

Guía 5: Paradigma de Godel

Idea

- La idea es partir de un conjunto inicial de funciones muy simples y obviamente Σ -efectivamente computables, y luego obtener nuevas funciones Σ -efectivamente computables usando constructores que preservan la computabilidad efectiva
- Las funciones Σ -recursivas son las que se obtienen iterando el uso de estos constructores, partiendo del conjunto inicial $\{Suc, Pred, C_0^{0,0}, C_\varepsilon^{0,0}\} \cup \{d_a : a \in \Sigma\} \cup \{p_j^{n,m} : 1 \leq j \leq n+m\}$
- Los constructores que se usarán (y conservan computabilidad efectiva) serán:
 - *Composición*
 - *Recursión primitiva*
 - *Minimización de predicados totales*
- Una función es Σ -recursiva primitiva si se obtiene a partir del conjunto inicial usando solo composición y recursión primitiva

Composición

- *Definición:*
 - Dadas funciones Σ -mixtas f, f_1, \dots, f_r con $r \geq 1$, diremos que $f \circ [f_1, \dots, f_r]$ es obtenida por composición a partir de las funciones f, f_1, \dots, f_r
- *Lemas:*
 - Σ -mixta: Sean f, f_1, \dots, f_r funciones Σ -mixtas con $r \geq 1$ y $f \circ [f_1, \dots, f_r] \neq \emptyset$, entonces $\exists n, m, k, l \in \omega, s \in \{\#, *\}$ tales que:
 - $r = n + m$
 - f es de tipo (n, m, s)
 - f_i es de tipo $(k, l, \#)$ para $1 \leq i \leq n$
 - f_i es de tipo $(k, l, *)$ para $n + 1 \leq i \leq r$Y además, $f \circ [f_1, \dots, f_r]$ es Σ -mixta de tipo (k, l, s) con:
 - $D_{f \circ [f_1, \dots, f_r]} = \{(\vec{x}, \vec{a}) \in \cap_{i=1}^r D_{f_i} : (f_1(\vec{x}, \vec{a}), \dots, f_r(\vec{x}, \vec{a})) \in D_f\}$
 - $f \circ [f_1, \dots, f_r](\vec{x}, \vec{a}) = f(f_1(\vec{x}, \vec{a}), \dots, f_r(\vec{x}, \vec{a}))$
 - *Conservación de computabilidad efectiva:* Si f, f_1, \dots, f_r son Σ -efectivamente computables, entonces $f \circ [f_1, \dots, f_r]$ también lo es

Recursión primitiva

Conjuntos rectangulares

- *Definición:*

- Sea Σ un alfabeto finito, un conjunto Σ -mixto S es llamado *rectangular* si es de la forma $S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m$ con $S_i \subseteq \omega$ y $L_j \subseteq \Sigma^*$.
- *Lemas:*
 - Sea $S \subseteq \omega \times \Sigma^*$, entonces S es *rectangular* \Leftrightarrow si $(x, \alpha), (y, \beta) \in S$, entonces $(x, \beta) \in S$

Variable numérica - Valores numéricos

- *Definición:* sean f, g funciones dadas por:

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

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

con $S_i \subseteq \omega$ y $L_i \subseteq \Sigma^*$ conjuntos no vacíos, entonces existe una única función $R : \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \omega$ tal que:

$$R(0, \vec{x}, \vec{\alpha}) = f(\vec{x}, \vec{\alpha})$$

$$R(t+1, \vec{x}, \vec{\alpha}) = g(R(t, \vec{x}, \vec{\alpha}), t, \vec{x}, \vec{\alpha})$$

- Llamaremos $R(f, g)$ a esta función y diremos que $R(f, g)$ es obtenida por recursión primitiva a partir de f y g
- *Lemas:*
 - *Conservación de computabilidad efectiva:* Si f, g son Σ -efectivamente computables, entonces $R(f, g)$ también lo es

Variable numérica - Valores alfabéticos

- *Definición:* sean f, g funciones dadas por:

$$f : S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \Sigma^*$$

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

con $S_i \subseteq \omega$ y $L_i \subseteq \Sigma^*$ conjuntos no vacíos, entonces existe una única función $R(f, g) : \omega \times S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \Sigma^*$ tal que:

$$R(f, g)(0, \vec{x}, \vec{\alpha}) = f(\vec{x}, \vec{\alpha})$$

$$R(f, g)(t+1, \vec{x}, \vec{\alpha}) = g(t, \vec{x}, \vec{\alpha}, R(f, g)(t, \vec{x}, \vec{\alpha}))$$

- Llamaremos $R(f, g)$ a esta función y diremos que $R(f, g)$ es obtenida por recursión primitiva a partir de f y g
- *Lemas:*
 - *Conservación de computabilidad efectiva:* Si f, g son Σ -efectivamente computables, entonces $R(f, g)$ también lo es

Variable alfabética - Valores numéricos

- *Definiciones:*

- *Familia Σ -indexada de funciones:* Dado un alfabeto Σ , una familia Σ -indexada de funciones es una función \mathcal{G} tal que $D_{\mathcal{G}} = \Sigma$ y $\forall a \in D_{\mathcal{G}}, \mathcal{G}(a)$ es una función
 - Si \mathcal{G} es una familia Σ -indexada de funciones, entonces para $a \in \Sigma$ escribiremos \mathcal{G}_a en lugar de $\mathcal{G}(a)$
- *Recursión primitiva:* Sea Σ un alfabeto finito, y sean f una función y \mathcal{G} una familia Σ -indexada de funciones tales que:

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

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

para cada $a \in \Sigma$, y con $S_i \subseteq \omega$ y $L_i \subseteq \Sigma^*$ conjuntos no vacíos, entonces definimos

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

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

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

- Diremos que $R(f, \mathcal{G})$ es obtenida por recursión primitiva a partir de f y \mathcal{G}
- *Lemas:*
 - *Conservación de computabilidad efectiva:* Si f y cada \mathcal{G}_a son Σ -efectivamente computables, entonces $R(f, \mathcal{G})$ también lo es

Variable alfabética - Valores alfabéticos

- *Definición:*

- Sea Σ un alfabeto finito, y sean f una función y \mathcal{G} una familia Σ -indexada de funciones tales que:

$$f : S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \rightarrow \Sigma^*$$

$$\mathcal{G}_a : S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \times \Sigma^* \times \Sigma^* \rightarrow \Sigma^*$$

para cada $a \in \Sigma$, y con $S_i \subseteq \omega$ y $L_i \subseteq \Sigma^*$ conjuntos no vacíos, entonces definimos

$$R(f, \mathcal{G}) : S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m \times \Sigma^* \rightarrow \Sigma^*$$

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

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

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

- *Conservación de computabilidad efectiva:* Si f y cada \mathcal{G}_a son Σ -efectivamente computables, entonces $R(f, \mathcal{G})$ también lo es

Funciones Σ -recursivas primitivas

- *Definición:* Dados los conjuntos $PR_0^\Sigma \subseteq PR_1^\Sigma \subseteq \dots \subseteq PR^\Sigma$ definidos como

$$PR_0^\Sigma = \{Suc, Pred, C_0^{0,0}, C_\varepsilon^{0,0}\} \cup \{d_a : a \in \Sigma\} \cup \{p_j^{n,m} : 1 \leq j \leq n+m\}$$

$$PR_{k+1}^\Sigma = PR_k^\Sigma \cup \{f \circ [f_1, \dots, f_r] : f, f_1, \dots, f_r \in PR_k^\Sigma, r \geq 1\} \\ \cup \{R(f, \mathcal{G}) : f, \mathcal{G}_a \in PR_k^\Sigma \forall a \in \Sigma\} \cup \{R(f, g) : f, g \in PR_k^\Sigma\}$$

$$PR^\Sigma = \bigcup_{k \in \omega} PR_k^\Sigma$$

Diremos que una función es llamada Σ -recursiva primitiva (Σ -p.r.) si pertenece a PR^Σ

- *Proposiciones:*
 - *Computabilidad efectiva:* Si $f \in PR^\Sigma$, entonces f es Σ -efectivamente computable
- *Lemas:*
 - *Ejemplos de funciones Σ -p.r.:*
 - *Operaciones numéricas:*
 - $\lambda xy[x + y] \in PR^\Sigma$
 - $\lambda xy[x \cdot y] \in PR^\Sigma$
 - $\lambda x[x!] \in PR^\Sigma$
 - $\lambda xy[x^y] \in PR^\Sigma$
 - $\lambda xy[x \dot{-} y] \in PR^\Sigma$
 - $\lambda xy[max(x, y)] \in PR^\Sigma$
 - $\lambda xy[x = y] \in PR^\Sigma$
 - $\lambda xy[x \leq y] \in PR^\Sigma$
 - *Operaciones de palabras*
 - $\lambda \alpha \beta[\alpha \beta] \in PR^\Sigma$
 - $\lambda \alpha[|\alpha|] \in PR^\Sigma$
 - $\lambda t \alpha[\alpha^t] \in PR^\Sigma$
 - $\lambda \alpha \beta[\alpha = \beta] \in PR^\Sigma$
 - *Otras:*
 - $\emptyset \in PR^\Sigma$
 - $\forall n, m, k \in \omega, \alpha \in \Sigma^*; C_k^{n,m}, C_\alpha^{n,m} \in PR^\Sigma$
 - Si \leq es un orden total sobre Σ , entonces $s^\leq, \#^\leq, *^\leq \in PR^\Sigma$
 - *Operaciones lógicas entre predicados:* Si P, Q son predicados Σ -p.r. con igual dominio, entonces $P \wedge Q, P \vee Q, \neg P$ son Σ -p.r. también.

Conjuntos Σ -recursivos primitivos

- **Definición:**
 - Un conjunto Σ -mixto $S \subseteq \omega^n \times \Sigma^{*m}$ es llamado Σ -recursivo primitivo si su función característica $\chi_S^{\omega^n \times \Sigma^{*m}}$ es Σ -p.r.
 - *Notar que* $\chi_S^{\omega^n \times \Sigma^{*m}} = \lambda \vec{x} \vec{\alpha} [(\vec{x}, \vec{\alpha}) \in S]$
- **Lemas:**
 - Si $S_1, S_2 \subseteq \omega^n \times \Sigma^{*m}$ son Σ -p.r., entonces $S_1 \cup S_2, S_1 \cap S_2, S_1 - S_2$ son Σ -p.r. también
 - Si $S \subseteq \omega^n \times \Sigma^{*m}$ es *finito*, entonces S es Σ -p.r.
 - **Conjunto rectangular Σ -p.r.:** Sean $S_1, \dots, S_n \in \omega, L_1, \dots, L_m \in \Sigma^*$ conjuntos no vacíos, entonces $S_1 \times \dots \times S_n \times L_1 \times \dots \times L_m$ es Σ -p.r. $\Leftrightarrow S_1, \dots, S_n, L_1, \dots, L_m$ son Σ -p.r.
 - **Σ -p.r. para función restringida:** Sea $f : D_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$ que es Σ -p.r. (con $O \in \{\omega, \Sigma^*\}$), si $S \subseteq D_f$ es Σ -p.r., entonces $f|_S$ es Σ -p.r. también
 - **Definición:** Dada una función f y un conjunto $S \subseteq D_f$, usaremos $f|_S$ para denotar la restricción de f al conjunto S , es decir, $f|_S = f \cap (S \times I_f)$. Notar que $D_{f|_S} = S$ y $f|_S(e) = f(e) \forall e \in S$.
 - Sean $O \in \{\omega, \Sigma^*\}$ y $n, m \in \omega$, si $f : D_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$ es Σ -p.r., entonces existe una función Σ -p.r. $\bar{f} : \omega^n \times \Sigma^{*m} \rightarrow O$ tal que $f = \bar{f}|_{D_f}$
- **Proposiciones:**
 - **Relación Conjunto/Función Σ -p.r.:** Un conjunto S es Σ -p.r. si y solo si S es el dominio de alguna función Σ -p.r.

Lema de división por casos para funciones Σ -p.r.

- **Observación:** Si $f_i : D_{f_i} \rightarrow O \forall i = 1, \dots, k$ son funciones tales que $\forall i \neq j, D_{f_i} \cap D_{f_j} = \emptyset$, entonces $f_1 \cup \dots \cup f_k$ es la función dada por

$$D_{f_1} \cup \dots \cup D_{f_k} \rightarrow O$$

$$e \rightarrow \begin{cases} f_1(e) & \text{si } e \in D_{f_1} \\ \vdots & \vdots \\ f_k(e) & \text{si } e \in D_{f_k} \end{cases}$$

- **Lema:** Sean $O \in \{\omega, \Sigma^{*m}\}$ y $n, m \in \omega$, y supongamos que $f_i : D_{f_i} \subseteq \omega^n \times \Sigma^{*m} \rightarrow O \forall i = 1, \dots, k$ son Σ -p.r. tales que $\forall i \neq j, D_{f_i} \cap D_{f_j} = \emptyset$. Entonces $f_1 \cup \dots \cup f_k$ es Σ -p.r.
- **Consejo:** Si se quiere usar este lema para probar que una función f es Σ -p.r., entonces primero hay que definir las funciones f_1, \dots, f_k tales que $\forall i \neq j, D_{f_i} \cap D_{f_j} = \emptyset$ y $f = f_1 \cup \dots \cup f_k$ y luego comenzar a probar y ver si algo es o no Σ -p.r.
 - Determinar el k , es decir, la cantidad de casos en la descripción de f .
 - Para cada caso de la descripción de f , asociar un subconjunto del dominio de f el cual sea justamente definido por la propiedad correspondiente (tener en cuenta que debe ser *subconjunto* y una descripción puede no usar todas las variables)

- Notar que los subconjuntos S_1, \dots, S_k definidos deben ser disjuntos de a pares y, unidos, deben dar el dominio de f
- Para cada i define f_i de la siguiente manera:
 - Dominio de f_i es S_i
 - Regla de f_i dada por la regla de describe f para el caso i -ésimo
- En general, suele suceder que f_i es la restricción a S_i de una función con dominio más amplio. Por ello, a veces se prueba entonces que tanto dicha función como S_i son Σ -p.r., resultando así, por lema, que f_i es Σ -p.r.
- *Ejemplo:* Con esto, se puede probar que $\lambda i \alpha [[\alpha]_i]$ es Σ -p.r.