Lenguajes Formales y Computabilidad | FAMAF - UNC

Combos de definiciones y convenciones notacionales y los Combos de teoremas

27.06.2025

Matias Viola

Contenido

1. Convenciones

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

1. $x, y, z, u, v, w, n, m, k, ... \in \omega$
2. $\alpha, \beta, \gamma, \delta, \varepsilon, \psi, \eta, ... \in \Sigma^*$
3. $O \in \{\omega, \Sigma^*\}$
4. «tq» es «tal que»
5. Σ-pr es «Σ-primitivo recursivo»
6. Sea $f : \mathrm{Dom}_f \to \{0, 1\}$, 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--recursiva.

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

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

2.1.3. «f es una función Σ-mixta»

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

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

2.1.4. «familia Σ-indexada de funciones»

Dado un alfabeto Σ, una familia Σ-indexada de funciones sera una función $\varrho : \Sigma \to \mathrm{Im}_G$ 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 $S_1, ..., S_n \subseteq \omega$ y $L_1, ..., L_m \subseteq \Sigma^*$ conjuntos no vacíos.

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

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

$$R(f,\varrho) : S_1 \times ... \times S_n \times L_1 \times ... \times L_m \times \Sigma^* \to \omega$$
$$(\vec{x}, \vec{\alpha}, \varepsilon) \to f(\vec{x}, \vec{\alpha})$$
$$(\vec{x}, \vec{\alpha}, \alpha a) \to \varrho_a(R(f,\varrho)(\vec{x}, \vec{\alpha}, \alpha), \vec{x}, \vec{\alpha}, \alpha)$$

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. $d \overset{n}{\vdash} d'$ y $d \overset{*}{\vdash} d'$

(no hace falta que defina $\vdash$)

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

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 $\alpha_1 ... \alpha_n \in \Sigma^*$ es aceptada por M por alcance de estado final si partiendo de $Bq_0\alpha_1 ... \alpha_n B...$ 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 $D_f \subseteq \omega^n \times \Sigma^{*m}$.
  - 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 $(s_1, s_2, ...) \in \omega^{[\mathbb{N}]}$ tq $x = \langle s_1, s_2, ... \rangle = \prod_{i=1}^{\infty} \text{pr}(i)^{s_i}$

2.2.5. $(x)_i$

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

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-recursivamente enumerable

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

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

2.3.2. $s^{\leq}$

Sea $\leq$ un orden sobre Σ^*.

$$S^{\leq} : \Sigma^* \to \Sigma^*$$
$$(a_n)^m \to (a_1)^{m+1}$$
$$\alpha a_i (a_n)^m \to \alpha a_{i+1} (a_1)^m \text{ con } 1 \leq i < n$$

2.3.3. $*^{\leq}$

Sea $\leq$ un orden sobre Σ^*.

$$*^{\leq} : \omega \to \Sigma^*$$
$$0 \to \varepsilon$$
$$i+1 \to s^{\leq}(*^{\leq}(i))$$

2.3.4. $\#^{\leq}$

Sea $\leq$ un orden sobre Σ^*.

$$\#^{\leq} : \Sigma^* \to \omega$$
$$\varepsilon \to 0$$
$$a_{i_k} ... a_{i_0} \to i_k n^k + ... + i_0 n^0$$

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

Sea O. Una función Σ-mixta $f : \text{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \to O$ 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 res_pecto 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 $F : \omega \to S$ tq $F_{(i)}$ sea Σ-efectivamente computable, para cada $i \in \{1, ..., n + m\}$.

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

Diremos que una función $f : \mathrm{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \to \Sigma^*$ 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 $(\vec{x}, \vec{\alpha}) \in \mathrm{Dom}_f$, entonces hay un $p \in Q$ tq $\lfloor q_0 B \nu^{x_1} B ... B \nu^{x_n} B \alpha_1 B ... B \alpha_m \rfloor \overset{*}{\vdash} \lfloor pBf(\vec{x}, \vec{\alpha}) \rfloor$ y $\lfloor pBf(\vec{x}, \vec{\alpha}) \rfloor \nvdash d$ para cada $d \in \mathrm{Des}$
2. Si $(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m} - \mathrm{Dom}_f$, entonces M no se detiene partiendo de $\lfloor q_0 B \nu^{x_1} B ... B \nu^{x_n} B \alpha_1 B ... B \alpha_m \rfloor$.

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 $\lfloor pBf(\vec{x}, \vec{\alpha}) \rfloor \nvdash d$ para cada $d \in \mathrm{Des}$ 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 $P : \mathrm{Dom}_P \subseteq \omega^n \times \Sigma^{*m}$. 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

$\mathrm{Lt} : \mathbb{N} \to \omega$

$1 \to 0$

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

2.8.3. Conjunto rectangular

Sea Σ un alfabeto finito. Un conjunto Σ-mixto es llamado rectangular si es de la forma $S_1 \times ... \times S_n \times L_1 \times ... \times L_m$ 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 $S_1 \times ... \times S_n \times L_1 \times ... \times L_m$ 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 $\alpha \in \{L\overline{n} : n \in \mathbb{N}\}$ 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 $I_1 I_2 ... I_n$ donde $n \geq 1, I_1, ..., I_n \in \mathrm{Ins}^\Sigma$ y se cumple la ley de los GOTO.

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

2.9.4. I_i^P

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

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

2.9.5. $n(P)$

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

$$P \to m \text{ tq } P = I_1 I_2 ... I_m$$

2.9.6. Bas

$\mathrm{Bas} : \mathrm{Ins}^{\Sigma} \to \left(\Sigma \cup \Sigma_p\right)^*$

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

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

2.10.1. «estado»

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

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

2.10.2. «descripción instantánea»

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

Si $i \in \{1, ..., n(P)\}$, $(i, \vec{x}, \vec{\sigma})$ 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.

$S_P : \mathrm{Des}^{\Sigma} \to \mathrm{Des}^{\Sigma}$

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

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

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 $(\vec{x},\vec{\alpha})$»

Dado $\overbrace{S_P(...S_P(S_P}^{\text{t veces}}(1,\vec{x},\vec{\sigma}))...) = (j,\vec{u},\vec{\eta})$, 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. $\Psi_P^{\mathrm{n,m,\#}}$

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

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

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

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

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

Dado $s, O \in \{(\#,\omega),(*,\Sigma^*)\}$. Una función Σ-mixta $f : S \subseteq \omega^n \times \Sigma^{*m} \to O$ 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 $P : \mathrm{Dom}_P \subseteq \omega^n \times \Sigma^{*m} \times \Sigma^*$ 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 $F : \omega \to S$ tq $F_{(i)}$ sea Σ-computable, para cada $i \in \{1, ..., n+m\}$.

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

$$S = \mathrm{Im}\Big[\Psi^{1,0,\#}_{P_1}, ..., \Psi^{1,0,\#}_{P_n}, \Psi^{1,0,*}_{P_{n+1}}, ..., \Psi^{1,0,*}_{P_{n+m}}\Big]$$

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

2.13. Combo 13

Defina:

2.13.1. $i^{n,m}$

$$i^{n,m} : \omega \times \omega^n \times \Sigma^{*m} \times \mathrm{Pro}^{\Sigma} \to \omega$$
$$(0, \vec{x}, \vec{\alpha}, P) \to 1$$
$$(t, \vec{x}, \vec{\alpha}, P) \to j \text{ tq } \overbrace{S_P(...S_P(S_P}^{\text{t veces}}(1, \vec{x}, \vec{\sigma}))...) = (j, \vec{u}, \vec{\eta})$$

2.13.2. $E^{n,m}_{\#}$

$$E^{n,m}_{\#} : \omega \times \omega^n \times \Sigma^{*m} \times \mathrm{Pro}^{\Sigma} \to \omega^{[\mathbb{N}]}$$
$$(0, \vec{x}, \vec{\alpha}, P) \to (x_1, ..., x_n, 0, ...)$$
$$(t, \vec{x}, \vec{\alpha}, P) \to \vec{u} \text{ tq } \overbrace{S_P(...S_P(S_P}^{\text{t veces}}(1, \vec{x}, \vec{\sigma}))...) = (j, \vec{u}, \vec{\eta})$$

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

$$E^{n,m}_{*} : \omega \times \omega^n \times \Sigma^{*m} \times \mathrm{Pro}^{\Sigma} \to \Sigma^{*[\mathbb{N}]}$$
$$(0, \vec{x}, \vec{\alpha}, P) \to (\alpha_1, ..., \alpha_n, \varepsilon, ...)$$
$$(t, \vec{x}, \vec{\alpha}, P) \to \vec{\eta} \text{ tq } \overbrace{S_P(...S_P(S_P}^{\text{t veces}}(1, \vec{x}, \vec{\sigma}))...) = (j, \vec{u}, \vec{\eta})$$

2.13.4. $E^{n,m}_{\#_j}$

$$E^{n,m}_{\#_j} : \omega \times \omega^n \times \Sigma^{*m} \times \mathrm{Pro}^{\Sigma} \to \omega$$
$$E^{n,m}_{\#_j} = p^{n,m}_j \circ E^{n,m}_{\#}$$

2.13.5. $E^{n,m}_{*_j}$

$$E^{n,m}_{*_j} : \omega \times \omega^n \times \Sigma^{*m} \times \mathrm{Pro}^{\Sigma} \to \Sigma^{*}$$
$$E^{n,m}_{*_j} = p^{n,m}_j \circ E^{n,m}_{*}$$

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

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

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

2.13.7. $T^{n,m}$

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

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

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

2.13.8. AutoHalt^{Σ}

Dado $\Sigma \supseteq \Sigma_p$

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

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

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 $x, y, z, u, v, w, n, m, k, ...$
   - Se usan **variables alfabéticas** que se valúan en palabras sobre el alfabeto Σ. Se denotan por letras como $\alpha, \beta, \gamma, \delta, \varepsilon, \psi, \eta, ...$
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 $f : \mathrm{Dom}_f \subseteq \omega \times \Sigma^* \to \omega$, describa qué tipo de objeto es y qué propiedades debe tener el macro: [V2 ⟵ f(V1,W1)]

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

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

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 $V\overline{k'}$ por variables concretas $N\overline{k}$ con k distintos entre si.
   2. Las variables $W\overline{j'}$ por variables concretas $P\overline{j}$ con j distintos entre si.
   3. Los labels $A\overline{z'}$ por labels concretos $L\overline{z}$ con z distintos entre si.

   Obtenemos la palabra $N\overline{k_2} \leftarrow f\left(N\overline{k_1}, P\overline{j_1}\right)$ 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 $N\overline{k_1}, N\overline{k_2}, P\overline{j_1}$ valores x_1, x_2, α_1, se dará que
   1. Si $(x_1, \alpha_1) \notin \mathrm{Dom}_P$, el programa no se detiene.
   2. Si $(x_1, \alpha_1) \in \mathrm{Dom}_P$, luego de una cantidad finita de pasos el programa se detiene llegando a un estado e' tq:
      1. e' asigna a $N\overline{k_2}$ el valor $f(x_1, \alpha_1)$;
      2. e' solo difiere de e en el valor de $N\overline{k_2}$ y en las variables que reemplazaron a las auxiliares de M.

La palabra $N\overline{k_2} \leftarrow f\left(N\overline{k_1}, P\overline{j_1}\right)$ 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 $p : D_f \subseteq \omega \times \Sigma^* \to \omega$, describa qué tipo de objeto es y qué propiedades debe tener el macro: [IF P(V1,W1) GOTO A1]

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

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

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 $V\overline{k'}$ por variables concretas $N\overline{k}$ con k distintos entre si.
   2. Las variables $W\overline{j'}$ por variables concretas $P\overline{j}$ con j distintos entre si.
   3. Los labels $A\overline{z'}$ por labels concretos $L\overline{z}$ con z distintos entre si.

   Obtenemos la palabra $\left[\text{IF } P\left(N\overline{k_1}, P\overline{j_1}\right) \text{ GOTO } L\overline{z_1}\right]$ la cual, si se cumple la ley del GOTO respecto a $L\overline{z_1}$, 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 $N\overline{k_1}, P\overline{j_1}$ 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 $\left[\text{IF } P\left(N\overline{k_1}, P\overline{j_1}\right) \text{ GOTO } L\overline{z_1}\right]$ 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 $f : S \subseteq A \to B$ para expresar que f es una función tq $\text{Dom}_f = S \subseteq A$ y $\text{Im}_f \subseteq B$. También escribimos $f : A \to B$ 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 $f = \left\{(x, f(x)) : x \in \text{Dom}_f\right\}$.

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

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 ⇒

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 ⇐

Sea S el dominio de una función Σ-pr $f : \mathrm{Dom}_f \subseteq \omega^n \times \Sigma^{*m} \to O$.
Probaremos por inducción en k que Dom_F es Σ-pr, para cada $F \in \mathrm{PR}_k^\Sigma$:

1. Caso $k = 0$: $\mathrm{PR}_0^\Sigma = \left\{\mathrm{suc}, \mathrm{pred}, C_0^{0,0}, C_\varepsilon^{0,0}\right\} \cup \{d_a : a \in \Sigma\} \cup \left\{p_j^{n,m} : 1 \leq j \leq n+m\right\}$
   Los dominios de las funciones $\mathrm{suc}, C_0^{0,0}, C_\varepsilon^{0,0}, d_a, p_j^{n,m}$ son de la forma $\omega^n \times \Sigma^{*m}$ y ω y Σ^* son Σ-pr, por el «Lema 16» son Σ-pr.
   Finalmente, $\chi^\omega_{\mathrm{Dom}_{\mathrm{Pred}}} = \lambda x[x \neq 0]$ es Σ-pr, por definición $\mathrm{Dom}_{\mathrm{Pred}}$ es Σ-pr
2. Supongamos que Dom_F es Σ-pr $\forall F \in \mathrm{PR}_k^\Sigma$.
3. Sea $F \in \mathrm{PR}_{k+1}^\Sigma$. Veremos entonces que Dom_F es Σ-pr solo para el caso de composición:
   Si $F = \emptyset$, entonces es claro que $\mathrm{Dom}_F = \emptyset$ es Σ-pr.
   Sea $F = g \circ [g_1, ..., g_{n+m}]$ no vacío, con $g, g_1, ..., g_{n+m} \in \mathrm{PR}_k^\Sigma$.
   - $g : \mathrm{Dom}_g \subseteq \omega^n \times \Sigma^{*m} \to O$
   - $g_i : \mathrm{Dom}_{g_i} \subseteq \omega^k \times \Sigma^{*l} \to \omega$ para $i = 1, ..., n$
   - $g_i : \mathrm{Dom}_{g_i} \subseteq \omega^k \times \Sigma^{*l} \to \Sigma^*$ para $i = n+1, ..., n+m$

   Por hipótesis inductiva, los conjuntos $\mathrm{Dom}_g, \mathrm{Dom}_{g_i}$ son Σ-pr.
   Por «Lema 15», $S = \bigcap_{i=1}^{n+m} \mathrm{Dom}_{g_i}$ es Σ-pr.
   Por «Lema 20», $\chi^{\omega^k \times \Sigma^{*l}}_{\mathrm{Dom}_F}(\vec{x}, \vec{\alpha}) = \begin{cases} \chi^{\omega^n \times \Sigma^{*m}}_{\mathrm{Dom}_g} \circ [g_1, ..., g_{n+m}] & \text{si } (\vec{x}, \vec{\alpha}) \in S \\ C_0^{k,l} & \text{si } (\vec{x}, \vec{\alpha}) \in \omega^k \times \Sigma^{*l} - S \end{cases}$ es Σ-pr.
   Por lo tanto Dom_F es Σ-pr

3.1.2. Teorema: Neumann vence a Gödel

Si h es Σ-recursiva, 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: $\mathrm{R}_0^\Sigma = \mathrm{PR}_0^\Sigma = \left\{\mathrm{suc}, \mathrm{pred}, C_0^{0,0}, C_\varepsilon^{0,0}\right\} \cup \{d_a : a \in \Sigma\} \cup \left\{p_j^{n,m} : 1 \leq j \leq n+m\right\}$ 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 $\mathrm{Im}_h \subseteq \omega$.
   Sean
   - $\Sigma = \{a_1, ..., a_r\}$
   - $\left(f \in R_k^\Sigma\right) f : S_1 \times ... \times S_n \times L_1 \times ... \times L_m \to \omega$
   - $\left(\forall a \in \Sigma, \varrho_a \in R_k^\Sigma\right) \varrho_a : \omega \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m \times \Sigma^* \to \omega$

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

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

Entonces, construimos el siguiente programa usando macros:

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

Este programa computa h.

3.2. Combo 2

3.2.1. Lema 20: Lema de división por casos para funciones Σ-pr

Si $f_i : \mathrm{Dom}_{f_i} \subseteq \omega^n \times \Sigma^{*m} \to O$ para $i = 1, ..., k$ son Σ-pr tq si $i \neq j \Rightarrow \mathrm{Dom}_{f_i} \cap \mathrm{Dom}_{f_j} = \emptyset$, 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 $f_i : \mathrm{Dom}_{f_i} \subseteq \omega^2 \times \Sigma^{*2} \to \Sigma^*$ Σ-pr tq si $i \neq j \Rightarrow \mathrm{Dom}_{f_i} \cap \mathrm{Dom}_{f_j} = \emptyset$.

Por «Lema 18», existen funciones Σ-totales Σ-pr $\overline{f}_i : \omega^2 \times \Sigma^{*2} \to \Sigma^*$ tq $f_i = \overline{f}_i|_{\mathrm{Dom}_{f_i}}$.
Por «Proposición 19», los conjuntos Dom_{f_1} y Dom_{f_2} son Σ-pr.
Por lo tanto, por «Lema 15», también lo es su unión: $\mathrm{Dom}_{f_1} \cup \mathrm{Dom}_{f_2}$.
Finalmente, por «Lema 17»,

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

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 $P \in \mathrm{Pro}^{\Sigma}$ tq:

1. Para cada $x \in \omega$, P se detiene partiendo desde el estado $[\![x]\!]$ y llega a un estado de la forma $((x_1, ..., x_n, y_1, ...), (\alpha_1, ..., \alpha_m, \beta_1, ...))$, donde $(x_1, ..., x_n, \alpha_1, ..., \alpha_m) \in S$
2. Para cada $(x_1, ..., x_n, \alpha_1, ..., \alpha_m) \in S$ hay un $x \in \omega$ tq P se detiene partiendo desde el estado $[\![x]\!]$ y llega a un estado como en $((x_1, ..., x_n, y_1, ...), (\alpha_1, ..., \alpha_m, \beta_1, ...))$

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

3.3. Combo 3

1. **Teorema** (Gödel vence a Neumann): Si $f : D_f \subseteq \omega^n \times {\Sigma^*}^m \to \Sigma^*$ es Σ-computable, entonces f es Σ-recursiva

2. **Teorema** (Caracterización de conjuntos Σ-efectivamente computables): Sea $S \subseteq \omega^n \times {\Sigma^*}^m$. Son equivalentes:

(a) S es Σ-efectivamente computable (b) S y $(\omega^n \times {\Sigma^*}^m) - S$ son Σ-efectivamente enumerables (Hacer solo $(b) \to (a)$)

3.4. Combo 4

1. **Proposición** (Caracterización básica de conjuntos Σ-enumerables): (igual a Combo 2, hacer caso $n = 2$, $m = 1$)

2. **Lema** (Lema de la sumatoria): Sea Σ un alfabeto finito. Si $f : \omega \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m \to \omega$ es Σ-pr, con $S_i \subseteq \omega$ y $L_j \subseteq \Sigma^*$ no vacíos, entonces

$\lambda x y \vec{x} \vec{\alpha}. \sum_t = x^y f(t, \vec{x}, \vec{\alpha})$ es Σ-pr

3.5. Combo 5

1. **Lema**: Sea $\Sigma = @, \%, !$ y $f : S_1 \times S_2 \times L_1 \times L_2 \to \omega$, con $S_1, S_2 \subseteq \omega$ y $L_1, L_2 \subseteq \Sigma^*$ no vacíos. Sea ϱ una familia Σ-indexada de funciones $\varrho_a : \omega \times S_1 \times S_2 \times L_1 \times L_2 \times \Sigma^* \to \omega$ para cada $a \in \Sigma$.

Si f y cada ϱ_a son Σ-efectivamente computables, entonces $R(f, \varrho)$ lo es. (Ejercicio de la Guía 5)

1. **Lema** (Lema de cuantificación acotada): Sea $p : S \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m \to \omega$ un predicado Σ-pr, y $\bar{S} \subseteq S$ un conjunto 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

3.6. Combo 6

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

2. **Teorema** (Caracterización de conjuntos Σ-r.e.): Sea $S \subseteq \omega^n \times {\Sigma^*}^m$. Son equivalentes:

(1) S es Σ-recursivamente enumerable (2) $S = \text{IF}$, para alguna $F : D_F \subseteq \omega^k \times {\Sigma^*}^l \to \omega^n \times {\Sigma^*}^m$ tq cada $F(i)$ es Σ-recursiva (3) $S = D_f$, para alguna función Σ-recursiva f (Hacer la prueba de $(2) \to (3)$, con $k = l = 1$ y $n = m = 2$)

3.7. Combo 7

1. **Lema** (Lema de minimización acotada): Sean $n, m \geq 0$. Sea $p : D_\nu \subseteq \omega \times \omega^n \times \Sigma^{*m} \to \omega$ un predicado Σ-pr

(a) $M(P)$ es Σ-recursiva (b) Si existe una función $f : \omega^n \times \Sigma^{*m} \to \omega$ 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

1. **Lema**: Si $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to O$ es Σ-recursiva y $S \subseteq D_f$ es Σ-r.e., entonces $f|S$ es Σ--recursiva

(Hacer solo el caso S no vacío, $n = m = 1$ y $O = \Sigma^*$)

3.8. Combo 8

1. **Lema**: Si $\Sigma \supseteq \Sigma_p$, entonces AutoHalt^Σ no es Σ-recursivo
2. **Teorema**: Si $\Sigma \supseteq \Sigma_p$, entonces AutoHalt^Σ no es Σ-efectivamente computable
3. **Lema**: Sea $A = p \in \text{Pro}^\Sigma : \text{AutoHalt}^{\Sigma(P)} = 1$, entonces A es Σ-r.e. y no Σ-recursivo

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

1. **Teorema** (Neumann vence a Gödel): Si h es Σ-recursiva, entonces h es Σ-computable

(Hacer solo el caso $h = M(P)$)

3.9. Combo 9

1. **Lema** (Lema de división por casos para funciones Σ-recursivas): Supongamos $f_i : D_{f_i} \subseteq \omega^n \times \Sigma^{*m} \to O$ para $i = 1, ..., k$, tales que $D_{f_i} \Rightarrow D_{f_j} = \emptyset$ para $i \neq j$. Entonces $f_1 \vDash ... \vDash f_k$ es Σ-recursiva

(Hacer el caso $k = 2$, $n = m = 1$ y $O = \omega$)

1. **Teorema** (Gödel vence a Neumann): Si $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to \omega$ es Σ-computable, entonces f es Σ-recursiva

4. Utilidades

4.1. Lema 14.

Sean $P : S \subseteq \omega^n \times \Sigma^{*m} \to \{0, 1\}$ y $Q : S \subseteq \omega^n \times \Sigma^{*m} \to \{0, 1\}$ 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 15.

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 16.

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

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

4.5. Lema 17.

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

4.6. Lema 18.

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

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 20: Lema de division por casos para funciones Σ-pr

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

4.9. Lema 22.

Sea Sigma un alfabeto finito.

(a) SI $f : \omega \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m \to \omega$ es Σ-pr, con $S_1, ..., S_n \subseteq \omega$ y $L_1, ..., L_m \subseteq \Sigma^*$ 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 $f : \omega \times S_1 \times ... \times S_n \times L_1 \times ... \times L_m \to \Sigma^*$ es Σ-pr, con $S_1, ..., S_n \subseteq \omega$ y $L_1, ..., L_m \subseteq \Sigma^*$ no vacíos, entonces la función $\lambda xy\vec{x}\vec{\alpha}. \underset{t=x}{\overset{y}{\subset}} f(t, \vec{x}, \vec{\alpha})$ es Σ-pr