Lenguajes Formales y Computabilidad
Teoremas: Combo 6

Nicolás Cagliero

July 1, 2025

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

Proof. Supongamos $S \neq \emptyset$. Sea $(\vec{z}, \vec{\gamma}) \in S$ fijo. Sea $\mathbb{P}$ un procedimiento efectivo que computa a $\chi_S^{\omega^n \times \Sigma^{*m}}$. Sea $\mathbb{P}_1$ un procedimiento efectivo que enumera a $\omega^n \times \Sigma^{*m}$. Entonces el siguiente procedimiento enumera a S:

Etapa 1: Realizar $\mathbb{P}_1$ con $x \in \omega$ de entrada para obtener $(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m}$.
Etapa 2: Realizar $\mathbb{P}$ con $(\vec{x}, \vec{\alpha})$ de entrada para obtener el valor Booleano e de salida.
Etapa 3: Si $e = 1$ dar como dato de salida $(\vec{x}, \vec{\alpha})$. Si $e = 0$ dar como dato de salida $(\vec{z}, \vec{\gamma})$.

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

1. S es Σ-recursivamente enumerable.
2. $S = I_F$, para alguna $F : D_F \subseteq \omega^k \times \Sigma^{*l} \to \omega^n \times \Sigma^{*m}$ tal que cada $F_{(i)}$ es Σ-recursiva.
3. $S = D_f$, para alguna función Σ-recursiva f.

Proof. (2) $\Rightarrow$ (3). Haremos el caso $k = l = 1$ y $n = m = 2$. El caso general es completamente análogo. Nótese que entonces tenemos que

$$S \subseteq \omega^2 \times \Sigma^{*2} \text{ y } F : D_F \subseteq \omega \times \Sigma^* \to \omega^2 \times \Sigma^{*2} \text{ es tal que } I_F = S \text{ y } F_{(1)}, F_{(2)}, F_{(3)}, F_{(4)} \text{ son } \Sigma\text{-recursivas.}$$

Para cada $i \in \{1, 2, 3, 4\}$, sea $\mathcal{P}_i$ un programa el cual computa a $F_{(i)}$. Sea $\leq$ un orden total sobre Σ. Definamos

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

Notar que $D_{H_i} = \omega^2 \times \Sigma^*$ y que H_i es Σ-mixta. Además sabemos que la función $Halt^{1,1}$ es $(\Sigma \cup \Sigma_p)$-p.r. por lo cual resulta fácilmente que H_i es $(\Sigma \cup \Sigma_p)$-p.r.

Por la Proposición de Independencia del Alfabeto tenemos que H_i es Σ-p.r., lo cual por el Segundo Manantial nos dice que hay un macro:

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

Para hacer más intuitivo el uso de este macro lo escribiremos de la siguiente manera:

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

Para $i = 1, 2$, definamos

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

Para $i = 3, 4$, definamos

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

Con el mismo análisis que hicimos para H_i llegamos a que cada E_i es Σ-pr. O sea que para cada $i \in \{1, 2\}$ hay un macro

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

y para cada $i \in \{3, 4\}$ hay un macro

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

Haremos más intuitiva la forma de escribir estos macros, por ejemplo para $i = 1$, lo escribiremos de la siguiente manera

$$[\text{IF V2 } \neq E^{1,1}_{\#1}(\text{V3}, \text{V1}, \text{W1}, \mathcal{P}_1) \text{ GOTO A1}]$$

Ya que la función $f = \lambda x[(x)_1]$ es Σ-p.r. hay un macro

$$[\text{V2} \leftarrow f(\text{V1})]$$

el cual escribiremos de la siguiente manera:

$$[\text{V2} \leftarrow (\text{V1})_1]$$

Similarmente hay macros:

$$[\text{W1} \leftarrow *^{\leq}(\text{V1})_3]$$

$$[\text{V2} \leftarrow (\text{V1})_2]$$

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

L1 $\text{N20} \leftarrow \text{N20} + 1$

$[\text{N10} \leftarrow (\text{N20})_1]$

$[\text{N3} \leftarrow (\text{N20})_2]$

$[\text{P3} \leftarrow *^{\leq}(\text{N20})_3]$

$[\text{IF } \neg Halt^{1,1}(\text{N10}, \text{N3}, \text{P3}, \mathcal{P}_1) \text{ GOTO L1}]$

$[\text{IF } \neg Halt^{1,1}(\text{N10}, \text{N3}, \text{P3}, \mathcal{P}_2) \text{ GOTO L1}]$

$[\text{IF } \neg Halt^{1,1}(\text{N10}, \text{N3}, \text{P3}, \mathcal{P}_3) \text{ GOTO L1}]$

$[\text{IF } \neg Halt^{1,1}(\text{N10}, \text{N3}, \text{P3}, \mathcal{P}_4) \text{ GOTO L1}]$

$[\text{IF N1} \neq E^{1,1}_{\#1}(\text{N10}, \text{N3}, \text{P3}, \mathcal{P}_1) \text{ GOTO L1}]$

$[\text{IF N2} \neq E^{1,1}_{\#1}(\text{N10}, \text{N3}, \text{P3}, \mathcal{P}_2) \text{ GOTO L1}]$

$[\text{IF P1} \neq E^{1,1}_{*1}(\text{N10}, \text{N3}, \text{P3}, \mathcal{P}_3) \text{ GOTO L1}]$

$[\text{IF P2} \neq E^{1,1}_{*1}(\text{N10}, \text{N3}, \text{P3}, \mathcal{P}_4) \text{ GOTO L1}]$

Es claro que este programa computa una proyección, no modifica ni N1 ni P1 así que computa tanto a $p^{2,2}_1$ como a $p^{2,2}_3$. Además, podemos notar que el programa no termina si los valores representados en N1, N2, P1 y P2 no pertenecen a S pues nunca van a cumplirse las 4 comparaciones pues las $F_{(i)}$ no enumeran a esos valores. Entonces, el programa computa $p^{2,2}_1|_S$. Entonces $p^{2,2}_1|_S$ es Σ-computable por lo cual es Σ-recursiva, lo cual prueba (3) ya que $D_{(p^{2,2}_1|_S)} = S$.