finito. Una función f es Σ -mixta si:

1. $(\exists n, m \in \omega) \text{Dom}_f \subseteq \omega^n \times \Sigma^{*m}$
2. $\text{Im}_f \subseteq O$

2.1.4. «familia Σ -indexada de funciones»

Dado un alfabeto Σ , una familia Σ -indexada de funciones sera una función $\varrho : \Sigma \rightarrow \text{Im}_G$ donde Im_G es el conjunto de funciones $\varrho(a)$ asociadas a cada $a \in \Sigma$.

NOTACIÓN: Si ϱ es una familia Σ -indexada de funciones, entonces para $a \in \Sigma$, escribiremos ϱ_a en lugar de $\varrho(a)$.

2.1.5. $R(f, \varrho)$: Recursion primitiva sobre variable alfabética con valores numéricos.

Sean $S_1, \dots, S_n \subseteq \omega$ y $L_1, \dots, L_m \subseteq \Sigma^*$ conjuntos no vacíos.

Sea una función $f : S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \omega$.

Sea una familia Σ -indexada de funciones $\varrho_a : \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \times \Sigma^* \rightarrow \omega$ para cada $a \in \Sigma$.

$$\begin{aligned}
R(f, \varrho) : S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \times \Sigma^* &\rightarrow \omega \\
(\vec{x}, \vec{\alpha}, \varepsilon) &\rightarrow f(\vec{x}, \vec{\alpha}) \\
(\vec{x}, \vec{\alpha}, \alpha a) &\rightarrow \varrho_a(R(f, \varrho)(\vec{x}, \vec{\alpha}, \alpha), \vec{x}, \vec{\alpha}, \alpha)
\end{aligned}$$

También diremos que $R(f, \varrho)$ es obtenida por recursion primitiva a partir de f y ϱ .

2.2. Combo 2: Defina:

2.2.1. $d \vdash^n d'$ y $d \vdash^* d'$

(no hace falta que defina \vdash)

- $d \vdash^n d'$ si $(\exists d_2, \dots, d_n \in \text{Des}) d \vdash d_2 \vdash \dots \vdash d_n \vdash d'$.
- $d \vdash^* d'$ sii $(\exists n \in \omega) d \vdash^n d'$

2.2.2. $L(M)$

Llamamos $L(M)$ al conjunto formado por todas las palabras que son aceptadas por alcance de estado final.

Una palabra $\alpha_1 \dots \alpha_n \in \Sigma^*$ es aceptada por M por alcance de estado final si partiendo de $Bq_0 \alpha_1 \dots \alpha_n B \dots$ en algún momento de la computación M esta en un estado de F .

2.2.3. «f es una función de tipo (n, m, s) »

Dada una función Σ -mixta f ,

- Si $f = \emptyset$, entonces es una función de tipo (n, m, s) cualquiera sean $n, m \in \omega$ y $s \in \{\#, *\}$.
- Si $f \neq \emptyset$, entonces hay únicos $n, m \in \omega$ tales que $\text{Dom}_f \subseteq \omega^n \times \Sigma^{*m}$.
 - Si $I_f \subseteq \omega$, entonces es una función de tipo $(n, m, \#)$.
 - Si $I_f \subseteq \Sigma^*$, entonces es una función de tipo $(n, m, *)$.

De esta forma, cuando $f \neq \emptyset$, hablaremos de «el tipo de f » para referirnos a esta única terna (n, m, s) .

2.2.4. (x)

Dado $x \in \mathbb{N}$, usaremos (x) para denotar a la única infinitupla $(s_1, s_2, \dots) \in \omega^{[\mathbb{N}]}$ tq $x = \langle s_1, s_2, \dots \rangle = \prod_{i=1}^{\infty} \text{pr}(i)^{s_i}$

2.2.5. $(x)_i$

Dados $x, i \in \mathbb{N}$, usaremos $(x)_i$ para denotar a s_i de $(s_1, s_2, \dots) = (x)$.

Se le suele llamar la «i-esima bajada de x» al numero $(x)_i$ (al «bajar» el i-esimo exponente de la única posible factorización de x como producto de primos).

2.3. Combo 3: Defina:

2.3.1. Cuando un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ es llamado Σ -re

(no hace falta que defina «función Σ -r»)

Diremos que un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ sera llamado Σ -re cuando sea vacío o haya una función sobreyectiva $F : \omega \rightarrow S$ tq $F(i) = p_i^{n,m} \circ F$ sea Σ -r para cada $i \in \{1, \dots, n+m\}$.

2.3.2. s^{\leq}

Sea \leq un orden sobre Σ^* .

$$S^{\leq} : \Sigma^* \rightarrow \Sigma^*$$

$$(a_n)^m \rightarrow (a_1)^{m+1}$$

$$\alpha a_i (a_n)^m \rightarrow \alpha a_{i+1} (a_1)^m \text{ con } 1 \leq i < n$$

2.3.3. $*^{\leq}$

Sea \leq un orden sobre Σ^* .

$$*^{\leq} : \omega \rightarrow \Sigma^*$$

$$0 \rightarrow \varepsilon$$

$$i+1 \rightarrow s^{\leq}(*^{\leq}(i))$$

2.3.4. $\#^{\leq}$

Sea \leq un orden sobre Σ^* .

$$\#^{\leq} : \Sigma^* \rightarrow \omega$$

$$\varepsilon \rightarrow 0$$

$$a_{i_k} \dots a_{i_0} \rightarrow i_k n^k + \dots + i_0 n^0$$

2.4. Combo 4: Defina cuando una función $f : \text{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow \omega$ es llamada Σ -efectivamente computable y defina «el procedimiento P computa a la función f »

Sea O . Una función Σ -mixta $f : \text{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$ sera llamada Σ -efectivamente computable si hay un procedimiento efectivo P tq

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

En ambos casos diremos que P computa a la función f .

Obs: $f = \emptyset$ es un procedimiento que nunca se detiene cualesquiera sea su dato de entrada. Por lo tanto es Σ -efectivamente computable, cualesquiera sean n, m, O y Σ .

2.5. Combo 5: Defina cuando un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ es llamado Σ -efectivamente computable y defina: «el procedimiento efectivo P decide la pertenencia a S »

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

Si P es un procedimiento efectivo el cual computa a $\chi_S^{\omega^n \times \Sigma^{*m}}$, entonces diremos que P decide la pertenencia a S , con respecto al conjunto $\omega^n \times \Sigma^{*m}$.

Obs: $f = \emptyset$ es un procedimiento que siempre da 0 cualesquiera sea su dato de entrada. Por lo tanto es Σ -efectivamente computable, cualesquiera sean n, m, O y Σ .

2.6. Combo 6: Defina cuando un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ es llamado Σ -efectivamente enumerable y defina: «el procedimiento efectivo P enumera a S »

Un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ sera llamado Σ -efectivamente enumerable cuando sea vacío o haya una función sobreyectiva $F: \omega \rightarrow S$ tq $F_{(i)}$ sea Σ -efectivamente computable, para cada $i \in \{1, \dots, n+m\}$.

2.7. Combo 7: Defina cuando una función $f: \text{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow \omega$ es llamada Σ -Turing computable y defina «la máquina de Turing M computa a la función f »

Diremos que una función $f: \text{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow \Sigma^*$ es Σ -Turing computable si existe una máquina de Turing con unit, $M = (Q, \Sigma^*, \Gamma, \delta, q_0, B, \nu, F)$ tq:

1. Si $(\vec{x}, \vec{\alpha}) \in \text{Dom}_f$, entonces hay un $p \in Q$ tq $[q_0 B \nu^{x_1} B \dots B \nu^{x_n} B \alpha_1 B \dots B \alpha_m] \vdash^* [p B f(\vec{x}, \vec{\alpha})]$ y $[p B f(\vec{x}, \vec{\alpha})] \not\vdash d$ para cada $d \in \text{Des}$
2. Si $(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m} - \text{Dom}_f$, entonces M no se detiene partiendo de $[q_0 B \nu^{x_1} B \dots B \nu^{x_n} B \alpha_1 B \dots B \alpha_m]$.

Cuando una maquina de Turing con unit M cumpla ambos items, diremos que M computa a la función f o que f es computada por M .

Cabe destacar que la condición $[p B f(\vec{x}, \vec{\alpha})] \not\vdash d$ para cada $d \in \text{Des}$ es equivalente a que (p, B) no este en el dominio de δ o que si lo este y que la tercer coordenada de $\delta(p, B)$ sea L .

2.8. Combo 8: Defina:

2.8.1. $M(P)$ Minimización de variable numérica

Sea Σ un alfabeto finito y sea $P: \text{Dom}_P \subseteq \omega^n \times \Sigma^{*m}$. Dado $(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m}$, cuando exista al menos un $t \in \omega$ tq $P(t, \vec{x}, \vec{\alpha}) = 1$, usaremos $\min_t P(t, \vec{x}, \vec{\alpha})$ para denotar al menor de tales t 's.

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

Diremos que $M(P)$ es obtenida por minimización de variable numérica a partir de P .

Obs: $M(P)$ esta definida solo para aquellas $(n + m)$ -uplas $(\vec{x}, \vec{\alpha})$ para las cuales hay al menos un t tq se da $P(t, \vec{x}, \vec{\alpha}) = 1$

2.8.2. Lt

$Lt : \mathbb{N} \rightarrow \omega$

$1 \rightarrow 0$

$x \rightarrow \max_i (x)_i \neq 0$

2.8.3. Conjunto rectangular

Sea Σ un alfabeto finito. Un conjunto Σ -mixto es llamado rectangular si es de la forma $S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m$ con cada $S_i \subseteq \omega$ y cada $L_i \subseteq \Sigma^*$.

2.8.4. « S es un conjunto de tipo (n, m) »

Dado un conjunto Σ -mixto $S \neq \emptyset$, decimos que S es un conjunto de tipo (n, m) para referirnos a los únicos $n, m \in \omega$ tq $S \subseteq \omega^n \times \Sigma^{*m}$

\emptyset es un conjunto de tipo (n, m) cualesquiera sean $n, m \in \omega$ por lo cual cuando hablemos de el tipo de un conjunto deberemos estar seguros de que dicho conjunto es no vacío.

2.9. Combo 9

2.9.1. Conjunto rectangular

Sea Σ un alfabeto finito. Un conjunto Σ -mixto es llamado rectangular si es de la forma $S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m$ con cada $S_i \subseteq \omega$ y cada $L_i \subseteq \Sigma^*$.

2.9.2. « I es una instrucción de S^Σ »

Una instrucción de S^Σ es ya sea una instrucción básica de S^Σ o una palabra de la forma αI , donde $\alpha \in \{L\bar{n} : n \in \mathbb{N}\}$ y I es una instrucción básica de S^Σ . Llamamos Ins^Σ al conjunto de todas las instrucciones de S^Σ .

2.9.3. « P es un programa de S^Σ »

Un programa de S^Σ 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 se cumple la ley de los GOTO.

Ley de los GOTO: Para cada $i \in \{1, \dots, n\}$, si GOTO $L\bar{m}$ es un tramo final de I_i , entonces existe $j \in \{1, \dots, n\}$ tq I_j tiene label $L\bar{m}$.

2.9.4. I_i^P

$\lambda i P[I_i^P] : \omega \times \text{Pro}^\Sigma \rightarrow \Sigma^*$

$$(i, P) \rightarrow \begin{cases} \text{i-esima instrucción de P} & \text{si } i \in \{1, \dots, n(P)\} \\ \varepsilon & \text{si } i \notin \{1, \dots, n(P)\} \end{cases}$$

2.9.5. $n(P)$

$\lambda P[n(P)] : \text{Pro}^\Sigma \rightarrow \omega$

$$P \rightarrow m \text{ tq } P = I_1 I_2 \dots I_m$$

2.9.6. Bas

$\text{Bas} : \text{Ins}^\Sigma \rightarrow (\Sigma \cup \Sigma_p)^*$

$$I \rightarrow \begin{cases} J & \text{si } I \text{ es de la forma } L\bar{k}J \text{ con } J \in \text{Ins}^\Sigma \\ I & \text{c.c.} \end{cases}$$

2.10. Combo 10: Defina relativo al lenguaje S^Σ :

2.10.1. «estado»

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

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

2.10.2. «descripción instantánea»

Es una terna $(i, \vec{x}, \vec{\sigma}) \in \text{Des}^\Sigma = \omega \times \omega^{[\mathbb{N}]} \times \Sigma^{*[\mathbb{N}]}$ tq $(\vec{x}, \vec{\sigma})$ es un estado.

Si $i \in \{1, \dots, n(P)\}$, $(i, \vec{x}, \vec{\sigma})$ nos dice que las variables están en el estado $(\vec{x}, \vec{\sigma})$ y que la instrucción que debemos realizar es I_i^P

2.10.3. S_P

Dado un programa P .

$S_P : \text{Des}^\Sigma \rightarrow \text{Des}^\Sigma$

$$(i, \vec{x}, \vec{\sigma}) \rightarrow \begin{cases} (i, \vec{x}, \vec{\sigma}) & \text{si } i \notin \{1, \dots, n(P)\} \\ (i+1, (s_1, \dots, s_k-1, \dots), \vec{\sigma}) & \text{si Bas}(I_i^P) = N\bar{k} \leftarrow N\bar{k}-1 \\ (i+1, (s_1, \dots, s_k+1, \dots), \vec{\sigma}) & \text{si Bas}(I_i^P) = N\bar{k} \leftarrow N\bar{k}+1 \\ (i+1, (s_1, \dots, s_n, \dots), \vec{\sigma}) & \text{si Bas}(I_i^P) = N\bar{k} \leftarrow N\bar{n} \\ (i+1, (s_1, \dots, 0, \dots), \vec{\sigma}) & \text{si Bas}(I_i^P) = N\bar{k} \leftarrow 0 \\ (i+1, \vec{s}, \vec{\sigma}) & \text{si Bas}(I_i^P) = \text{IF } N\bar{k} \neq 0 \text{ GOTO } L\bar{m} \wedge s_k = 0 \\ (\min\{l: I_l^P \text{ tiene label } L\bar{m}\}, \vec{s}, \vec{\sigma}) & \text{si Bas}(I_i^P) = \text{IF } N\bar{k} \neq 0 \text{ GOTO } L\bar{m} \wedge s_k \neq 0 \\ (i+1, \vec{s}, (\sigma_1, \dots, \sim\sigma_k, \dots)) & \text{si Bas}(I_i^P) = P\bar{k} \leftarrow \sim P\bar{k} \\ (i+1, \vec{s}, (\sigma_1, \dots, \sigma_k a, \dots)) & \text{si Bas}(I_i^P) = P\bar{k} \leftarrow P\bar{k}.a \\ (i+1, \vec{s}, (\sigma_1, \dots, \sigma_{\bar{n}}, \dots)) & \text{si Bas}(I_i^P) = P\bar{k} \leftarrow P\bar{n} \\ (i+1, \vec{s}, (\sigma_1, \dots, \varepsilon, \dots)) & \text{si Bas}(I_i^P) = P\bar{k} \leftarrow \varepsilon \\ (\min\{l: I_l^P \text{ tiene label } L\bar{m}\}, \vec{s}, \vec{\sigma}) & \text{si Bas}(I_i^P) = \text{IF } P\bar{k} \text{ BEGINS } a \text{ GOTO } L\bar{m} \wedge [\sigma_k]_1 = a \\ (i+1, \vec{s}, \vec{\sigma}) & \text{si Bas}(I_i^P) = \text{IF } P\bar{k} \text{ BEGINS } a \text{ GOTO } L\bar{m} \wedge [\sigma_k]_1 \neq a \\ (\min\{l: I_l^P \text{ tiene label } L\bar{m}\}, \vec{s}, \vec{\sigma}) & \text{si Bas}(I_i^P) = \text{GOTO } L\bar{m} \\ (i+1, \vec{s}, \vec{\sigma}) & \text{si Bas}(I_i^P) = \text{SKIP} \end{cases}$$

2.10.4. «estado obtenido luego de t pasos, partiendo del estado $(\vec{x}, \vec{\alpha})$ »

Dado un programa P y la descripción instantánea obtenida luego de t pasos desde el estado $(\vec{x}, \vec{\sigma})$

$$\overbrace{S_P(\dots S_P(S_P(1, \vec{x}, \vec{\sigma}))\dots)}^{t \text{ veces}} = (j, \vec{u}, \vec{\eta})$$

diremos que $(\vec{u}, \vec{\eta})$ es el estado obtenido luego de t pasos, partiendo del estado $(\vec{x}, \vec{\sigma})$.

2.10.5. « P se detiene (luego de t pasos), partiendo desde el estado $(\vec{x}, \vec{\sigma})$ »

Dado $\overbrace{S_P(\dots S_P(S_P(1, \vec{x}, \vec{\sigma}))\dots)}^{t \text{ veces}} = (j, \vec{u}, \vec{\eta})$, si su primer coordenada j es igual a $n(P) + 1$, diremos que P se detiene (luego de t pasos), partiendo desde el estado $(\vec{x}, \vec{\sigma})$.

2.11. Combo 11: Defina:

2.11.1. $\Psi_P^{n,m,\#}$

Dado $P \in \text{Pro}^\Sigma$.

$D_{\Psi_P^{n,m,\#}} = \{(\vec{x}, \vec{\sigma}) \in \omega^n \times \Sigma^{*m} : P \text{ termina partiendo de } \|x_1, \dots, x_n, \alpha_1, \dots, \alpha_m\|\}$

$\Psi_P^{n,m,\#} : D_{\Psi_P^{n,m,\#}} \rightarrow \omega$

$(\vec{x}, \vec{\sigma}) \rightarrow \text{valor de } N_1 \text{ cuando } P \text{ termina partiendo de } \|x_1, \dots, x_n, \alpha_1, \dots, \alpha_m\|$

2.11.2. « f es Σ -computable» y « P computa a f »

Dado $s, O \in \{(\#, \omega), (*, \Sigma^*)\}$. Una función Σ -mixta $f : S \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$ sera llamada Σ -computable si hay un programa P de S^Σ tq $f = \Psi_P^{n,m,s}$.

En tal caso diremos que la función f es computada por P .

2.11.3. $M^\leq(P)$ Minimización de variable alfabética

Sea que $\Sigma \neq \emptyset$. Sea \leq un orden total sobre Σ , \leq puede ser naturalmente extendido a un orden total sobre Σ^* . Sea $P : \text{Dom}_P \subseteq \omega^n \times \Sigma^{*m} \times \Sigma^*$ un predicado. Cuando $(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m}$ es tq existe al menos un $\alpha \in \Sigma^*$ tq $P(\vec{x}, \vec{\alpha}, \alpha) = 1$, usaremos $\min_\alpha^\leq P(\vec{x}, \vec{\alpha}, \alpha)$ para denotar al menor $\alpha \in \Sigma^*$ tq $P(\vec{x}, \vec{\alpha}, \alpha) = 1$.

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

Diremos que $M^\leq(P)$ es obtenida por minimización de variable alfabética a partir de P .

Obs: $M^\leq(P)$ esta definida solo para aquellas $(n + m)$ -uplas $(\vec{x}, \vec{\alpha})$ para las cuales hay al menos un α tq se da $P(\vec{x}, \vec{\alpha}, \alpha) = 1$

2.12. Combo 12: Defina cuando un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ es llamado Σ -computable, cuando es llamado Σ -enumerable y defina «el programa P enumera a S »

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

Un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$ sera llamado Σ -enumerable cuando sea vacío o haya una función sobreyectiva $F : \omega \rightarrow S$ tq $F_{(i)}$ sea Σ -computable, para cada $i \in \{1, \dots, n+m\}$.

Nótese que, un conjunto no vacío $S \subseteq \omega^n \times \Sigma^{*m}$ es Σ -enumerable sii hay programas P_1, \dots, P_{n+m} con dato de entrada $x \in \omega$ tales que:

$$S = \text{Im} \left[\Psi_{P_1}^{1,0,\#}, \dots, \Psi_{P_n}^{1,0,\#}, \Psi_{P_{n+1}}^{1,0,*}, \dots, \Psi_{P_{n+m}}^{1,0,*} \right]$$

Como puede notarse, los programas P_1, \dots, P_{n+m} puestos secuencialmente a funcionar desde el estado $\|x\|$ producen, en forma natural, un procedimiento efectivo que enumera a S . Es decir que los programas P_1, \dots, P_{n+m} enumeran a S .

2.13. Combo 13

Defina:

2.13.1. $i^{n,m}$

$$i^{n,m} : \omega \times \omega^n \times \Sigma^{*m} \times \text{Pro}^\Sigma \rightarrow \omega$$

$$(0, \vec{x}, \vec{\alpha}, P) \rightarrow 1$$

$$(t, \vec{x}, \vec{\alpha}, P) \rightarrow j \text{ tq } \overbrace{S_P(\dots S_P(S_P(1, \vec{x}, \vec{\sigma})))}^{t \text{ veces}} \dots = (j, \vec{u}, \vec{\eta})$$

2.13.2. $E_{\#}^{n,m}$

$$E_{\#}^{n,m} : \omega \times \omega^n \times \Sigma^{*m} \times \text{Pro}^\Sigma \rightarrow \omega^{[\mathbb{N}]}$$

$$(0, \vec{x}, \vec{\alpha}, P) \rightarrow (x_1, \dots, x_n, 0, \dots)$$

$$(t, \vec{x}, \vec{\alpha}, P) \rightarrow \vec{u} \text{ tq } \overbrace{S_P(\dots S_P(S_P(1, \vec{x}, \vec{\sigma})))}^{t \text{ veces}} \dots = (j, \vec{u}, \vec{\eta})$$

2.13.3. $E_{\#}^{n,m} + E_{*}^{n,m}$

$$E_{*}^{n,m} : \omega \times \omega^n \times \Sigma^{*m} \times \text{Pro}^\Sigma \rightarrow \Sigma^{*[\mathbb{N}]}$$

$$(0, \vec{x}, \vec{\alpha}, P) \rightarrow (\alpha_1, \dots, \alpha_n, \varepsilon, \dots)$$

$$(t, \vec{x}, \vec{\alpha}, P) \rightarrow \vec{\eta} \text{ tq } \overbrace{S_P(\dots S_P(S_P(1, \vec{x}, \vec{\sigma})))}^{t \text{ veces}} \dots = (j, \vec{u}, \vec{\eta})$$

2.13.4. $E_{\#j}^{n,m}$

$$E_{\#j}^{n,m} : \omega \times \omega^n \times \Sigma^{*m} \times \text{Pro}^\Sigma \rightarrow \omega$$

$$E_{\#j}^{n,m} = p_j^{n,m} \circ E_{\#}^{n,m}$$

2.13.5. $E_{*j}^{n,m}$

$$E_{*j}^{n,m} : \omega \times \omega^n \times \Sigma^{*m} \times \text{Pro}^\Sigma \rightarrow \Sigma^*$$

$$E_{*j}^{n,m} = p_j^{n,m} \circ E_{*}^{n,m}$$

2.13.6. $\text{Halt}^{n,m}$

$\text{Halt}^{n,m} : \omega \times \omega^n \times \Sigma^{*m} \times \text{Pro}^\Sigma \rightarrow \{0, 1\}$

$$(t, \vec{x}, \vec{\sigma}, P) \rightarrow i^{n,m}(t, \vec{x}, \vec{\sigma}, P) = n(P) + 1$$

2.13.7. $T^{n,m}$

$\text{Dom}_{T^{n,m}} = \{(\vec{x}, \vec{\sigma}, P) : P \text{ se detiene partiendo de } \|x_1, \dots, x_n, \alpha_1, \dots, \alpha_m\|\}$

$T^{n,m} : \text{Dom}_{T^{n,m}} \rightarrow \omega$

$$(t, \vec{x}, \vec{\sigma}, P) \rightarrow \min_t(\text{Halt}^{n,m}(t, \vec{x}, \vec{\sigma}, P))$$

2.13.8. AutoHalt^Σ

Dado $\Sigma \supseteq \Sigma_p$

$\text{AutoHalt}^\Sigma : \text{Pro}^\Sigma \rightarrow \{0, 1\}$

$$P \rightarrow (\exists t \in \omega) \text{Halt}^{0,1}(t, P, P)$$

2.13.9. Los conjuntos A y N

Dado $\Sigma \supseteq \Sigma_p$

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

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

2.14. Combo 14: Explique en forma detallada la notación lambda

Usamos la notación lambda de Church de la forma descrita a continuación.

Esta notación se define en función de un alfabeto finito previamente fijado, que denotaremos por Σ .

Solo se usan expresiones tq:

1. Variables permitidas:

- Se usan **variables numéricas** que se valúan en números de (ω) , y se denotan por letras como $x, y, z, u, v, w, n, m, k, \dots$
- Se usan **variables alfabéticas** que se valúan en palabras sobre el alfabeto Σ . Se denotan por letras como $\alpha, \beta, \gamma, \delta, \varepsilon, \psi, \eta, \dots$

2. **Dominio parcial:** Las expresiones lambda pueden ser **parcialmente definidas**. Es decir, puede haber valuaciones de sus variables para las cuales la expresión no este definida.

3. **Libertad sintáctica:** Las expresiones pueden ser descritas informalmente.

4. **Valores booleanos:** Consideramos que las expresiones booleanas toman valores en el conjunto $\{0, 1\} \subseteq \omega$ (usando 0 para falso y 1 para verdadero).

Dado un alfabeto Σ a las expresiones que cumplan las características dadas anteriormente las llamaremos **lambdificables** con respecto a Σ .

2.15. Combo 15: Dada una función $f : \text{Dom}_f \subseteq \omega \times \Sigma^* \rightarrow \omega$, describa qué tipo de objeto es y qué propiedades debe tener el macro: $[V2 \leftarrow f(V1, W1)]$

Dada una función $f : \text{Dom}_f \subseteq \omega \times \Sigma^* \rightarrow \omega$ Σ -computable, la palabra

$$V\bar{2} \leftarrow f(V1, W1)$$

denota a un macro M que cumple lo siguiente:

1. Sus variables oficiales son: $V1, V2, W1$
2. No tiene labels oficiales.
3. Si reemplazamos (tanto oficiales como auxiliares en cada caso):
 1. Las variables $V\bar{k}'$ por variables concretas $N\bar{k}$ con k distintos entre si.
 2. Las variables $W\bar{j}'$ por variables concretas $P\bar{j}$ con j distintos entre si.
 3. Los labels $A\bar{z}'$ por labels concretos $L\bar{z}$ con z distintos entre si.

Obtenemos la palabra $N\bar{k}_2 \leftarrow f(N\bar{k}_1, P\bar{j}_1)$ la cual es un programa de S^Σ .

El cual debe cumplir que: Si lo hacemos correr partiendo de un estado e que le asigne a las variables $N\bar{k}_1, N\bar{k}_2, P\bar{j}_1$ valores x_1, x_2, α_1 , se dará que

1. Si $(x_1, \alpha_1) \notin \text{Dom}_P$, el programa no se detiene.
2. Si $(x_1, \alpha_1) \in \text{Dom}_P$, luego de una cantidad finita de pasos el programa se detiene llegando a un estado e' tq:
 1. e' asigna a $N\bar{k}_2$ el valor $f(x_1, \alpha_1)$;
 2. e' solo difiere de e en el valor de $N\bar{k}_2$ y en las variables que reemplazaron a las auxiliares de M .

La palabra $N\bar{k}_2 \leftarrow f(N\bar{k}_1, P\bar{j}_1)$ se denomina la expansión del macro $V2 \leftarrow f(V1, W1)$ respecto de la elección concreta de variables y labels realizada.

2.16. Combo 16: Dado un predicado $p : \text{Dom}_p \subseteq \omega \times \Sigma^* \rightarrow \{0, 1\}$, describa qué tipo de objeto es y qué propiedades debe tener el macro: $[IF P(V1, W1) GOTO A1]$

Dado un predicado $P : \text{Dom}_P \subseteq \omega \times \Sigma^* \rightarrow \{0, 1\}$ Σ -computable, la palabra

$$[IF P(V1, W1) GOTO A1]$$

denota a un macro M que cumple lo siguiente:

1. Sus variables oficiales son: $V1, W1$
2. $A1$ es su único label oficial.
3. Si reemplazamos (tanto oficiales como auxiliares en cada caso):
 1. Las variables $V\bar{k}'$ por variables concretas $N\bar{k}$ con k distintos entre si.
 2. Las variables $W\bar{j}'$ por variables concretas $P\bar{j}$ con j distintos entre si.
 3. Los labels $A\bar{z}'$ por labels concretos $L\bar{z}$ con z distintos entre si.

Obtenemos la palabra $[IF P(N\bar{k}_1, P\bar{j}_1) GOTO L\bar{z}_1]$ la cual, si se cumple la ley del GOTO respecto a $L\bar{z}_1$, es un programa de S^Σ .

El cual debe cumplir que: Si lo hacemos correr partiendo de un estado e que le asigne a las variables $\overline{Nk_1}, \overline{Pj_1}$ valores x_1, α_1 , se dará que

1. Si $(x_1, \alpha_1) \notin \text{Dom}_P$, el programa no se detiene.
2. Si $(x_1, \alpha_1) \in \text{Dom}_P$, luego de una cantidad finita de pasos:
 1. Si $P(x_1, \alpha_1) = 1$, se salta al label $L\overline{z_1}$.
 2. Si $P(x_1, \alpha_1) = 0$, el programa se detiene.

En ambos casos, el estado alcanzado e' solo puede diferir de e en las variables que reemplazaron a las auxiliares de M .

La palabra $[\text{IF } P(\overline{Nk_1}, \overline{Pj_1}) \text{ GOTO } L\overline{z_1}]$ se denomina la expansión del macro $[\text{IF } P(V1, W1) \text{ GOTO } A1]$ respecto de la elección concreta de variables y labels realizada.

2.17. Combo 17: Defina el concepto de función y desarrolle las tres Convenciones Notacionales asociadas a dicho concepto

Una función es un conjunto de pares tq, si $(x, y) \in f$ y $(x, z) \in f$, entonces $y = z$.

Dada una función f , definimos:

- $\text{Dom}_f = \{x : (x, y) \in f \text{ para algún } y\}$
- $\text{Im}_f = \{y : (x, y) \in f \text{ para algún } x\}$

Las convenciones notacionales son:

- Dado $x \in \text{Dom}_f$, usaremos $f(x)$ para denotar al único $y \in \text{Im}_f$ tq $(x, y) \in f$.
- Escribimos $f : S \subseteq A \rightarrow B$ para expresar que f es una función tq $\text{Dom}_f = S \subseteq A$ y $\text{Im}_f \subseteq B$. También escribimos $f : A \rightarrow B$ si $S = A$. En tal contexto llamaremos a B conjunto de llegada.
- Muchas veces para definir una función f , lo haremos dando su dominio y su regla de asignación. Esto determina por completo a f ya que $f = \{(x, f(x)) : x \in \text{Dom}_f\}$.

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

3. Combos de teoremas

3.1. Combo 1

3.1.1. Proposición: Caracterización de conjuntos Σ -pr

Sea $S \subseteq \omega^n \times \Sigma^{*m}$. Entonces, S es Σ -pr sii S es el dominio de alguna función Σ -pr. (En la inducción de la prueba hacer solo el caso de la composición)

Prueba \Rightarrow

Sea S Σ -pr.

Entonces, $\chi_S^{\omega^n \times \Sigma^{*m}}$ es Σ -pr para algún $n, m \in \omega$.

Para ese caso, $\text{pred} \circ \chi_S^{\omega^n \times \Sigma^{*m}}$ es una función Σ -pr y $S = \text{Dom}_{\text{pred} \circ \chi_S^{\omega^n \times \Sigma^{*m}}}$

Prueba \Leftarrow

Sea S el dominio de una función Σ -pr $f : \text{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$.

Probaremos por inducción en k que Dom_F es Σ -pr, para cada $F \in \text{PR}_k^\Sigma$:

1. Caso $k = 0$: $\text{PR}_0^\Sigma = \{\text{suc}, \text{pred}, C_0^{0,0}, C_\varepsilon^{0,0}\} \cup \{d_a : a \in \Sigma\} \cup \{p_j^{n,m} : 1 \leq j \leq n+m\}$
 Los dominios de las funciones $\text{suc}, C_0^{0,0}, C_\varepsilon^{0,0}, d_a, p_j^{n,m}$ son de la forma $\omega^n \times \Sigma^{*m}$ y ω y Σ^* son Σ -pr, por el «Lema 4.32» son Σ -pr.
 Finalmente, $\chi_{\text{Dom}_{\text{Pred}}}^\omega = \lambda x[x \neq 0]$ es Σ -pr, por definición Dom_{Pred} es Σ -pr
2. Supongamos que Dom_F es Σ -pr $\forall F \in \text{PR}_k^\Sigma$.

3. Sea $F \in \text{PR}_{k+1}^\Sigma$. Veremos entonces que Dom_F es Σ -pr solo para el caso de composición:

Si $F = \emptyset$, entonces es claro que $\text{Dom}_F = \emptyset$ es Σ -pr.

Sea $F = g \circ [g_1, \dots, g_{n+m}]$ no vacío, con $g, g_1, \dots, g_{n+m} \in \text{PR}_k^\Sigma$.

- $g : \text{Dom}_g \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$
- $g_i : \text{Dom}_{g_i} \subseteq \omega^k \times \Sigma^{*l} \rightarrow \omega$ para $i = 1, \dots, n$
- $g_i : \text{Dom}_{g_i} \subseteq \omega^k \times \Sigma^{*l} \rightarrow \Sigma^*$ para $i = n+1, \dots, n+m$

Por hipótesis inductiva, los conjuntos $\text{Dom}_g, \text{Dom}_{g_i}$ son Σ -pr.

Por «Lema 4.31», $S = \bigcap_{i=1}^{n+m} \text{Dom}_{g_i}$ es Σ -pr.

Por «Lema 4.35» y «Lema 4.34», $\chi_{\text{Dom}_F}^{\omega^k \times \Sigma^{*l}}(\vec{x}, \vec{\alpha}) = \begin{cases} \chi_{\text{Dom}_g}^{\omega^n \times \Sigma^{*m} \circ [g_1, \dots, g_{n+m}]} & \text{si } (\vec{x}, \vec{\alpha}) \in S \\ C_0^{k,l} & \text{si } (\vec{x}, \vec{\alpha}) \in \omega^k \times \Sigma^{*l} - S \end{cases}$ es Σ -pr.

Por lo tanto Dom_F es Σ -pr

3.1.2. Teorema: Neumann vence a Gödel

Si h es Σ -r, entonces h es Σ -computable. (En la inducción de la prueba hacer solo el caso $h = R(f, \varrho)$, con $I_h \subseteq \omega$)

Prueba:

Probaremos por inducción en k que: Si $h \in R_k^\Sigma$, entonces h es Σ -computable:

1. Caso $k=0$: $R_0^\Sigma = \text{PR}_0^\Sigma = \{\text{suc}, \text{pred}, C_0^{0,0}, C_\varepsilon^{0,0}\} \cup \{d_a : a \in \Sigma\} \cup \{p_j^{n,m} : 1 \leq j \leq n+m\}$ Por lo que dados los programas que los computan (dejado al lector), entonces son Σ -computables.
2. Supongamos que $h \in R_k^\Sigma \Rightarrow h$ es Σ -computable.
3. Veamos que $h \in R_{k+1}^\Sigma - R_k \Rightarrow h$ es Σ -computable para el caso $h = R(f, \varrho)$ con $\text{Im}_h \subseteq \omega$.

Sean

- $\Sigma = \{a_1, \dots, a_r\}$
- $(f \in R_k^\Sigma) f : S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \omega$
- $(\forall a \in \Sigma, \varrho_a \in R_k^\Sigma) \varrho_a : \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \times \Sigma^* \rightarrow \omega$

Por hipótesis inductiva, f y cada ϱ_a son Σ -computables por lo que existen sus macros.

Recordemos:

$$\begin{aligned}
R(f, \varrho) : S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \times \Sigma^* &\rightarrow \omega \\
(\vec{x}, \vec{\alpha}, \varepsilon) &\rightarrow f(\vec{x}, \vec{\alpha}) \\
(\vec{x}, \vec{\alpha}, \alpha a) &\rightarrow \varrho_a(R(f, \varrho)(\vec{x}, \vec{\alpha}, \alpha), \vec{x}, \vec{\alpha}, \alpha)
\end{aligned}$$

Entonces, construimos el siguiente programa usando macros:

$$\begin{aligned}
&N\overline{n+1} \leftarrow f(N1, \dots, N\overline{n}, P1, \dots, P\overline{m}) \\
Lr+1 : &\text{IF } P\overline{m+1} \text{ BEGINS } a_1 \text{ GOTO } L1 \\
&\vdots \\
&\text{IF } P\overline{m+1} \text{ BEGINS } a_r \text{ GOTO } Lr \\
&\text{GOTO } Lr+2 \\
L1 : &P\overline{m+1} \leftarrow \sim P\overline{m+1} \\
&N\overline{n+1} \leftarrow \varrho_{a_1}(N\overline{n+1}, N1, \dots, N\overline{n}, P1, \dots, P\overline{m}, P\overline{m+2}) \\
&P\overline{m+2} \leftarrow P\overline{m+2}.a_1 \\
&\text{GOTO } Lr+1 \\
&\vdots \\
Lr : &P\overline{m+1} \leftarrow \sim P\overline{m+1} \\
&N\overline{n+1} \leftarrow \varrho_{a_r}(N\overline{n+1}, N1, \dots, N\overline{n}, P1, \dots, P\overline{m}, P\overline{m+2}) \\
&P\overline{m+2} \leftarrow P\overline{m+2}.a_r \\
&\text{GOTO } Lr+1 \\
Lr+2 : &N1 \leftarrow N\overline{n+1}
\end{aligned}$$

Este programa computa h .

3.2. Combo 2

3.2.1. Lema 4.35: Lema de división por casos para funciones Σ -pr

Si $f_i : \text{Dom}_{f_i} \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$ para $i = 1, \dots, k$ son Σ -pr tq si $i \neq j \Rightarrow \text{Dom}_{f_i} \cap \text{Dom}_{f_j} = \emptyset$, entonces la función $f = \bigcup_{i=1}^k f_i$ es también Σ -pr. (Hacer el caso $O = \Sigma^*$, $k = 2$, $n = 2$ y $m = 1$)

Prueba:

Supongamos $O = \Sigma^*$, $i = 1, 2$, $n = 2$ y $m = 1$.

Sean $f_i : \text{Dom}_{f_i} \subseteq \omega^2 \times \Sigma^{*2} \rightarrow \Sigma^*$ Σ -pr tq si $i \neq j \Rightarrow \text{Dom}_{f_i} \cap \text{Dom}_{f_j} = \emptyset$.

Por «Lema 4.34», existen funciones Σ -totales Σ -pr $\overline{f}_i : \omega^2 \times \Sigma^{*2} \rightarrow \Sigma^*$ tq $f_i = \overline{f}_i|_{\text{Dom}_{f_i}}$.

Por «Proposición 19», los conjuntos Dom_{f_1} y Dom_{f_2} son Σ -pr.

Por lo tanto, por «Lema 4.31», también lo es su unión: $\text{Dom}_{f_1} \cup \text{Dom}_{f_2}$.

Finalmente, por «Lema 4.33»,

$$f_1 \cup f_2 = \left(\lambda \alpha \beta [\alpha \beta] \circ \left[\lambda x \alpha [\alpha^x] \circ \left[\chi_{\text{Dom}_{f_1}}^{\omega^n \times \Sigma^{*m}}, \overline{f}_1 \right] \cup \lambda x \alpha [\alpha^x] \circ \left[\chi_{\text{Dom}_{f_2}}^{\omega^n \times \Sigma^{*m}}, \overline{f}_2 \right] \right] \right) |_{\text{Dom}_{f_1} \cup \text{Dom}_{f_2}}$$

es Σ -pr.

3.2.2. Proposición: Caracterización básica de conjuntos Σ -enumerables

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

1. S es Σ -enumerable
2. Hay un programa $\mathbb{P} \in \text{Pro}^\Sigma$ tq:
 1. Para cada $x \in \omega$, \mathbb{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 $(x_1, \dots, x_n, \alpha_1, \dots, \alpha_m) \in S$
 2. Para cada $(x_1, \dots, x_n, \alpha_1, \dots, \alpha_m) \in S$ hay un $x \in \omega$ tq \mathbb{P} se detiene partiendo desde el estado $\|x\|$ y llega a un estado como en $((x_1, \dots, x_n, y_1, \dots), (\alpha_1, \dots, \alpha_m, \beta_1, \dots))$

(Hacer el caso $n = 2$ y $m = 1$)

Sea $n = 2$, $m = 1$ e $i = 1, 2, 3$.

Prueba \Rightarrow :

Hipótesis: Dado $S \subseteq \omega^2 \times \Sigma^{*1}$ no vacío y Σ -enumerable.

Por definición existe una función sobreyectiva $F : \omega \rightarrow S$ tq $F_{(i)}$ son Σ -computable.

Por «Proposición 4.8», existen macros para $F_{(1)}$, $F_{(2)}$ y $F_{(3)}$.

Sea \mathbb{Q} el siguiente programa:

$$P1 \leftarrow F_{(3)}(N1)$$

$$N2 \leftarrow F_{(2)}(N1)$$

$$N1 \leftarrow F_{(1)}(N1)$$

donde:

- Ninguna expansion usa las variables auxiliares $N1, N2, P1$
- Dos expansiones distintas no usan el mismo label auxiliar.

\mathbb{Q} cumple las condiciones ya que emula el comportamiento de F quien enumera a S .

Prueba \Leftarrow :

Hipótesis: Dado $\mathbb{P} \in \text{Pro}^\Sigma$ tq cumple las condiciones (a) y (b).

Sean:

- $\mathbb{P}_1 = \mathbb{P}N1 \leftarrow N1$
- $\mathbb{P}_2 = \mathbb{P}N1 \leftarrow N2$
- $\mathbb{P}_3 = \mathbb{P}P1 \leftarrow P1$

Tenemos:

- $F_{(1)} = \Psi_{\mathbb{P}_1}^{1,0,\#}$
- $F_{(2)} = \Psi_{\mathbb{P}_2}^{1,0,\#}$
- $F_{(3)} = \Psi_{\mathbb{P}_3}^{1,0,*}$

Por definición, cada $F_{(i)}$ es Σ -computable.

Por lo tanto $F = [F_{(1)}, F_{(2)}, F_{(3)}]$ es Σ -computable.

Por hipótesis, dado que F emula a \mathbb{P} , $\text{Dom}_F = \omega$ y $\text{Im}_F = S$.

Por lo tanto S es Σ -enumerable.

3.3. Combo 3

3.3.1. Teorema: Gödel vence a Neumann

Si $f : \text{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$ es Σ -computable, entonces f es Σ -r (Hacer caso $O = \Sigma^*$)

Prueba:

Hipótesis: Sea $O = \Sigma^*$ y dado el programa \mathbb{P}_0 que computa a f .

$$f = \Phi_*^{n,m} \circ [p_1^{n,m}, \dots, p_{n+m}^{n,m}, C_{\mathbb{P}_0}^{n,m}]$$

donde $p_1^{n,m}, \dots, p_{n+m}^{n,m}$ (respecto del alfabeto $\Sigma \cup \Sigma_p$) y $\Phi_*^{n,m}$ son $(\Sigma \cup \Sigma_p)$ -recursivas.

Por lo tanto f es $(\Sigma \cup \Sigma_p)$ -recursiva.

Por «Teorema 4.2», f es Σ -r.

3.3.2. Teorema: Caracterización de conjuntos Σ -efectivamente computables

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

1. S es Σ -efectivamente computable
2. S y $(\omega^n \times \Sigma^{*m}) - S$ son Σ -efectivamente enumerables

(Hacer solo \Leftarrow)

Prueba \Leftarrow :

Hipótesis: Dado que $S \subseteq \omega^n \times \Sigma^{*m}$ y S y $(\omega^n \times \Sigma^{*m}) - S$ son Σ -efectivamente enumerables.

Si $S = \emptyset$ o $S = \omega^n \times \Sigma^{*m}$, entonces es claro que S es Σ -efectivamente computable.

Supongamos que S no es vacío ni Σ -total.

Por definición, existen procedimientos efectivos \mathbb{P}_1 y \mathbb{P}_2 que enumeran a S y $(\omega^n \times \Sigma^{*m}) - S$.

El siguiente procedimiento efectivo computa a $\chi_S^{\omega^n \times \Sigma^{*m}}$:

Dada la entrada $(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m}$

$E1$: Asignar a T el valor 0

$E2$: Realizar \mathbb{P}_1 con entrada T obteniendo $(\vec{y}, \vec{\beta})$

$E3$: Realizar \mathbb{P}_2 con entrada T obteniendo $(\vec{z}, \vec{\gamma})$

$E4$: Si $(\vec{x}, \vec{\alpha}) = (\vec{y}, \vec{\beta})$ entonces detenerse y devolver 1

$E5$: Si $(\vec{x}, \vec{\alpha}) = (\vec{z}, \vec{\gamma})$ entonces detenerse y devolver 0

$E6$: Asignar a T el valor $T + 1$

$E7$: Saltar a $E2$

Por lo tanto, S es Σ -efectivamente computable.

3.4. Combo 4

3.4.1. Proposición: Caracterización básica de conjuntos Σ -enumerables

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

1. S es Σ -enumerable
2. Hay un programa $\mathbb{P} \in \text{Pro}^\Sigma$ tq:
 1. Para cada $x \in \omega$, \mathbb{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 $(x_1, \dots, x_n, \alpha_1, \dots, \alpha_m) \in S$
 2. Para cada $(x_1, \dots, x_n, \alpha_1, \dots, \alpha_m) \in S$ hay un $x \in \omega$ tq \mathbb{P} se detiene partiendo desde el estado $\|x\|$ y llega a un estado como en $((x_1, \dots, x_n, y_1, \dots), (\alpha_1, \dots, \alpha_m, \beta_1, \dots))$

(Hacer el caso $n = 2$ y $m = 1$)

Sea $n = 2$, $m = 1$ e $i = 1, 2, 3$.

Prueba \Rightarrow :

Hipótesis: Dado $S \subseteq \omega^2 \times \Sigma^{*1}$ no vacío y Σ -enumerable.

Por definición existe una función sobreyectiva $F : \omega \rightarrow S$ tq $F_{(i)}$ son Σ -computable.

Por «Proposición 4.8», existen macros para $F_{(1)}$, $F_{(2)}$ y $F_{(3)}$.

Sea \mathbb{Q} el siguiente programa:

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

donde:

- Ninguna expansion usa las variables auxiliares $N1, N2, P1$
- Dos expansiones distintas no usan el mismo label auxiliar.

\mathbb{Q} cumple las condiciones ya que emula el comportamiento de F quien enumera a S .

Prueba \Leftarrow :

Hipótesis: Dado $\mathbb{P} \in \text{Pro}^\Sigma$ tq cumple las condiciones (a) y (b).

Sean:

- $\mathbb{P}_1 = \mathbb{P}N1 \leftarrow N1$
- $\mathbb{P}_2 = \mathbb{P}N1 \leftarrow N2$
- $\mathbb{P}_3 = \mathbb{P}P1 \leftarrow P1$

Tenemos:

- $F_{(1)} = \Psi_{\mathbb{P}_1}^{1,0,\#}$
- $F_{(2)} = \Psi_{\mathbb{P}_2}^{1,0,\#}$
- $F_{(3)} = \Psi_{\mathbb{P}_3}^{1,0,*}$

Por definición, cada $F_{(i)}$ es Σ -computable.

Por lo tanto $F = [F_{(1)}, F_{(2)}, F_{(3)}]$ es Σ -computable.

Por hipótesis, dado que F emula a \mathbb{P} , $\text{Dom}_F = \omega$ y $\text{Im}_F = S$.

Por lo tanto S es Σ -enumerable.

3.4.2. Lema: Lema de la sumatoria

Sea Σ 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 Σ -pr, con $S_1, \dots, S_n \subseteq \omega$ y $L_1, \dots, L_m \subseteq \Sigma^*$ no vacíos, entonces, la función $\lambda xy\vec{x}\vec{\alpha} \left[\sum_{i=x}^y f(i, \vec{x}, \vec{\alpha}) \right]$ es también Σ -pr.

Prueba:

$$\begin{aligned} \lambda xy\vec{x}\vec{\alpha} \left[\sum_{i=x}^y f(i, \vec{x}, \vec{\alpha}) \right] &= \lambda yx\vec{x}\vec{\alpha} \left[\sum_{t=0}^y h(t, x, \vec{x}, \vec{\alpha}) \right] \circ [p_2^{2+n,m}, p_1^{2+n,m}, p_3^{2+n,m}, \dots, p_{2+n+m}^{2+n,m}] \\ &= R(h, g) \circ [p_2^{2+n,m}, p_1^{2+n,m}, p_3^{2+n,m}, \dots, p_{2+n+m}^{2+n,m}] \end{aligned}$$

Donde:

$$\begin{aligned} h : \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m &\rightarrow \omega \\ (x, \vec{x}, \vec{\alpha}) &\rightarrow \begin{cases} 0 & \text{si } (x, \vec{x}, \vec{\alpha}) \in H_1 \\ f(0, \vec{x}, \vec{\alpha}) & \text{si } (x, \vec{x}, \vec{\alpha}) \in H_2 \end{cases} \\ g : \omega^3 \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m &\rightarrow \omega \\ (A, t, x, \vec{x}, \vec{\alpha}) &\rightarrow \begin{cases} 0 & \text{si } (A, t, x, \vec{x}, \vec{\alpha}) \in G_1 \\ A + f(t+1, \vec{x}, \vec{\alpha}) & \text{si } (A, t, x, \vec{x}, \vec{\alpha}) \in G_2 \end{cases} \end{aligned}$$

Donde:

$$\begin{aligned} H_1 &= \{(x, \vec{x}, \vec{\alpha}) \in \text{Dom}_h : x > 0\} \\ H_2 &= \{(x, \vec{x}, \vec{\alpha}) \in \text{Dom}_h : x = 0\} \\ G_1 &= \{(A, t, x, \vec{x}, \vec{\alpha}) \in \text{Dom}_g : x > t+1\} \\ G_2 &= \{(A, t, x, \vec{x}, \vec{\alpha}) \in \text{Dom}_g : x \leq t+1\} \end{aligned}$$

Por «Lema 4.35» y «Lema 4.34», h y g son Σ -pr.

3.5. Combo 5

3.5.1. Lema, Ejercicio de la Guía 5

Sean $\Sigma = \{\textcircled{\cdot}, \%, !\}$ y $S_1, S_2 \subseteq \omega$ y $L_1, L_2 \subseteq \Sigma^*$ no vacíos.

Sea $f : S_1 \times S_2 \times L_1 \times L_2 \rightarrow \omega$ y ϱ una familia Σ -indexada de funciones $\varrho_a : \omega \times S_1 \times S_2 \times L_1 \times L_2 \times \Sigma^* \rightarrow \omega$ para cada $a \in \Sigma$.

Si f y cada ϱ_a son Σ -efectivamente computables, entonces $R(f, \varrho) : S_1 \times S_2 \times L_1 \times L_2 \times \Sigma^* \rightarrow \omega$ lo es.

Prueba:

Por definición, existen procedimientos efectivos $\mathbb{P}_f, \mathbb{P}_{\varrho_{\textcircled{\cdot}}}, \mathbb{P}_{\varrho_{\%}}$ y $\mathbb{P}_{\varrho_{!}}$.

El siguiente procedimiento efectivo computa a $R(f, \varrho)$:

Dada la entrada $(x, y, \alpha, \beta, \gamma) \in S_1 \times S_2 \times L_1 \times L_2 \times \Sigma^*$

$E1$: Asignar a τ el valor ε

$E2$: Asignar a A el resultado de correr \mathbb{P}_f con entrada (x, y, α, β)

$E3$: Si $\gamma = \varepsilon$ detenerse y devolver A

$E4$: Si $[\gamma]_1 = @$, Asignar a A el resultado de correr $\mathbb{P}_{\varrho@}$ con entrada $(A, x, y, \alpha, \beta, \tau)$

$E5$: Si $[\gamma]_1 = \%$, Asignar a A el resultado de correr $\mathbb{P}_{\varrho\%}$ con entrada $(A, x, y, \alpha, \beta, \tau)$

$E6$: Si $[\gamma]_1 = !$, Asignar a A el resultado de correr $\mathbb{P}_{\varrho!}$ con entrada $(A, x, y, \alpha, \beta, \tau)$

$E7$: Asignar a τ el resultado de correr $\tau.[\gamma]_1$

$E8$: Asignar a γ el resultado de correr $\simeq \gamma$

$E9$: Saltar a $E3$

3.5.2. Lema: Lema de cuantificación acotada

Sean $P : S \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \{0, 1\}$ y $\bar{S} \subseteq S$ Σ -pr.

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 Σ -pr

Prueba:

Sea:

$$Q : \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \{0, 1\}$$

$$(t, \vec{x}, \vec{\alpha}) \rightarrow \begin{cases} P(t, \vec{x}, \vec{\alpha}) & \text{si } (t, \vec{x}, \vec{\alpha}) \in \bar{S} \\ 1 & \text{si } (t, \vec{x}, \vec{\alpha}) \in \omega - \bar{S} \end{cases}$$

Por «Lema 4.35» y «Lema 4.34», Q es Σ -pr.

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

Por «Lema 4.21», es Σ -pr.

3.6. Combo 6

3.6.1. Lema:

Si $S \subseteq \omega^n \times \Sigma^{*m}$ es Σ -efectivamente computable, entonces S es Σ -efectivamente enumerable

Prueba:

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

Supongamos que S no es vacío.

Por definición existe un procedimiento efectivo que computa a $\chi_S^{\omega^n \times \Sigma^{*m}}$.

El siguiente procedimiento efectivo enumera a S :

Dada la entrada $x \in \omega$

$E1$: Asignar a \vec{x} los resultados de correr $((x))_1, \dots, ((x))_n$

$E2$: Asignar a $\vec{\alpha}$ los resultados de correr $*^{\leq} ((x))_1, \dots, *^{\leq} ((x))_n$

$E3$: Si $\chi_S^{\omega^n \times \Sigma^{*m}}(\vec{x}, \vec{\alpha}) = 1$ detenerse y devolver $(\vec{x}, \vec{\alpha})$

$E4$: Asignar a x el valor 0

$E5$: Asignar a \vec{x} los resultados de correr $((x))_1, \dots, ((x))_n$

$E6$: Asignar a $\vec{\alpha}$ los resultados de correr $*^{\leq} ((x))_1, \dots, *^{\leq} ((x))_n$

$E7$: Si $\chi_S^{\omega^n \times \Sigma^{*m}}(\vec{x}, \vec{\alpha}) = 1$ detenerse y devolver $(\vec{x}, \vec{\alpha})$

$E8$: Asignar a x el resultado de correr $x + 1$

$E9$: Saltar a $E5$

Cada elemento en S es equivalente a un posible estado de $(\vec{x}, \vec{\alpha})$ que tarde o temprano sera alcanzado.

Ademas, dado que $S \neq \emptyset$, el procedimiento efectivo siempre se detiene y devuelve un elemento de S . Por lo tanto, S es Σ -efectivamente enumerable.

3.6.2. Teorema 4.12: Caracterización de conjuntos Σ -re

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

1. S es Σ -re
2. $S = \text{Im}_F$, para alguna $F : \text{Dom}_F \subseteq \omega^k \times \Sigma^{*l} \rightarrow \omega^n \times \Sigma^{*m}$ tq cada $F(i)$ es Σ -r
3. $S = \text{Dom}_f$, para alguna función Σ -r f

(Hacer la prueba de $(2) \rightarrow (3)$, con $k = l = 1$ y $n = m = 2$)

Prueba \Rightarrow :

Hipótesis: Dado $F : \text{Dom}_F \subseteq \omega \times \Sigma^* \rightarrow S \subseteq \omega^2 \times \Sigma^{*2}$ sobreyectiva tq cada $F(i)$ es Σ -r.

Por teorema, existen programas \mathbb{P}_i que computan $F(i)$.

Sea \leq un orden total sobre Σ .

Sea $\text{NotHalt}_{\mathbb{P}_i} = \lambda t x \alpha [\neg \text{Halt}^{1,1}(t, x, \alpha, \mathbb{P}_i)]$

Notar que, al abstraer \mathbb{P} , su dominio esta bajo Σ , por lo que la función es Σ -mixta.

Por definición, $\text{Halt}^{1,1}$ es $(\Sigma \cup \Sigma_p)$ -pr por lo que $\text{NotHalt}_{\mathbb{P}_i}$ también.

Por «Teorema 4.2», $\text{NotHalt}_{\mathbb{P}_i}$ es Σ -pr, por lo que es Σ -computable y existe su macro.

Definimos:

- $\text{NeqE}_{\mathbb{P}_i \#} = \lambda z t x \alpha [y \neq E_{\#1}^{1,1}(t, x, \alpha, P_i)]$ para $i = 1, 2$
- $\text{NeqE}_{\mathbb{P}_i *} = \lambda x t \alpha \beta [\beta \neq E_{*1}^{1,1}(t, x, \alpha, P_i)]$ para $i = 3, 4$

Notar que, al abstraer \mathbb{P} , sus dominios están bajo Σ , por lo que las funciones son Σ -mixtas.

Por definición, $E_s^{1,1}$ son $(\Sigma \cup \Sigma_p)$ -pr por lo que $\text{NeqE}_{\mathbb{P}_i s}$ también.

Por «Teorema 4.2», $\text{NeqE}_{\mathbb{P}_i s}$ son Σ -pr, por lo que son Σ -computable y existen sus macros.

Luego, para algún $j \in \mathbb{N}$, $\lambda x [(x)_j]$ y $\lambda x [*^{\leq} (x)_j]$ son Σ -pr, por lo que son Σ -computable y existen sus macros.

$p_1^{2,2}|_S$ es una función tq $\text{Dom}_{p_1^{2,2}|_S} = S$. Demos un programa que la compute:

$$\begin{aligned}
L1 : N5 &\leftarrow N5 + 1 \\
N4 &\leftarrow (N5)_1 \\
N3 &\leftarrow (N5)_2 \\
P3 &\leftarrow *^{\leq} ((N5)_3) \\
&[\text{IF NotHalt}_{\mathbb{P}_1}(N4, N3, P3) \text{ GOTO } L1] \\
&[\text{IF NotHalt}_{\mathbb{P}_2}(N4, N3, P3) \text{ GOTO } L1] \\
&[\text{IF NotHalt}_{\mathbb{P}_3}(N4, N3, P3) \text{ GOTO } L1] \\
&[\text{IF NotHalt}_{\mathbb{P}_4}(N4, N3, P3) \text{ GOTO } L1] \\
&[\text{IF NeqE}_{\mathbb{P}_1\#}(N1, N4, N3, P3) \text{ GOTO } L1] \\
&[\text{IF NeqE}_{\mathbb{P}_2\#}(N2, N4, N3, P3) \text{ GOTO } L1] \\
&[\text{IF NeqE}_{\mathbb{P}_3*}(P1, N4, N3, P3) \text{ GOTO } L1] \\
&[\text{IF NeqE}_{\mathbb{P}_4*}(P2, N4, N3, P3) \text{ GOTO } L1]
\end{aligned}$$

Por lo tanto, $p_1^{2,2}|_S$ es Σ -computable y Σ -r.

3.7. Combo 7

3.7.1. Lema 4.25: Lema de minimización acotada

Sea $P : \text{Dom}_P \subseteq \omega \times \omega^n \times \Sigma^{*m} \rightarrow \{0, 1\}$ Σ -pr, entonces:

1. $M(P)$ es Σ -r
2. Si existe $f : \omega^n \times \Sigma^{*m} \rightarrow \omega$ Σ -pr tq $M(P)(\vec{x}, \vec{\alpha}) = \min_t P(t, \vec{x}, \vec{\alpha}) \leq f(\vec{x}, \vec{\alpha})$, entonces $M(P)$ es Σ -pr

Prueba 1:

Hipótesis: Dado $P : \text{Dom}_P \subseteq \omega \times \omega^n \times \Sigma^{*m} \rightarrow \{0, 1\}$ Σ -pr.

Entonces, por definición, P es Σ -r.

Sea

$$\begin{aligned}
Q : \omega \times \omega^n \times \Sigma^{*m} &\rightarrow \{0, 1\} \\
(x, \vec{x}, \vec{\alpha}) &\rightarrow \begin{cases} P(x, \vec{x}, \vec{\alpha}) & \text{si } (x, \vec{x}, \vec{\alpha}) \in \text{Dom}_P \\ 0 & \text{si } (x, \vec{x}, \vec{\alpha}) \in \omega \times \omega^n \times \Sigma^{*m} - \text{Dom}_P \end{cases}
\end{aligned}$$

Por «Lema 4.35» y «Lema 4.34», Q es Σ -r.

Notar que $M(Q) = M(P)$.

Como Q es Σ -r, entonces existe un k tq $Q \in R_k^\Sigma$.

Como Q es Σ -total y Σ -r, por definición $M(Q) \in R_{k+1}^\Sigma \subseteq R^\Sigma$.

Prueba 2:

Hipótesis: Dado $P : \text{Dom}_P \subseteq \omega \times \omega^n \times \Sigma^{*m} \rightarrow \{0, 1\}$ Σ -pr y $f : \omega^n \times \Sigma^{*m} \rightarrow \omega$ Σ -pr tq $M(P)(\vec{x}, \vec{\alpha}) = \min_t P(t, \vec{x}, \vec{\alpha}) \leq f(\vec{x}, \vec{\alpha})$.

Ya que $M(P) = M(Q)$, basta probar que $M(Q)$ es Σ -pr.

Nótese que:

$$\begin{aligned}\chi_{\text{Dom}_{M(Q)}}^{\omega^n \times \Sigma^{*m}} &= \lambda \vec{x} \vec{\alpha} \left[(\exists t \in \omega)_{t \leq f(\vec{x}, \vec{\alpha})} Q(t, \vec{x}, \vec{\alpha}) \right] \\ &= \lambda x \vec{x} \vec{\alpha} \left[(\exists t \in \omega)_{t \leq x} Q(t, \vec{x}, \vec{\alpha}) \right] \circ [f, p_1^{n,m}, \dots, p_{n+m}^{n,m}]\end{aligned}$$

Por «Lema 4.22» es Σ -pr, por lo que $\text{Dom}_{M(Q)}$ es Σ -pr.

Sea

$$G = \lambda t \vec{x} \vec{\alpha} \left[Q(t, \vec{x}, \vec{\alpha}) \wedge \neg (\exists j \in \omega)_{j \leq t} (j \neq t \wedge Q(j, \vec{x}, \vec{\alpha})) \right]$$

G es Σ -total y es fácil demostrar que es Σ -pr por «Lema 4.22».

Notar que $G(t, \vec{x}, \vec{\alpha}) = 1 \Leftrightarrow (\vec{x}, \vec{\alpha}) \in \text{Dom}_{M(Q)} \wedge t = M(Q)(\vec{x}, \vec{\alpha})$ Por lo tanto,

$$\begin{aligned}M(Q) &= \lambda \vec{x} \vec{\alpha} \left[\prod_{t=0}^{f(\vec{x}, \vec{\alpha})} t^{G(t, \vec{x}, \vec{\alpha})} \right] \Big|_{\text{Dom}_{M(Q)}} \\ &= \lambda x y \vec{x} \vec{\alpha} \left[\prod_{t=x}^y t^{G(t, \vec{x}, \vec{\alpha})} \right] \circ [C_0^{n,m}, f, p_1^{n,m}, \dots, p_{n+m}^{n,m}]\end{aligned}$$

Por «Lema 4.21», es Σ -pr.

3.7.2. Lema

Si $f : \text{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$ es Σ -r y $S \subseteq \text{Dom}_f$ es Σ -re, entonces $f|_S$ es Σ -r (Hacer solo el caso S no vacío, $n = m = 1$ y $O = \Sigma^*$)

Prueba:

Hipótesis: Dado $f : \text{Dom}_f \subseteq \omega^1 \times \Sigma^{*1} \rightarrow \Sigma^*$ Σ -r y $S \subseteq \text{Dom}_f$ Σ -re no vacío.

Entonces, por definición, existen programas que computan a f , $F_{(1)}$ y $F_{(2)}$ tq $[F_{(1)}, F_{(2)}]$ enumera a S .

El siguiente programa computa a $f|_S$:

```
L2 : [IF N1 ≠ F(1)(N3) GOTO L1]
      [IF P1 = F(2)(N3) GOTO L3]
L1 : N3 ← N3 + 1
      GOTO L2
L3 : [P1 ← f(N1, P1)]
      SKIP
```

3.8. Combo 8

3.8.1. Lema 4.62

Si $\Sigma \supseteq \Sigma_p$, entonces AutoHalt^Σ no es Σ -recursivo

Prueba:

Supongamos que AutoHalt^Σ es Σ -recursivo tq existe su macro.

Sea el siguiente programa \mathbb{P} :

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

Este programa termina partiendo de $\|\mathbb{P}\|$ sii $\text{AutoHalt}^\Sigma(\mathbb{P}) = 0$. Lo cual es un absurdo por definición de AutoHalt^Σ .

3.8.2. Teorema 4.15.

Si $\Sigma \supseteq \Sigma_p$, entonces AutoHalt^Σ no es Σ -efectivamente computable

Prueba:

Si AutoHalt^Σ fuera Σ -efectivamente computable, la Tesis de Church nos diría que es Σ -recursivo, contradiciendo el lema anterior.

3.8.3. Lema 4.63.

Si $\Sigma \supseteq \Sigma_p$. Sea $A = \{\mathbb{P} \in \text{Pro}^\Sigma : \text{AutoHalt}^\Sigma(P) = 1\}$, entonces A es Σ -re y no Σ -r.

Además, el conjunto $N = \{\mathbb{P} \in \text{Pro}^\Sigma : \text{AutoHalt}^\Sigma(P) = 0\}$ no es Σ -re.

Prueba:

Sea $P = \lambda t P[\text{Halt}^{0,1}(t, P, P)]$.

Como P es Σ -pr, por «Lema 4.25» $M(P)$ es Σ -r.

Como $\text{Dom}_{M(P)} = A$, por «Teorema 4.12» A es Σ -re.

Supongamos que N es Σ -re.

Por «Lema 4.35» y «Lema 4.34»,

$$\begin{aligned} \text{AutoHalt}^\Sigma : \text{Pro}^\Sigma &\rightarrow \{0, 1\} \\ P &\rightarrow \begin{cases} 1 & \text{si } P \in A \\ 0 & \text{si } P \in N \end{cases} \end{aligned}$$

es Σ -r, lo que contradice el «Lema 4.62».

Finalmente supongamos A es Σ -r. Entonces el conjunto $N = (\Sigma^* - A) \cap \text{Pro}^\Sigma$ debería serlo, lo cual es absurdo.

3.8.4. Teorema: Neumann vence a Gödel

Si h es Σ -r, entonces h es Σ -computable (Hacer solo el caso $h = M(P)$)

Prueba:

Probaremos por induccion en k :

1. Caso $k = 0$: $R_0^\Sigma = PR_0^\Sigma = \{\text{suc}, \text{pred}, C_0^{0,0}, C_\varepsilon^{0,0}\} \cup \{d_a : a \in \Sigma\} \cup \{p_j^{n,m} : 1 \leq j \leq n + m\}$
 - $\text{suc}: N1 \leftarrow N1 + 1$
 - $\text{pred}: L1 : \text{IF } N1 = 0 \text{ GOTO } L1 \ N1 \leftarrow N1 - 1$
 - $C_0^{0,0}$ y $C_\varepsilon^{0,0}$: SKIP

- $d_a: P1 \leftarrow P1.a$
 - $p_j^{n,m}: Nj \leftarrow Nj$ para $j = 1, \dots, n$
 - $p_j^{n,m}: Pj \leftarrow Pj$ para $j = n+1, \dots, m$
2. Supongamos: Si $h \in R_k^\Sigma$, entonces h es Σ -computable.
3. Veamos para el caso $h = M(P)$ con $P: \omega \times \omega^n \times \Sigma^{*m} \rightarrow \{0, 1\}$ tq $P \in R_k^\Sigma$ si se cumple:
 Si $h \in R_{k+1}^\Sigma$, entonces h es Σ -computable.
 Sea $h = M(P) \in R_{k+1}^\Sigma$.
 Por hipótesis inductiva, P es Σ -computable por lo que tenemos su macro.
 El siguiente programa computa a h :

```

L2 : IF  $P(\overline{Nn+1}, N1, \dots, N\bar{n}, P1, \dots, P\bar{m})$  GOTO L1
       $\overline{Nn+1} \leftarrow \overline{Nn+1} + 1$ 
      GOTO L2
L1 :  $N1 \leftarrow \overline{Nn+1}$ 

```

3.9. Combo 9

3.9.1. Lema: Lema de división por casos para funciones Σ -r

Sean funciones Σ -r $f_i: \text{Dom}_{f_i} \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$ para $i = 1, \dots, k$, tq $\text{Dom}_{f_i} \cap \text{Dom}_{f_j} = \emptyset$ para $i \neq j$. Entonces $f_1 \cup \dots \cup f_k$ es Σ -r (Hacer el caso $k = 2, n = m = 1$ y $O = \omega$)

Prueba:

Por «Tesis de Church», existen programas $\mathbb{P}_1, \mathbb{P}_2$ que computan f_1, f_2 .

Sea $\text{Halt}_{\mathbb{P}_i} = \lambda t x \alpha [\text{Halt}^{1,1}(t, x, \alpha, \mathbb{P}_i)]$

Notar que, al abstraer \mathbb{P} , su dominio esta bajo Σ , por lo que la función es Σ -mixta.

Por definición, $\text{Halt}^{1,1}$ es $(\Sigma \cup \Sigma_p)$ -pr por lo que $\text{Halt}_{\mathbb{P}_i}$ también.

Por «Teorema 4.2», $\text{Halt}_{\mathbb{P}_i}$ es Σ -pr. Por «Tesis de Church» es Σ -computable tq existe su macro.

El siguiente programa computa a $f_1 \cup f_2$:

```

L1 :  $N2 \leftarrow N2 + 1$ 
      [IF  $\text{Halt}_{\mathbb{P}_1}(N2, N1, P1)$  GOTO L2]
      [IF  $\text{Halt}_{\mathbb{P}_2}(N2, N1, P1)$  GOTO L3]
      GOTO L1
L2 :  $[P1 \leftarrow f_1(N1, P1)]$ 
      GOTO L4
L3 :  $[P1 \leftarrow f_2(N1, P1)]$ 
L4 : SKIP

```

Como es Σ -computable, por «Tesis de Church», $f_1 \cup f_2$ es Σ -r.

3.9.2. Teorema 4.5: Gödel vence a Neumann

Si $f: \text{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow \omega$ es Σ -computable, entonces f es Σ -r

Prueba:

Como f es Σ -computable, existe un programa \mathbb{P} que lo computa.
 Nótese que $f = \Phi_s^{n,m} \circ [p_1^{n,m}, \dots, p_{n+m}^{n,m}, C_{\mathbb{P}}^{n,m}]$ bajo el alfabeto $\Sigma \cup \Sigma_p$.
 Por lo tanto f es $(\Sigma \cup \Sigma_p)$ -r. Por «Teorema 4.2», f es Σ -r.

4. Utilidades

4.1. Lema 14.

Sean $P : S \subseteq \omega^n \times \Sigma^{*m} \rightarrow \{0, 1\}$ y $Q : S \subseteq \omega^n \times \Sigma^{*m} \rightarrow \{0, 1\}$ Σ -pr, entonces también lo son: $(P \vee Q)$, $(P \wedge Q)$ y $\neg P$.

4.2. Def Conjuntos Σ -pr

Un conjunto Σ -mixto $S \subseteq \omega^n \times \Sigma^{*m}$ se llama Σ -recursivo primitivo si su función característica $\chi_S^{\omega^n \times \Sigma^{*m}} \equiv \lambda \vec{x} \vec{\alpha} [(\vec{x}, \vec{\alpha}) \in S]$ es Σ -pr

4.3. Lema 4.31.

Si $S_1, S_2 \subseteq \omega^n \times \Sigma^{*m}$ son Σ -pr, entonces también lo son: $S_1 \cup S_2$, $S_1 \cap S_2$ y $S_1 - S_2$.

4.4. Lema 4.32.

Sean $S_1, \dots, S_n \subseteq \omega$ y $L_1, \dots, L_m \subseteq \Sigma^*$ conjuntos no vacíos.

Entonces $S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m$ es Σ -pr sii $S_1, \dots, S_n, L_1, \dots, L_m$ son Σ -pr

4.5. Lema 4.33.

Sea $f : \text{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$ una función Σ -pr. Si $S \subseteq \text{Dom}_f$ es Σ -pr, entonces la función $f|_S$ también es Σ -pr

4.6. Lema 4.34.

Si $f : \text{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$ es Σ -pr, entonces existe una función $\bar{f} : \omega^n \times \Sigma^{*m} \rightarrow O$ Σ -pr tal que $f = \bar{f}|_{\text{Dom}_f}$.

4.7. Proposición 19.

Sea $S \subseteq \omega^n \times \Sigma^{*m}$. Entonces, S es Σ -pr sii S es el dominio de alguna función Σ -pr.

4.8. Lema 4.35: Lema de division por casos para funciones Σ -pr

Si $f_i : \text{Dom}_{f_i} \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$ para $i = 1, \dots, k$ son Σ -pr tq si $i \neq j \Rightarrow \text{Dom}_{f_i} \cap \text{Dom}_{f_j} = \emptyset$, entonces la función $f = \bigcup_{i=1}^k f_i$ es también Σ -pr.

4.9. Lema 4.21.

Sea Sigma un alfabeto finito.

(a) Si $f : \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \omega$ es Σ -pr, con $S_1, \dots, S_n \subseteq \omega$ y $L_1, \dots, L_m \subseteq \Sigma^*$ no vacíos, entonces, las funciones $\lambda xy \vec{x} \vec{\alpha}. \sum_{t=x}^y f(t, \vec{x}, \vec{\alpha})$ y $\lambda xy \vec{x} \vec{\alpha}. \prod_{t=x}^y f(t, \vec{x}, \vec{\alpha})$ son también Σ -pr

(b) Si $f : \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \Sigma^*$ es Σ -pr, con $S_1, \dots, S_n \subseteq \omega$ y $L_1, \dots, L_m \subseteq \Sigma^*$ no vacíos, entonces la función $\lambda x y \vec{x} \vec{\alpha}. \bigcup_{t=x} f(t, \vec{x}, \vec{\alpha})$ es Σ -pr

4.10. Proposición 4.8

Sea $f : \text{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow \omega$ una función Σ -computable. Entonces, en S^Σ hay un macro de la forma: $V\overline{n} + \overline{1} \leftarrow f(V\overline{1}, \dots, V\overline{n}, W\overline{1}, \dots, W\overline{m})$

Sea $f : \text{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow \Sigma^*$ una función Σ -computable. Entonces, en S^Σ hay un macro de la forma: $W\overline{n} + \overline{1} \leftarrow f(V\overline{1}, \dots, V\overline{n}, W\overline{1}, \dots, W\overline{m})$

4.11. Definición de función Σ -computable

Una función Σ -mixta $f : S \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$ será llamada Σ -computable si hay un programa P tal que $f = \Psi_P^{n,m,s}$.

4.12. Definición de conjuntos Σ -enumerables

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

4.13. Teorema 4.2 (Independencia del alfabeto)

Sean Σ y Γ alfabetos cualesquiera:

1. (a) Supongamos que una función f es Σ -mixta y Γ -mixta. Entonces f es Σ -r (resp. Σ -pr) sii f es Γ -recursiva (resp. Γ -pr).
2. (b) Supongamos que un conjunto S es Σ -mixto y Γ -mixto. Entonces S es Σ -recursivo (resp. Σ -re, Σ -pr) sii S es Γ -recursivo (resp. Γ -re, Γ -pr).

4.13.1. Lema 4.22 (Lema de cuantificación acotada)

Sea Σ un alfabeto finito.

1. Sea $P : S \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \omega$ un predicado Σ -pr, con $S, S_1, \dots, S_n \subseteq \omega$ y $L_1, \dots, L_m \subseteq \Sigma^*$ no vacíos. Sea $\overline{S} \subset S$ Σ -pr. Entonces estos predicados son Σ -pr:

$$\lambda x \vec{x} \vec{\alpha} \left[\forall (t \in \overline{S})_{t \leq x} P(t, \vec{x}, \vec{\alpha}) \right]$$

$$\lambda x \vec{x} \vec{\alpha} \left[\exists (t \in \overline{S})_{t \leq x} P(t, \vec{x}, \vec{\alpha}) \right]$$

2. Sea $P : S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \times L \rightarrow \omega$ un predicado Σ -pr, con $S_1, \dots, S_n \subseteq \omega$ y $L, L_1, \dots, L_m \subseteq \Sigma^*$ no vacíos. Sea $\overline{L} \subset L$ Σ -pr. Entonces estos predicados son Σ -pr:

$$\lambda x \vec{x} \vec{\alpha} \left[\forall (\alpha \in \overline{L})_{|\alpha| \leq x} P(\vec{x}, \vec{\alpha}, \alpha) \right]$$

$$\lambda x \vec{x} \vec{\alpha} \left[\exists (\alpha \in \overline{L})_{|\alpha| \leq x} P(\vec{x}, \vec{\alpha}, \alpha) \right]$$

4.13.2. Lema 4.30

Sean $P : S \subseteq \omega^n \times \Sigma^{*m} \rightarrow \omega$ y $Q : S \subseteq \omega^n \times \Sigma^{*m} \rightarrow \omega$ predicados Σ -recursivos.

Entonces también lo son: $(P \vee Q)$, $(P \wedge Q)$ y $\neg P$

4.13.3. Lema 4.25: Lema de minimización acotada

Sea $P : \text{Dom}_P \subseteq \omega \times \omega^n \times \Sigma^{*m} \rightarrow \{0, 1\}$ Σ -pr, entonces:

1. $M(P)$ es Σ -r
2. Si existe $f : \omega^n \times \Sigma^{*m} \rightarrow \omega$ Σ -pr tq $M(P)(\vec{x}, \vec{\alpha}) = \min_t P(t, \vec{x}, \vec{\alpha}) \leq f(\vec{x}, \vec{\alpha})$, entonces $M(P)$ es Σ -pr