TRABAJO FIN DE GRADO

Grado en Matemáticas

Curso 2021-2022

EL TEOREMA DE INCOMPLETITUD

UNIVERSIDAD COMPLUTENSE DE MADRID

Facultad de Ciencias Matemáticas

Departamento de Álgebra, Geometría y Topología

Alumno: Esteban Joaquín Jiménez Párraga

Tutor: Daniel Palacín Cruz

Junio, 2022

Resumen

En el campo de la Lógica Matemática, el Teorema de Incompletitud de Gödel es uno de los resultados más significativos tanto por su importancia histórica como por su desarrollo dentro del marco de la Lógica de Primer Orden. El objetivo de este trabajo es presentar un recorrido a través de los resultados que nos permitirán demostrar este teorema.

Abstract

In the field of Mathematical Logic, Gödel's Incompleteness Theorem is one of the most relevant results due to its historic relevance and its contribution to First Order Logic. The aim of this paper is to present a guide of results that will allow us to prove this theorem.

Índice general

Introducción

A comienzos del Siglo XX, las matemáticas se encontraban en plena crisis de los fundamentos. Que la Teoría de Conjuntos de Cantor se viera comprometida por la Paradoja de Russell, así como la gran división que había entre la comunidad matemática acerca del concepto de infinito, había dejado a la lógica en un punto muerto desde el cual los matemáticos difícilmente podían avanzar.

Fue entonces cuando David Hilbert, Catedrático en la Universidad de Göttingen y uno de los fundadores de la teoría de la demostración, propuso entre 1920 y 1930 lo que se conocería como el Programa de Hilbert. Este era un proyecto de investigación que conciliase los conflictos del infinito y diera unas bases sólidas dentro del marco de la lógica a partir de las cuales construir el conocimiento matemático.

Pensaba que su objetivo podía lograrse mostrando que toda la matemática se sigue de un sistema finito de axiomas escogidos correctamente y que dicho conjunto de axiomas era consistente y que permitía que las demostraciones fueran computables. Además, de una forma natural, creía que el conjunto axiomático sobre el que empezar a buscar era el de la aritmética.

Sin embargo, cuando estaba a punto de presentar oficialmente esta idea como el nuevo rumbo a seguir de las matemáticas y como uno de los grandes éxitos de su carrera, un matemático austrohúngaro llamado Kurt Gödel anunció en 1931 que había demostrado un teorema que probaba precisamente lo contrario. Si se exige que estas demostraciones sean computables, entonces es imposible dar axiomas para la aritmética que permitan, a partir de ellos, demostrar todas las verdades de la teoría. Este teorema se conoce bajo el nombre de Teorema de Incompletitud de Gödel y supondría uno de los resultados más importantes y sorprendentes acerca de los límites de las matemáticas.

A continuación, presentamos un recorrido acerca de todas las ideas y conceptos necesarios para probar este teorema. En el Capítulo 1 empezaremos recordando algunos conceptos claves de la lógica de primer orden como son la sintaxis, la semántica y la noción de completitud. Después, en el Capítulo 2, trataremos las funciones computables definidas en el sentido de Gödel. En el Capítulo 3 exploraremos un poco más en profundidad la Aritmética de Peano y la representación de esta en el lenguaje de la lógica de primer orden. Después desarrollamos la numeración de Gödel en el Capítulo 4, la herramienta de codificación para las fórmulas lógicas que él creó y que constituyen la base de la demostración del famoso teorema y finalmente, en el Capítulo 5, presentamos la demostración formal.

Capítulo 1

Conceptos preliminares de Lógica de Primer Orden

1.1. Lenguaje y sintaxis

Para aproximarnos a la demostración del Teorema de Incompletitud de Gödel debemos antes especificar algunos conceptos previos pertenecientes a la Lógica de Primer Orden presentados en la asignatura de Lógica Matemática [2]. En primer lugar, las nociones de lenguaje y estructura sobre un universo.

Definición 1.1. (Lenguaje). Un lenguaje es un terna $\mathfrak{L} = \mathcal{C} \dot{\cup} \mathcal{F} \dot{\cup} \mathcal{R}$, donde:

- $\mathcal{C}$ es un conjunto de símbolos de constante,
- $\mathcal{F}$ es un conjunto de símbolos de función,
- $\mathcal{R}$ es un conjunto de símbolos de relación.

donde cada símbolo de $\mathcal{F}$ y $\mathcal{R}$ tiene asociado un número natural $n \geq 1$ que llamamos aridad.

Además de los símbolos propios del lenguaje, disponemos también de unos símbolos lógicos auxiliares que son los paréntesis "(" y ")", la igualdad "$=$", la negación "$\neg$", los conectores "$\wedge$" , "$\vee$" y "$\rightarrow$", los cuantificadores existencial "$\exists$" y universal "$\forall$"; además de variables que podemos denotar como $x_0, x_1, \ldots, x_n, \ldots$

Definición 1.2. Sea $\mathfrak{L}$ un lenguaje de primer orden. Una $\mathfrak{L}$-estructura $\mathcal{A}$ consiste de un conjunto no vacío A, que denominamos universo de $\mathcal{A}$, y de una interpretación $s^{\mathcal{A}}$ para cada símbolo $s \in \mathfrak{L}$ de acuerdo con lo siguiente:

- la interpretación de un símbolo de constante es un elemento de A, es decir, $c^{\mathcal{A}} \in A$ para cada símbolo de constante $c \in \mathcal{C}$.

- la interpretación de un símbolo de función $f \in \mathcal{F}$ de aridad n es una función $f^{\mathcal{A}} : A^n \to A$.
- la interpretación de un símbolo de relación $R \in \mathcal{R}$ de aridad n es un subconjunto $R^{\mathcal{A}}$ de A^n.

Veamos algunos ejemplos de este concepto.

Ejemplo 1.3.
1. Consideramos el lenguaje vacío $\mathfrak{L}_\emptyset = \emptyset$, cualquier conjunto no vacío es una $\mathfrak{L}_\emptyset$-estructura.
2. El lenguaje de grafos es $\mathfrak{L}_{gf} = \{R\}$ donde R es un símbolo de relación de aridad 2. Si consideramos $\mathcal{V} =(V, E)$ un grafo entonces este actúa como universo interpretando el símbolo de relación R como $R^{\mathcal{V}} = E$.
3. Cualquier cuerpo forma una $\mathfrak{L}_{ring}$-estructura siendo $\mathfrak{L}_{ring} = \{+, -, \cdot, 0, 1\}$ donde 0 y 1 son símbolos de constante y $+, \cdot$ y $-$ son funciones de aridad 2, 2 y 1 respectivamente.

Una vez visto esto, podemos trabajar la sintaxis sobre la que construimos la lógica, que son fundamentalmente los términos, las fórmulas y, dentro de estas; destacamos especialmente las sentencias ligadas al concepto de aparición libre de una variable. De ahora en adelante, consideramos $\mathfrak{L}$ como un lenguaje cualquiera.

Definición 1.4. ($\mathfrak{L}$-término) Un $\mathfrak{L}$-término es una palabra constuida a partir de los símbolos de constante y de función del lenguaje, así como variables, siguiendo las siguientes reglas:

- cualquier variable x y cualquier constante c es un $\mathfrak{L}$-término.
- si f es un símbolo de función de aridad n y $t_1, \ldots, t_n$ son $\mathfrak{L}$-términos, entonces $f(t_1, \ldots, t_n)$ es también un $\mathfrak{L}$-término.

Dada una $\mathfrak{L}-$estructura $\mathcal{A}$, los elementos $a_1, a_2, \ldots, a_n \in A$ y un término $t = t(x_1, \ldots, x_n)$ con variables $x_1, \ldots, x_n$ escribimos $t[a_1, \ldots, a_n]$ para denotar el elemento de A obtenido al sustituir cada x_i por a_i, $i = 1, \ldots, n$.

Llamamos complejidad del término al número de símbolos de función que aparece en este. A partir de los términos, construimos las fórmulas de un lenguaje.

Definición 1.5. ($\mathfrak{L}$-fórmula) Una $\mathfrak{L}$-fórmula atómica es una de las siguientes:

- una ecuación $(t_1 = t_2)$ donde t_1 y t_2 son $\mathfrak{L}$-términos.
- una expresión de la fórmula $R(t_1, \ldots, t_n)$ donde R es un símbolo de relación n-ario y $t_1, \ldots, t_n$ son $\mathfrak{L}$-términos.

Con las fórmulas atómicas, definimos las $\mathfrak{L}$-fórmulas de un lenguaje son las expresiones obtenidas mediantes las siguientes reglas.

- Si φ es una $\mathfrak{L}$-fórmula atómica, entonces es una $\mathfrak{L}$-fórmula.

- Si φ es una $\mathfrak{L}$-fórmula, entonces también lo es $\neg\varphi$.
- Si φ y ψ son $\mathfrak{L}$-fórmulas, entonces también lo es $(\varphi \wedge \psi)$.
- Si x es una variable y φ es una $\mathfrak{L}$-fórmula, entonces $\exists x\varphi$ también lo es.

De forma similar a los términos, definimos la complejidad de una fórmula como el número de apariciones de los símbolos $\neg, \exists$ y $\wedge$. También utilizaremos la extendida notación:

- $(\varphi \vee \psi) = \neg(\neg\varphi \wedge \neg\psi)$
- $(\varphi \to \psi) = \neg(\varphi \wedge \neg\psi)$
- $(\varphi \leftrightarrow \psi) = ((\varphi \to \psi) \wedge (\psi \to \varphi)$
- $\forall x\varphi = \neg\exists x\neg\varphi$

Definición 1.6. (Aparición libre) Dada una $\mathfrak{L}$-fórmula φ, decimos que la variable x tiene una aparición libre en φ si no viene precedida por ningún cuantificador $\exists$. Cuando una aparición no es libre, decimos que es ligada.

Definición 1.7. (Sentencia) Una sentencia es una fórmula sin variables libres.

1.2. Semántica

Exponemos ahora el concepto de satisfacción adentrándonos en el campo de la semántica de la lógica, que nos va a permitir interpretar estás fórmulas lógicas dentro de una estructrura determinada.

Definición 1.8. (Satisfacción). Sea $\varphi = \varphi(x_1, \ldots, x_n)$ una fórmula de $\mathfrak{L}$ y sea $\{a_1, a_2, \ldots, a_n\}$ un conjunto formado por valores dentro del universo de una $\mathfrak{L}$-estructura $\mathcal{A}$. Definimos la relación de satisfacción

$$\mathcal{A} \models \varphi[a_1, a_2, \ldots, a_n]$$

recursivamente sobre los términos que compone φ siguiendo que

- $\mathcal{A} \models (t_1 \doteq t_2)[a_1, a_2, \ldots, a_n]$ si y solo si $t_1[a_1, a_2, \ldots, a_n] = t_2[a_1, a_2, \ldots, a_n]$
- $\mathcal{A} \models \mathcal{R}(t_1, \ldots, t_n)[a_1, a_2, \ldots, a_n]$ si y solo si $(t_1[a_1, a_2, \ldots, a_n], \ldots, t_n[a_1, a_2, \ldots, a_n]) \in \mathcal{R}$
- $\mathcal{A} \models \neg\varphi[a_1, a_2, \ldots, a_n]$ si y solo si $\mathcal{A} \not\models \varphi[a_1, a_2, \ldots, a_n]$
- $\mathcal{A} \models (\varphi \wedge \psi)[a_1, a_2, \ldots, a_n]$ si y solo si $\mathcal{A} \models \varphi[a_1, a_2, \ldots, a_n]$ y $\mathcal{A} \models \psi[a_1, a_2, \ldots, a_n]$
- $\mathcal{A} \models \exists x\varphi[a_1, a_2, \ldots, a_n]$ si y solo si existe un elemento a tal que $\mathcal{A} \models \varphi[a, a_1, a_2, \ldots, a_n]$ donde $\varphi = \varphi(x, x_1, \ldots, x_n)$.

Esta relación $\mathcal{A} \models \varphi[a_1, a_2, \ldots, a_n]$ se lee como $\mathcal{A}$ satisface φ con $a_1, a_2, \ldots, a_n$ o φ es verdadera en $\mathcal{A}$ respecto a $a_1, a_2, \ldots, a_n$. En el caso de que la satisfacción se cumpla para cualquier valoración, lo expresaremos simplemente como $\mathcal{A} \models \varphi$.

1.3. Demostración formal

Por último en esta sección, mostramos la definición de demostración formal y de que una fórmula sea demostrable. Para ello necesitamos establecer algunas reglas previas.

Sea x una variable y s un $\mathfrak{L}$-término. La sustitución de x por s en un $\mathfrak{L}$-término es un nuevo $\mathfrak{L}$-término que denotamos por $t_{s/x}$ o $t(s/x)$, el cual se obtiene de reemplazar cada variable x que aparezca en t por s. Análogamente, para una $\mathfrak{L}$-fórmula φ, será la sustitución de x por s en $\mathfrak{L}$-fórmula φ es una nueva $\mathfrak{L}$-fórmula que denotamos por $\varphi_{s/x}$ o $\varphi(s/x)$ que se obtiene sustituyendo en los términos de φ.

Una asignación en lógica proposicional es una función v que asigna a cada variable proposicional un valor 0 («falso») o 1 («verdadero»). Esta asignación se extiende recursivamente a v^* mediante las reglas $v^*(\neg p) = 1 - v^*(p)$ y $v^*(p \wedge q) = v^*(p) \cdot v^*(q)$.

Con esto, decimos que una $\mathfrak{L}$-fórmula φ es una tautología si para toda interpretación proposicional v de $\mathfrak{L}$ se tiene que $v^*(\varphi) = 1$.

Exponemos ahora las siguientes definiciones que usaremos en la definición de demostración formal.

Definición 1.9. (Axiomas de igualdad). Se consideran las siguientes $\mathfrak{L}$-fórmulas como axiomas de igualdad, que se satisfacen en cualquier $\mathfrak{L}$-estructura para toda valoración:

- $\forall x\ (x = x)$.
- $\forall x, y(x = y \to y = x)$.
- $\forall x, y, z((x = y \wedge y = z) \to x = z)$.
- $\forall x_1, \ldots, x_{2n}(\bigwedge_{i=1}^{n} x_i = x_{n+i} \to f(x_1, \ldots., x_n) = f(x_{n+1}, \ldots, x_{2n})$, donde f es una función n-aria.
- $\forall x_1, \ldots, x_{2n}(\bigwedge_{i=1}^{n} x_i = x_{n+i} \to R(x_1, \ldots., x_n) = R(x_{n+1}, \ldots, x_{2n})$, donde R es una función n-aria.

Definición 1.10. (Demostración formal). Una demostración formal en una $\mathfrak{L}$-teoría T es una sucesión finita de $\mathfrak{L}$-fórmulas $\varphi_1, \ldots, \varphi_n$ tal que para todo i con $1 \leq i \leq n$, se tiene que φ_i cumple una de las condiciones siguientes:

- es una tautología o un axioma de igualdad.
- es una sentencia de T.
- es una instancia $(\varphi_{t/x} \to \exists x\varphi)$ del axioma de $\exists$-Sustitución.
- existen índices $j, k < i$ tales que $\varphi_j = (\varphi_k \to \varphi_i)$, es decir, φ_i se deduce de φ_j y φ_k por Modus Ponens.
- existe $j < i$ tal que $\varphi_j = (\psi \to \chi)$ y $\varphi_i = (\exists x\psi \to \chi)$ donde x no es una variable libre de ψ, es decir, φ_i se deduce de φ_j por el axioma de $\exists$-Introducción.
  Las reglas mencionadas en esta definición son las siguientes:

1. Axioma de $\exists$-sustitución: Sea φ una $\mathfrak{L}$-fórmula, t un $\mathfrak{L}$-término y x una variable no precedida por ningún cuantificador existencial en el término t en φ. La fórmula $(\varphi_{t/x} \to \exists x\varphi)$ se satisface en cualquier $\mathfrak{L}$-estructura para toda valoración.
2. Modus Ponens: Si φ y $(\varphi \to \psi)$ se satisfacen en cualquier $\mathfrak{L}$-estructura para toda valoración, también lo hace ψ.
3. Axioma de $\exists$-introducción: Sean φ y ψ $\mathfrak{L}$-fórmulas, t un $\mathfrak{L}$-término y supongamos que x no es una variable libre de φ. Si $(\varphi \to \psi)$ se satisfacen en cualquier $\mathfrak{L}$-estructura para toda valoración, también lo hace $(\exists x\varphi \to \psi)$.

Con esto, decimos que una $\mathfrak{L}$-fórmula φ es demostrable en una $\mathfrak{L}$-teoría T si es la última fórmula de una demostración formal sobre T. Esto lo denotamos como $T \vdash \varphi$.

Definición 1.11. (Teoría y modelo). Una $\mathfrak{L}$-teoría T es un conjunto de sentencias de $\mathfrak{L}$. Si existe una $\mathfrak{L}$-estructura $\mathcal{A}$ que cumple $\mathcal{A} \models \varphi$ para toda φ sentencia de T, decimos que $\mathcal{A}$ es un modelo de T. Escribimos $\mathcal{A} \models T$.

1.4. Consistencia y Completitud

En esta última sección preliminar tratamos de dar algunas propiedades de las teorías y que usaremos en todo lo que resta de trabajo:

Definición 1.12. (Consistencia) Decimos que una teoría T es consistente si no existe ninguna sentencia del lenguaje χ tal que $T \vdash \chi$ y $T \vdash \neg\chi$. En caso contrario decimos que T es inconsistente.

Un resultado importante relacionado con la consistencia, es el Teorema de Henkin, que se demuestra en [2, Capítulo 2.4].

Teorema 1.13. *(Teorema de Henkin) Una teoría es consistente si y solo si tiene un modelo*

Definición 1.14. (Completitud) Una teoría T es completa si es consistente y para cualquier sentencia χ se verifica $T \vdash \chi$ o $T \vdash \neg\chi$.

Otro resultado fundamental es el Teorema de Completitud de Gödel, también demostrado en [2, Capítulo 2.5].

Teorema 1.15. *(Teorema de Completitud de Gödel) Sea T una $\mathfrak{L}$-teoría y χ una $\mathfrak{L}$-sentencia. Entonces*

$$T \models \chi \iff T \vdash \chi$$

Con esto ya tenemos todas las definiciones preliminares que necesitamos para expresar el Teorema de Incompletitud, que comenzaremos a tratar en la siguiente sección; así como los pasos detallados que nos permitiran llegar a su demostración.

Capítulo 2

Funciones Computables

2.1. Recursión como computabilidad

Considerando el conjunto que nos definirá el lenguaje de la aritmética $L = \{0, S, +, \cdot, <\}$ donde S es un símbolo de función de aridad 1, $+$ y $\cdot$ son símbolos de función de aridad 2 y $<$ es un símbolo de relación de aridad 2. Podemos expresar de una forma sencilla el Teorema de Incompletitud como que para que cualquier conjunto finito de axiomas computable, la teoría aritmética resultante será incompleta.

A priori, la intuición nos dice que el Teorema de Completitud expuesto al final del capítulo 1 y el Teorema de Incompletitud no son compatibles, que uno deniega directamente al otro. Sin embargo, esta idea de buscar un conjunto de axiomas para la aritmética marca la diferencia de estos dos en el primer requisito de esta expresión simplificada del Teorema.

El Teorema de Completitud nos dice que tomando una teoría, ser una sentencia verdadera es equivalente a ser demostrable. Por otra parte el Teorema de Incompletitud nos dice que si buscamos, sobre la aritmética, una teoría con un conjunto de axiomas consistentes y *computables,* estos no podrán dar lugar a una teoría completa. Es decir, si quisieramos podríamos dotar a la aritmética de una serie de axiomas completos, pero estos no podrían ser computables, luego no serían útiles en la búsqueda de la mecanización del proceso de demostraciones que buscaba David Hilbert. Por otro lado, podemos encontrar estos axiomas «mecanizables» pero estos no tendrán un poder expresivo suficiente para captar la verdad de todas las demostraciones aritméticas.

Es por esto que dedicamos este capítulo al concepto de lo computable. Intuitivamente, que el conjunto de axiomas Σ sea computable puede comprenderse como que existe un algoritmo que reconoce si cualquier sentencia en el lenguaje L pertenece o no a Σ. Aunque la noción de computabilidad encuentra su reflejo en muchos formalismos matemáticos equivalentes, como indica la Tesis Church-Turing, siguiendo el camino marcado por el Teorema de Incompletitud, presentamos en este trabajo la dada por el propio Gödel: las funciones recursivas. Por comodidad, nos referiremos a ellas directamente bajo el nombre de «computables».

Antes de proceder con ello, necesitamos establecer la siguiente notación.

Definición 2.1. Sea R una relación no vacía en $\mathbb{N}$. Denotamos como $\mu x(R(x))$ como el menor $x \in \mathbb{N}$ que cumple la relación $R(x)$. Asimismo, denotamos por $\mu x_{<a}(R(x))$ al menor $x \in \mathbb{N}$ con $x < a$, $a \in \mathbb{N}$, que satisface la relación $R(x)$. Si no existe tal x decimos que $\mu x_{<a}(R(x)) = a$.

Ejemplo 2.2. Tenemos $\mu x(x^2 < 7) = 3$, $\mu x_{<4}(x^2 > 3) = 2$ y $\mu x_{<2}(x > 5) = 2$ como ejemplos de cada caso respectivamente.

Definición 2.3. (Función Característica). Sea $R \subseteq \mathbb{N}^n$ una relación, se define la función característica $\chi_R : R^n \to \mathbb{N}$ como

$$\chi_R(a) = \begin{cases} 1 & \text{si } a \in R \\ 0 & \text{si } a \notin R \end{cases}$$

La función característica tiene la virtud de caracterizar elementos de un conjunto dado por una relación n-aria. Este es uno de los motivos por los que, a menudo, denotaremos $R(a_1, \ldots, a_n)$ en lugar de $(a_1, \ldots, a_n) \in R$. Igualmente, nos servirá para expresar una relación o un conjunto como una función, y viceversa. Así, el formalismo de las funciones computables se extenderá de una manera natural al concepto de relación y conjunto.

Un ejemplo de usos de esta función y el que es necesario tener en cuenta para dar el formalismo de las funciones computables es la función característica para la relación de orden $<$

$$\chi_<(m, n) = \begin{cases} 1 & \text{si } m < n \\ 0 & \text{si } m \geqslant n \end{cases}$$

Por último, necesitaremos definir las funciones coordenada:

Definición 2.4. (Funciones coordenada). Para $i = 1, \ldots, n$ definimos la función coordenada $I_i^n : \mathbb{N}^n \to \mathbb{N}$ como $I_i^n(a_1, \ldots, a_n) = a_i$.

A continuación presentamos la definición de función computable.

Definición 2.5. (Función computable). Las funciones computables son aquellas funciones con dominio $\mathbb{N}^n$ e imagen en $\mathbb{N}$ que se obtienen inductivamente aplicando las siguientes reglas:

- (R1) Las funciones $+ : \mathbb{N}^2 \to \mathbb{N}, \cdot : \mathbb{N}^2 \to \mathbb{N}$, $\chi_< : \mathbb{N}^2 \to \mathbb{N}$ y las funciones coordenada $I_i^n : \mathbb{N}^n \to \mathbb{N}$ para $i = 1, \ldots, n$ son computables.
- (R2) Dada $G : \mathbb{N}^m \to \mathbb{N}$ y $H_1, \ldots, H_m : \mathbb{N}^n \to \mathbb{N}$ computables, también lo es la función $F = G(H_1, \ldots, H_m) : \mathbb{N}^n \to \mathbb{N}$ definida como $F(a) = G(H_1(a), \ldots, H_m(a))$.
- (R3) Si $G : \mathbb{N}^{n+1} \to \mathbb{N}$ es computable y para todo $a \in \mathbb{N}^n$ existe $x \in \mathbb{N}$ tal que $G(a, x) = 0$, entonces la función $F : \mathbb{N}^n \to \mathbb{N}$ dada por

  $$F(a) = \mu x(G(a, x) = 0)$$

  es computable.

Con esta definición ya tenemos un formalismo que nos va a servir para determinar qué es computable y qué no. Como veremos a continuación, esto se aplica para todas aquellas funciones que podríamos decir son "intuitivamente computables". Veamos algunos ejemplos y lemas que trabajan sobre ello. En primer lugar, observamos un útil ejemplo para exponer el manejo de las funciones coordenada en la composición de funciones computables.

Ejemplo 2.6. Supongamos que $F : \mathbb{N}^3 \to \mathbb{N}$ y $G : \mathbb{N}^2 \to \mathbb{N}$ son funciones computables. Entonces la función definida como $H(x_1, x_2, x_3, x_4) = F(G(x_1, x_4), x_2, x_4)$ es computable. Esto se deduce de tomar las funciones coordenadas presentadas en (R1) y mediante la regla de composición que nos deja (R2). Así, H se expresa como, sea $x = (x_1, x_2, x_3, x_4)$,

$$H(x) = F(G(I_1^4(x), I_4^4(x)), I_2^4(x), I_4^4(x)).$$

Como se puede ver, el uso de la función coordenada es claro, aunque en ocasiones demasiado engorroso, por lo que en lo siguiente se omitirá si no se considera necesario explicitar su uso, escribiendo directamente la coordenada devuelta en la imagen de la función.

Ejemplo 2.7. Encontramos otros ejemplos en los casos de funciones que todos conocemos: La suma de n elementos se expresa de forma recursiva aplicando la regla (R2) sobre

$$+_n(x_1, \ldots, x_n) = +(+_{n-1}(x_1, \ldots, x_{n-1}), x_n).$$

Lema 2.8. *Sea $a \in \mathbb{N}^n$ y $k \in \mathbb{N}$. Cada función constante $c_k^n : \mathbb{N}^n \to \mathbb{N}$ definida como $c_k^n(a) = k$ es computable.*

Demostración. Procederemos por inducción sobre k. Para $k = 0$ empleamos (R3) sobre

$$c_0^n(a) = \mu x(I_{n+1}^{n+1}(a, x) = 0).$$

Supongamos que se cumple para un k genérico, probemos que también se da para $k + 1$.

$$c_{k+1}^n(a) = k + 1 = \mu x(k < x) = \mu x(c_k^n(a) < x) = \mu x(\chi_<(c_k^n(a), I_{n+1}^{n+1}(a, x)) = 0).$$

Y sabemos que por hipótesis de inducción c_k^n es computable y que $\chi_<$ también es computable por definición de (R1). □

Ejemplo 2.9. La función $\chi_\leq(m, n)$ definida por

$$\chi_\leq(m, n) = \begin{cases} 1 & \text{si } m \leq n \\ 0 & \text{si } m > n \end{cases}$$

es computable. Esto es debido a que $m \leq n$ si y solo si $m < n+1$, luego $\chi_\leq(m, n) = \chi_<(m, n+c_1^1(n)))$, que sabemos computable por (R1).

Antes de ver las siguientes propiedades de la computabilidad, aclaramos que gracias a la función

característica χ podemos representar una relación R como una función, por lo que tendrá sentido hablar de "relaciones computables"

Definición 2.10. Dadas las funciones $H_1, \ldots, H_m : \mathbb{N}^n \to \mathbb{N}$ y la relación $R \subseteq \mathbb{N}^m$. Para $a \in \mathbb{N}^n$ decimos

$$R(H_1, \ldots, H_m)(a) \text{ si y solo si } R(H_1(a), \ldots, H_m(a)).$$

Lema 2.11. *Sean las funciones $H_1, \ldots, H_m : \mathbb{N}^n \to \mathbb{N}$ y la relación $R \subseteq \mathbb{N}^m$ todas computables. Entonces $R(H_1, \ldots, H_m) \subseteq \mathbb{N}^n$ es computable.*

Demostración. Dado $a \in \mathbb{N}^n$ se tiene que $\chi_{R(H_1,\ldots,H_m)}(a) = \chi_R(H_1(a), \ldots, H_m(a))$. Así $R(H_1, \ldots, H_m)$ se deduce de (R2) por la definición de χ_R. □

Lema 2.12. *(Definición por casos). Sean $R_1, \ldots, R_k \subseteq \mathbb{N}^n$ relaciones computables tales que para cada $a \in \mathbb{N}^n$ exactamente una de las relaciones $R_1(a), \ldots, R_k(a)$ se cumple, es decir, $\mathbb{N}^n$ es la unión disjunta de $R_1, \ldots, R_k$. Si las funciones $G_1, \ldots, G_k : \mathbb{N}^n \to \mathbb{N}$ son computables. Entonces la función $G : \mathbb{N}^n \to \mathbb{N}$ dada por*

$$G(a) = \begin{cases} G_1(a) & \text{si } R_1(a) \\ \vdots & \vdots \\ G_k(a) & \text{si } R_k(a) \end{cases}$$

es computable.

Demostración. G puede expresarse a partir de la función característica y las funciones computables por (R1) + y $\cdot$ usando que $\mathbb{N}^n$ es la unión disjunta de $R_1, \ldots, R_k$, que sabemos por hipótesis computables. Con esto, se define $G = G_1 \cdot \chi_{R_1} + \ldots + G_k \cdot \chi_{R_k}$ y finalmente por la composición de (R2) deducimos que es computable. □

Veremos ahora cómo se pueden emplear estas reglas para crear otros operadores muy comunes en las matemáticas.

Lema 2.13. *Las funciones $\chi_{\geq}$,$\chi_{=} : \mathbb{N}^2 \to \mathbb{N}$ son computables.*

Demostración. Empecemos por $\chi_{\geq}$. Este se puede expresar como

$$\chi_{\geq}(m, n) = \chi_{\leq}(I_2^2(m, n), I_1^2(m, n)).$$

Básicamente, aplicamos $\chi_{\leq}$ sobre las variables intercambiando su posición. Para $\chi_{=}$, basta definirla como $\chi_{=}(m, n) = \chi_{\geq}(m, n) \cdot \chi_{\leq}(m, n)$. Así, $\chi_{=}(m, n) = 1$ si y solo si $m \leq n$ y $m \geq n$. □

Lema 2.14. *Supongamos que $P, Q \subseteq \mathbb{N}^n$ son relaciones computables. Entonces $\neg P$ y $P \wedge Q$ (donde $\neg P$ representa el complementario y $P \wedge Q$ la intersección) son también relaciones computables.*

Demostración. Sea $a \in \mathbb{N}^n$. Se puede expresar entonces $\neg P$ como $\chi_{\neg P}(a) = \chi_{=}(\chi_P(a), c_0^n(a))$,por lo que es computable por (R2) ya que sabemos por 2.13 que $\chi_{=}$ es computable y por 2.8 que la

función constante también lo es. La relación $P \wedge Q$ es computable porque se puede expresar como $\chi_{P\wedge Q} = \chi_P \cdot \chi_Q$. □

Corolario 2.15. *Supongamos que $P, Q \subseteq \mathbb{N}^n$ son relaciones computables. Entonces las relaciones $P \vee Q = \neg(\neg P \wedge \neg Q)$, $P \to Q = \neg(P \wedge \neg Q)$ y $P \leftrightarrow Q = (P \to Q) \wedge (Q \to P)$ son también computables.*

Demostración. Usando la proposición anterior se obtiene inmediatamente observando el uso de la negación $\neg$. □

Corolario 2.16. *Las relaciones $<, >$ y $\neq$ son computables.*

Demostración. Combinando los resultados anteriores se ve que estas tres son complementarias de $\geq, \leq$ y $=$ respectivamente, lo cual se puede interpretar como un caso de $\neg P$. □

Lema 2.17. *Sea $R \subseteq \mathbb{N}^{n+1}$ una relación computable tal que para todo $a \in \mathbb{N}^n$ existe un $x \in \mathbb{N}$ con $(a, x) \in R$. Entonces la función $F : \mathbb{N}^n \to \mathbb{N}$ dada por $F(a) = \mu x R(a, x)$ también es computable.*

Demostración. Se puede interpretar $F(a) = \mu x R(a, x)$ usando la función característica de R para redefinir la función como $F(a) = \mu x(\chi_{\neg R}(a, x) = 0)$. Observamos que F es computable por la regla (R3) ya que $\neg\chi$ es computable por el Lema 2.14. □

Lema 2.18. *Sea $R \subseteq \mathbb{N}^{n+1}$ una relación computable y sean $P, Q \subseteq \mathbb{N}^{n+1}$ las relaciones definidas por*

1. *$P(a, y) \iff$ existe un $x \in \mathbb{N}$ tal que $x < y$ y $R(a, x)$*
2. *$Q(a, y) \iff$ para todo $x \in \mathbb{N}$ si $x < y$ entonces $R(a, x)$.*

Entonces P y Q también son relaciones computables.

Demostración. 1. Que existe un $x \in \mathbb{N}$ tal que $x < y$ y $R(a, x)$ tiene dos condiciones, que $x < y$ y que $R(a, x)$. Lo que buscamos es expresar la existencia de esto, así que usamos lo siguiente $(\mu x(R(a, x) \vee (x = y)) < y$, esto nos acota el x por el que nos preguntamos a uno que satisfaga $R(a, x) \wedge x < y$ o bien a $x = y$, no dejando nunca un resultado vacío.

Después, comprueba si el resultado de la expresión $(\mu x(R(a, x) \vee (x = y))$ es menor que y. Para el primer resultado, devuelve un $x < y$ y $R(a, x)$, por lo que efectivamente tendríamos que $(\mu x(R(a, x) \vee (x = y)) < y$ es cierto. Si estamos en el segundo, como $x = y$, $(\mu x(R(a, x) \vee (x = y)) < y$ es falso. Por lo que efectivamente, $P(a, y) \iff (\mu x(R(a, x) \vee x = y) < y)$, que es computable por ser composición de funciones computables y por la regla (R3).

2. Usando lo demostrado respecto a la negación en 2.14, observamos que Q es computable si y solo si $\neg Q$ lo es. Y $\neg Q$ se expresa usando el apartado anterior como

$$\neg Q(a, y) \iff \text{existe un } x \in \mathbb{N} \text{ tal que } x < y \text{ y } \neg R(a, x).$$

□

Y también podemos ver su aplicación para definir funciones aritméticas.

Proposición 2.19. *La función* $\dot{-} : \mathbb{N}^2 \to \mathbb{N}$ *definida como* $\dot{-}(a,b) = \begin{cases} a-b & si\ a \geq b \\ 0 & si\ a < b \end{cases}$ *es computable.*

Demostración. Esta función así definida puede definirse disyuntivamente como

$$\dot{-}(a,b) = a\dot{-}b = \mu x(b + x = a \vee\ a < b).$$

□

Proposición 2.20. *La función de la división entera* $\div : \mathbb{N}^2 \to \mathbb{N}$ *definida como* $\div(a,b) = \begin{cases} \lfloor a/b \rfloor & si\ a \geq b \\ 0 & si\ a < b \end{cases}$ *es computable.*

Demostración. Podemos reescribir esta función como

$$\div(a,b) = a \div b = \mu x((\text{existe un } r \in \mathbb{N} \text{ tal que } r < b \text{ y } bx + r = a) \vee\ a < b).$$

y sabemos por el lema 2.18 que esta es la expresión de una función computable. □

2.2. La función β de Gödel

Como se ve, definir funciones simples o trabajar con operadores lógicos resulta relativamente sencillo, pero qué sucede con algunas operaciones más complejas como 2^x. Intuitivamente, son obviamente computables, tan solo se trata de multiplicar 2 recursivamente un número x de veces, pero ajustarlo al formalismo de las funciones computables no es para nada tan trivial. Para mostrar como se extiende finalmente esta noción de computabilidad a estas funciones más complejas y a fin de convencer de que, en efecto, la computabilidad y la recursión son lo mismo en el sentido que le da Gödel, se construye el siguiente mecanismo, que aplicaremos particularmente al caso de la función 2^x.

Necesitamos ahora dar una serie de definiciones previas a este.

Definición 2.21. Definimos la función de emparejamiento $\text{Pair} : \mathbb{N}^2 \to \mathbb{N}$ como

$$\text{Pair}(x,y) = \frac{(x+y)\cdot(x+y+1)}{2} + x$$

Lema 2.22. *Sean* $m, n \in \mathbb{N}$. *Si* $m + n < m' + n'$ *entonces* $\text{Pair}(m,n) < \text{Pair}(m',n')$.

Demostración. Sean $m, n \in \mathbb{N}$ y denotamos $m+n = k$. El valor máximo de $\text{Pair}(m,n)$ se alcanza para $\text{Pair}(k,0) = \frac{1}{2}k(k+1) + k$ y el mínimo para $\text{Pair}(0,k) = \frac{1}{2}k(k+1)$. Supongamos $m' + n' = k' \geq k+1$. Entonces $\text{Pair}(0,k') \geq \frac{1}{2}(k+1)(k+2) = \frac{1}{2}k(k+1) + k + 1 > \text{Pair}(k,0)$. Por lo que $\text{Pair}(m',n') \geq \text{Pair}(0,k') > \text{Pair}(k,0) \geq \text{Pair}(m,n)$. □

Proposición 2.23. *La función* Pair *es computable y establece una biyección entre* $\mathbb{N}^2$ *y* $\mathbb{N}$.

Demostración. Esta función es computable ya que sabemos que la división es computable, (el uso de $\div$ es adecuado porque el numerador siempre será un número par) y las funciones $+$ y $\cdot$ también lo son por (R1).

Para ver que es biyectiva, veamos que es inyectiva y sobreyectiva: Para ver la inyectividad, usamos que por la propiedad probada en el Lema 2.22 si $\text{Pair}(m,n) = \text{Pair}(m',n')$ entonces $m+n = m'+n'$. Además, por la definición de Pair, se sigue que $m = m'$ cancelando la fracción; luego $n = n'$.

Para ver la sobreyectividad tomemos $z \in \mathbb{N}$ cualquiera. Y sea $t_k = \frac{1}{2}k(k+1)$ el mayor número triangular no mayor que z. Tomamos $m = z - t_k$ y $n = k - m$. Entonces tenemos que

$$\text{Pair}(m,m) = \frac{1}{2}(m+n)(m+n+1)+m = \frac{1}{2}(m+k-m)(m+k-m+1)+z-t_k = \frac{1}{2}k(k+1)+z-t_k = z.$$

□

A partir de la función Pair podemos definir las siguientes funciones computables.

Definición 2.24. Definimos las funciones $\text{Left} : \mathbb{N} \to \mathbb{N}$ y $\text{Right} : \mathbb{N} \to \mathbb{N}$ a partir de Pair como

$$\text{Left}(a) = x \text{ si existe un } y \text{ tal que } \text{Pair}(x,y) = a$$

$$\text{Right}(a) = y \text{ si existe un } x \text{ tal que } \text{Pair}(x,y) = a$$

Proposición 2.25. *Las funciones* Left *y* Right *son computables.*

Demostración. Usamos su definición a partir de la función Pair y se deduce la propiedad de que esta es computable por el Lema 2.18 ya que tenemos que $\text{Left}(a) = \mu x(\text{existe un } y < a+1,\ \text{Pair}(x,y) = a)$ y $\text{Right}(a) = \mu y(\text{existe un } x < a+1,\ \text{Pair}(x,y) = a)$ y como sabemos por la proposición anterior, Pair es computable y por la regla (R3) se deduce el resultado. □

Lema 2.26. *La relación ternaria* $a \equiv b \mod c$ *en* $\mathbb{N}$ *es computable.*

Demostración. Consideremos $a, b, c \in \mathbb{N}$. La relación $\equiv \mod$ se da cuando el resto de dividir a y b entre c son iguales, o lo que es lo mismo, que al restar uno al otro, el resultado es múltiplo de c. Por tanto obtenemos que

$$a \equiv b \mod c \iff (\text{existe un } x < a+1,\ a = x \cdot c + b \text{ o existe un } x < b+1,\ b = x \cdot c + a).$$

y por el Lema 2.18 deducimos que es computable. □

Con esto ya podemos definir una función que será crucial de ahora en adelante y en la cual se basa la construcción de este método para dar lugar a funciones más complejas dentro de la teoría de la recursividad.

Definición 2.27. (Función β de Gödel). Definimos la función $\beta : \mathbb{N}^2 \to \mathbb{N}$ como

$$\beta(a,i) := \mu x(x \equiv \text{Left}(a) \mod (1 + (i+1)\text{Right}(a))).$$

Es decir, es el resto de dividir Left(a) entre $(1 + (i + 1)\text{Right}(a))$.

La utilidad de esta función viene dada por dos motivos que demostraremos a continuación. El primero, es que se trata de una función computable. El segundo, que dado una sucesión finita de números naturales $n_0, \ldots, n_k$ cualesquiera, para la función β existen siempre dos naturales a, i $(i = 0, \ldots, k)$ tales que $\beta(a, i) = n_i$. Es decir, va a ser nuestro método computable para poder llamar a un elemento concreto de una sucesión arbitraria.

Teorema 2.28. *La función β es computable. Además, para cualesquiera $a_0, \ldots, a_n \in \mathbb{N}$ existe un $a \in \mathbb{N}$ tal que*

$$\beta(a, 0) = a_0, \ldots, \beta(a, n) = a_n.$$

Demostración. La función β es computable ya que surge de aplicar (R3) sobre

$$(x \equiv \text{Left}(a) \mod (1 + (i + 1)\text{Right}(a))$$

ya que sabemos que la relación ternaria mód es computable por el Lema 2.26 y estamos aplicándola sobre Left(a) y $(1 + (i + 1)\text{Right}(a))$ y sabemos que tanto Left como Right son computables por 2.25.

Para lo segundo, definimos $m = \text{máx}(n, a_0, a_1, ..., a_n)$ y tomamos $N = m!$. Entonces por construcción tenemos que $1 + N, 1 + 2N, \ldots, 1 + nN, 1 + (n + 1)N$ son primos entre sí dos a dos.

Por reducción al absurdo, si esto no fuera así, existiría un p primo tal que $p|1 + iN$ y $p|1 + jN$ con $1 \leq i < j \leq n + 1$, con lo que p divide a la diferencia $p|(j - i)N$. Entonces tenemos que $p|N$ o $p|(j - i)$, pero como $(j - i) \leq n \leq m$ y $N = m!$, tenemos que, de cualquier forma, $p|N$ y en particular $p|iN$. Y de que $p|1 + iN$, deducimos que p divide a la diferencia $p|1 + iN - iN$, es decir, $p|1$, luego $p = 1$, contradicción.

Una vez probado que son primos entre sí, sabemos que por el Teorema Chino del Resto existe un $M \in \mathbb{N}$ tal que

$$M \equiv a_0 \mod 1 + N$$

$$M \equiv a_1 \mod 1 + 2N$$

$$\ldots$$

$$M \equiv a_n \mod 1 + (n + 1)N.$$

Ahora, tomando $a := \text{Pair}(M, N)$ obtenemos que $\text{Left}(a) = M$ y $\text{Right}(a) = N$ por lo que, por definición,

$$\beta(a, i) = \beta(\text{Pair}(M, N), i) = \mu x(x \equiv \text{Left}(\text{Pair}(M, N)) \mod (1 + (i + 1)\text{Right}(\text{Pair}(M, N)))) =$$

$$= \mu x(x \equiv M \mod (1 + (i + 1)N)) = a_i.$$

□

Como β es computable, lo podemos emplear para codificar una secuencia de números finita $a_0, \ldots, a_n$

en términos de un único número a. Veamos como aplicarlo para el caso 2^x.

Ejemplo 2.29. Supongamos que nuestra secuencia $a_0, \ldots, a_n$ cumple que $a_0 = 1$ y $a_{i+1} = 2 \cdot a_i$. Entonces $a_n = 2^n$ y se deduce de la proposición anterior que $\beta(a, n) = 2^n$ donde nos aseguramos de definir a como

$$a := \mu x(\beta(x, 0) = 1 \text{ y para todo } i < n, \beta(x, i+1) = 2 \cdot \beta(x, i)).$$

Con lo que, gracias a la función β, $x \mapsto 2^x$ es computable.

Esta construcción sugiere un método general para expresar funciones computables a partir de secuencias de números naturales. Esto motiva la siguiente definición.

Definición 2.30. (Números secuenciales). Dada una secuencia finita de números naturales $(a_1, \ldots, a_n)$ definimos su número secuencial $\langle a_1, \ldots, a_n \rangle = a \in \mathbb{N}$ como el menor número natural a tal que $\beta(a, 0) = n$ y $\beta(a, i) = a_i$ para $i = 1, \ldots, n$.

Si $n = 0$, se tiene que $\langle\rangle = 0$, siendo este el número secuencial de la lista vacía.

Observamos que la existencia del número secuencial está garantizada por el Teorema 2.28

Lema 2.31. *La función* $(a_1, \ldots, a_n) \mapsto \langle a_1, \ldots, a_n \rangle = a$ *es computable. Además,* $a_i < a$ *para* $i = 1, \ldots, n$.

Demostración. Esta función es computable, pues se puede expresar como

$$a = \langle a_1, \ldots, a_n \rangle = \mu a(\beta(a, 0) = n, \beta(a, 1) = a_1, \ldots, \beta(a, n) = a_n).$$

Visto esto, usamos para ver que $a_i < a$ para $i = 1, \ldots, n$ la definición de la función β a partir de las funciones Left y Right y la propiedad de que $\text{Left}(x) < x$ para $0 < x$. Es decir,

$$a_i = \beta(a, i) = \mu x(x \equiv \text{Left}(a) \; mod \; (1 + (i+1)\text{Right}(a))) < \text{Left}(a) < a, \text{ para todo } i = 1, \ldots, n.$$

□

Definición 2.32. (Función longitud) Definimos la función longitud $\text{long} : \mathbb{N} \to \mathbb{N}$ como

$$\text{long}(a) = \beta(a, 0)$$

que es por construcción computable.

Al aplicar la función longitud sobre un número secuencial entonces obtenemos la longitud de la secuencia que lo caracteriza $\text{long}(\langle a_1, \ldots, a_n \rangle) = \beta(\langle a_1, \ldots, a_n \rangle, 0) = n$.

Con esto, el número secuencial de una sucesión finita dada es aquel que empleamos en la función β para obtener el valor deseado. Si intuitivamente podemos dar esta sucesión de manera recursiva, entonces también será computable en el sentido formal.

Al conjunto de todos los números secuenciales lo denotamos como $Seq \subseteq \mathbb{N}$.

Lema 2.33. *El conjunto $Seq \subseteq \mathbb{N}$ es computable.*

Demostración. Es computable por el Lema 2.18 pues

$$a \in Seq \iff \text{para todo } x < a \ (\log(x) \neq \log(a) \text{ o existe } i < \log(a) \text{ tal que } \beta(x,i) \neq \beta(a,i)).$$

ya que un número a es secuencial si es el mínimo que caracteriza una secuencia, es decir, si para todos los valores menores a este, es imposible que caractericen la misma secuencia que él, ya sea porque su longitud sea distinta o si los valores de la función β difieren en algún valor. □

Para concluir, en este capítulo hemos sido capaces de dar una solución al problema de como formalizar la noción de computabilidad. Más allá, hemos mostrado un método para codificar secuencias finitas de números de forma unívoca a partir de la función β y los números secuenciales. Será a partir de esto con lo que construiremos la numeración de Gödel sobre las fórmulas en la aritmética, base de la demostración del Teorema de Incompletitud, ya que será el puente que nos permita cruzar el marco aparentemente insalvable entre la aritmética y la lógica.

Por ello, vamos a añadir un resultado que nos será útil en el Capítulo 4 cuando tratemos este tema para ver que esta numeración es computable en nuestra definición formal cuando una función computable se invoque a sí misma sobre valores menores.

Proposición 2.34. *Sea $G : \mathbb{N}^n + 2 \to \mathbb{N}$ computable y $F : \mathbb{N}^{n+1} \to \mathbb{N}$ definida como*

$$F(a,b) = G(a,b,\langle F(a,0),\ldots,F(a,b-1)\rangle)$$

donde $a \in \mathbb{N}^n$ y $b \in \mathbb{N}$. Entonces F es computable.

Demostración. Denotamos $\bar{F}(a,b) = \langle F(a,0),\ldots,F(a,b-1)\rangle$. Veamos en primer lugar que $\bar{F}$ es computable. Tenemos que $\bar{F}(a,b)$ es un número secuencial $x = \langle x_1,\ldots,x_b\rangle$ de longitud b en la que cada x_i debe cumplir que $x_i = F(a,i)$ para $1 \leq x \leq b$. Como sabemos que G es computable, y $F(a,b) = G(a,b,\langle F(a,0),\ldots,F(a,b-1)\rangle)$, podemos reescribir esta función como la siguiente expresión que es computable por (R3)

$$\bar{F}(a,b) = \mu x(Seq(x) \text{ y } \log(x) = b \text{ y } \forall i < b(I_i^b(x) = G(a,i,\langle x_1,\ldots,x_i\rangle))).$$

Solo resta ver que si $\bar{F}$ es computable entonces lo es F y esto se da porque $F(a,b) = I_b^b(\bar{F}(a,b+1))$. □

Capítulo 3

La Aritmética de Peano

3.1. Los axiomas de Peano

En el capítulo anterior conseguimos lidiar con el primer problema que nos presenta el Teorema de Incompletitud, que es ser capaces de dar con una definición formal de la noción de computable. Con esto ya podríamos comenzar a recorrer el camino para probar la incompletitud de la aritmética. Sin embargo, queda una cuestión muy importante en el tintero aún, ¿cuál es esa aritmética que estamos tratando? Históricamente, el Teorema de Incompletitud se plantea sobre lo que conocemos como la aritmética de Peano. Esta fue expuesta formalmente por el matemático decimonónico Giuseppe Peano en una publicación de 1889 llamada *«Aritmetices principia, nova methodo exposita»*. En ellos se recogen nueve axiomas como el principio básico de la aritmética y se destaca de estos tres aspectos fundamentales, el uso del signo "=", la aparición de la función sucesor S y el esquema de inducción. Esta aritmética es, a menudo, la aritmética que empleamos todos (aunque simplificada obviando los usos de la función sucesor). Esta fue la aritmética en torno a la cual se orquesta el Teorema de Incompletitud y sobre el cual trabajó Gödel. La Teoría que forma la Aritmética de Peano está constituida por los siguientes axiomas, denotamos a este conjunto por (AP) sobre el lenguaje $L = \{0, S, +, \cdot, <\}$:

1. $\forall x \neg(Sx \neq 0)$
2. $\forall x \forall y(Sx = Sy \rightarrow x = y)$
3. $\forall x(x + 0 = x)$
4. $\forall x \forall y(x + Sy = S(x + y))$
5. $\forall x(x \cdot 0 = 0)$
6. $\forall x \forall y(x \cdot Sy = x \cdot y + x)$
7. $\forall x \forall y(x < Sy \leftrightarrow (x < y \vee x = y))$
8. $\forall x \forall y(x < y \vee x = y \vee y < x)$

9. $\forall x[(\varphi(x,0) \wedge \forall y(\varphi(x,y) \to \varphi(x,Sy))) \to \forall y\varphi(x,y)]$ donde $x = (x_1,\ldots,x_n)$ para un n arbitrario y $\varphi(x_1,\ldots,x_n)$ es una L-fórmula.

El axioma número 9 es lo que conocemos como principio de inducción, y en realidad representa la capacidad de llevar a cabo este axioma con todas las fórmulas del lenguaje L.

Ya presentamos al inicio del Capítulo 2 la estructura $\mathfrak{N} = (\mathbb{N}; 0, S, +, \cdot, <)$ donde $S : \mathbb{N} \to \mathbb{N}$ se interpreta como la función sucesor y $+, \cdot$ y $<$ siguen la interpretación natural. Denotaremos $S^0 0 = 0, S^1 0 = S0, \ldots$ y así sucesivamente para ahorrar en notación. Esta estructura se conoce como el *modelo estándar* de la Aritmética de Peano, y será suficiente con trabajar sobre esta para probar el Teorema. Aunque podemos encontrar otros modelos en anillos de polinomios.

Ejemplo 3.1. Tomando como universo $\mathbb{Z}[x]$, 0 representa al polinomio nulo, S será la operación de sumar 1 al polinomio, $+$ y $\cdot$ se comportan de la forma esperada y definimos que $f(x) < g(x)$ si y solo si existe n_0 tal que para todo $n \geq n_0$ se cumple $f(n) < g(n)$.

Lo más relevante de este primer análisis es la aparición, y de una forma tan natural, de un modelo; ya que nos dice que la teoría (AP) es consistente como consecuencia del Teorema de Henkin. Tratándose de un fundamento tan básico, no es de extrañar que a comienzos del S.XX, en plena crisis de los fundamentos, David Hilbert y gran parte de los matemáticos de la época apostaran por su completitud y su prometedora capacidad para cubrir el hueco de ser la base que conformaría los cimientos de las matemáticas. Este interés era claro, el sistema de axiomas (AP), consistente, parecía que iba a ser capaz de clasificar todas las fórmulas lógicas φ en dos: aquellas demostrables $(AP) \vdash \varphi$ y, por tanto, verdaderas; y las refutables $(AP) \nvdash \varphi$ y tendrían que ser consideradas como falsas. Sin embargo, el Teorema de Incompletitud prueba la existencia de una tercera clase de fórmulas que acabarían con las esperanzas de esta corriente de pensamiento: las fórmulas indecidibles, que no son ni demostrables ni refutables.

3.2. Representabilidad

Teniendo ya claro el modelo aritmético sobre el que vamos a trabajar y la herramienta fundamental sobre el que nos sustentamos, las funciones computables, veremos como se formulan estos dos en la lógica de primer orden. Es por esto que dedicamos esta sección al concepto de representabilidad. De manera informal, diremos que una función o relación es representable para un determinado conjunto de sentencias si podemos encontrar una fórmula que la «represente», es decir, que exprese y nos permita identificar lo mismo que el objeto que buscamos representar. Esto nos servirá para dar un resultado fundamental, el Teorema de Representabilidad, que nos garantiza que las funciones computables y las relaciones serán representables en nuestro sentido formal.

Definición 3.2. (Representación) Sea $L = \{0, S, +, \cdot, <\}$ el lenguaje de la aritmética con su interpretación usual y sea Σ un conjunto de L-sentencias.

- Decimos que una relación $R \subseteq \mathbb{N}^m$ es Σ-representable si existe una L-fórmula $\varphi(x_1,\ldots,x_m)$ tal que para todo $(a_1,\ldots,a_m) \in \mathbb{N}^m$ se cumple:

(i) Si $R(a_1, \ldots, a_m)$ entonces $\Sigma \vdash \varphi(S^{a_1}0, \ldots, S^{a_m}0)$

(ii) Si $\neg R(a_1, \ldots, a_m)$ entonces $\Sigma \vdash \neg\varphi(S^{a_1}0, \ldots, S^{a_m}0)$.

- Decimos que una función $F : \mathbb{N}^m \to \mathbb{N}$ es Σ-representable si existe una L-fórmula $\varphi(x_1, \ldots, x_m, y)$ tal que para todo $(a_1, \ldots, a_m) \in \mathbb{N}^m$ tenemos que

$$\Sigma \vdash \varphi(S^{a_1}0, \ldots., S^{a_m}0, y) \longleftrightarrow y = S^{F(a_1,\ldots,a_m)}0.$$

Siguiendo esta definición, diremos que la fórmula φ representa a la relación R o la función F en cada caso.

Lema 3.3. *Supongamos que $\mathcal{A} \models (AP)$ y consideremos el modelo estándar de la aritmética de Peano $\mathfrak{N} = (\mathbb{N}; 0, S, +, \cdot, <)$. Entonces existe un único homomorfismo*

$$\imath : \mathfrak{N} \to \mathcal{A}$$

tal que $\mathfrak{N}$ queda embebido en $\mathcal{A}$.

Demostración. El homomorfismo se define como la interpretación del valor $n \in \mathbb{N}$ en la estructura $\mathcal{A}$

$$\imath(n) = (S^n 0)^{\mathcal{A}}$$

para que $\mathfrak{N}$ queda embebido en $\mathcal{A}$, para todo $n \in \mathbb{N}$. Observamos que para cualquier otro homomorfismo $\imath : \mathfrak{N} \to \mathcal{A}$ se tiene que $\imath(n) = (S^n 0)^{\mathcal{A}} = \imath(n)$, es decir, el homomorfismo es único. □

Lema 3.4. *(Representabilidad de Relaciones) Sea $L = \{0, S, +, \cdot, <\}$ el lenguaje de la aritmética, Σ un conjunto de L-sentencias tales que $\Sigma \vdash \neg(S0 = 0)$ y $R \subseteq \mathbb{N}^m$ una relación. Entonces*

$$R \text{ es } \Sigma\text{-representable} \iff \chi_R \text{ es } \Sigma\text{-representable.}$$

Demostración. ($\Leftarrow$) Si suponemos que χ_R es Σ-representable entonces existe una L-fórmula $\varphi(x_1, \ldots, x_m, y)$ que la representa. A partir de esta, definimos $\psi(x_1, \ldots, x_m)$ como $\varphi(x_1, \ldots, x_m, S0)$ que veremos que representa a R. Comprobémoslo:

Si $(a_1, \ldots, a_m) \in R$ entonces $\chi_R(a_1, \ldots, a_m) = 1$, por lo que aplicando la definición de representabilidad para funciones

$$\Sigma \vdash \varphi(S^{a_1}0, \ldots., S^{a_m}0, y) \longleftrightarrow y = S^{\chi_R(a_1,\ldots,a_m)}0 = S0,$$

así, sustituyendo y por el Teorema de Completitud 1.15, $\Sigma \vdash \varphi(S^{a_1}0, \ldots, S^{a_m}0, S0)$, por lo que $\Sigma \vdash \psi(S^{a_1}0, \ldots, S^{a_m}0)$.

Por otra parte, si $(a_1, \ldots, a_m) \notin R$ entonces $\chi_R(a_1, \ldots, a_m) = 0$. Aplicamos la definición de representabilidad de nuevo y obtenemos

$$\Sigma \vdash \varphi(S^{a_1}0, \ldots., S^{a_m}0, y) \longleftrightarrow y = S^{\chi_R(a_1,\ldots,a_m)}0 = 0,$$

por lo que, como $\Sigma \vdash \varphi(S^{a_1}0, \ldots, S^{a_m}0, 0)$ y $\Sigma \vdash S0 \neq 0$, deducimos sustituyendo y por el Teorema de Completitud 1.15, tenemos que $\Sigma \vdash \neg\varphi(S^{a_1}0, \ldots, S^{a_m}0, S0)$, y consecuentemente $\Sigma \vdash \neg\psi(S^{a_1}0, \ldots, S^{a_m}0)$.

($\Rightarrow$) Asumamos ahora que R es representable y sea $\psi(x_1, \ldots, x_m)$ la fórmula que lo representa. Probaremos que la fórmula dada por

$$\varphi(x_1, \ldots, x_m, y) = (\psi(x_1, \ldots, x_m) \wedge y = S0) \vee (\neg\psi(x_1, \ldots, x_m) \wedge y = 0)$$

representa a χ_R.

Sea $(a_1, \ldots, a_m) \in R$, entonces $\Sigma \vdash \psi(S^{a_1}0, \ldots, S^{a_m}0)$ de donde

$$\Sigma \vdash [(\psi(S^{a_1}0, \ldots, S^{a_m}0) \wedge y = S0) \vee (\neg\psi(S^{a_1}0, \ldots, S^{a_m}0) \wedge y = 0)] \longleftrightarrow y = S0$$

y ahora, por nuestra definición $\Sigma \vdash \varphi(S^{a_1}0, \ldots, S^{a_m}0, y) \longleftrightarrow y = S^{\chi_R(a_1,\ldots,a_m)}0$, por lo que $\Sigma \vdash \varphi(S^{a_1}0, \ldots, S^{a_m}0, S0)$, es decir, χ_R es representable.

Viendo el caso ahora en el que $(a_1, \ldots, a_m) \notin R$, entonces $\Sigma \vdash \neg\psi(S^{a_1}0, \ldots, S^{a_m}0)$ de donde

$$\Sigma \vdash [(\psi(S^{a_1}0, \ldots, S^{a_m}0) \wedge y = S0) \vee (\neg\psi(S^{a_1}0, \ldots, S^{a_m}0) \wedge y = 0)] \longleftrightarrow y = 0$$

y de nuevo por la definición $\Sigma \vdash \neg\varphi(S^{a_1}0, \ldots, S^{a_m}0, y) \longleftrightarrow y = S^{\chi_R(a_1,\ldots,a_m)}0 = 0$, con lo cual sustituyendo de nuevo y por el Teorema de Completitud 1.15, $\Sigma \vdash \neg\varphi(S^{a_1}0, \ldots, S^{a_m}0, 0)$, es decir, χ_R es representable. □

Teorema 3.5. *(Teorema de Representabilidad) Toda función computable $F : \mathbb{N}^m \to \mathbb{N}$ es representable con los Axiomas de Peano (es (AP)-representable). También, cada relación computable $R \subseteq \mathbb{N}^m$ es (AP)-representable.*

Demostración. Por el Lema 3.4, el caso de las relaciones queda cubierto: como uno de los axiomas de (AP) es precisamente $S0 \neq 0$, las relaciones son (AP)-representables. Tendremos que ver el caso de las funciones según las reglas de computabilidad:

- (R1) Las funciones $+ : \mathbb{N}^2 \to \mathbb{N}, \cdot : \mathbb{N}^2 \to \mathbb{N}$, $\chi_< : \mathbb{N}^2 \to \mathbb{N}$ y las funciones coordenada I_i^n $(i = 1, \ldots, n)$ son (AP)-representable.
- (R2) Dada $G : \mathbb{N}^m \to \mathbb{N}$ y $H_1, \ldots, H_m : \mathbb{N}^n \to \mathbb{N}$ (AP)-representables, también lo es la función $F = G(H_1, \ldots, H_m) : \mathbb{N}^n \to \mathbb{N}$ definida como $F(a) = G(H_1(a), \ldots, H_m(a))$.
- (R3) Si $G : \mathbb{N}^{n+1} \to \mathbb{N}$ es (AP)-representable y para todo $a \in \mathbb{N}^n$ existe $x \in \mathbb{N}$ tal que $G(a, x) = 0$, entonces la función $F : \mathbb{N}^n \to \mathbb{N}$ dada por

  $$F(a) = \mu x(G(a, x) = 0)$$

  es (AP)-representable.

Veamos cada una de estas por casos:

- (Prueba R1) Distinguimos cada caso:

  (i) La fórmula $x_1 + x_2$ representa $+ : \mathbb{N}^2 \to \mathbb{N}$ ya que si sustituyendo $a, b, c \in \mathbb{N}$ en el término se verifica $a + b = c$ entonces se cumple que $(AP) \vdash S^a + S^b = S^c$ ya que por el Lema 3.3 existe un único homomorfismo de $\mathfrak{N}$ en (AP) como $\mathbb{N}$ satisface $a + b = c$ entonces se satisface $S^a + S^b = S^c$ en cada modelo (AP).

  (ii) La fórmula $x_1 \cdot x_2$ representa $\cdot : \mathbb{N}^2 \to \mathbb{N}$ ya que si sustituyendo $a, b, c \in \mathbb{N}$ en el término se verifica $a \cdot b = c$ entonces se cumple que $(AP) \vdash S^a \cdot S^b = S^c$ por un razonamiento similar a (i).

  (iii) La fórmula $x_1 < x_2$ representa al conjunto $\{(a, b) \in \mathbb{N}^2 : a < b\}$ en (AP). Esto es porque si $a < b$ entonces $(AP) \vdash S^a < S^b$ y, por el contrario, si $a \geq b$ entonces $(AP) \vdash S^a \geq S^b$ por un razonamiento similar a (i). De este deducimos que $\chi_<$ es $(AP)-$representable.

  (iv) Para cada $n \geq 1$ y $1 \leq i \leq n$, la fórmula $t_i^n(x_1, \ldots, x_n) = x_i$ representa la función I_i^n.

- (Prueba R2) Sean $x_1, \ldots, x_n, y_1, \ldots, y_m, z$ variables y sea $G : \mathbb{N}^m \to \mathbb{N}$ (AP)-representable por una fórmula $\psi(y_1, \ldots, y_m, z)$ y sea $H_i : \mathbb{N}^m \to \mathbb{N}$ (AP)-representable por la fórmula $\varphi_i(x_1, \ldots, x_n, y_i)$ para $i = 1, \ldots, n$. Veremos que $F = G(H_1, \ldots, H_m)$ está (AP)-representada por

$$\theta(x_1, \ldots, x_n, z) := \exists y_1, \ldots, y_m((\bigwedge_{i=1}^{m} \varphi_i(x_1, \ldots, x_n, y_i)) \wedge \psi(y_1, \ldots, y_m, z)).$$

  Sea $a = (a_1, \ldots, a_n)$ y sea $c = F(a)$, tenemos que probar que $(AP) \vdash \theta(S^{a_1}0, \ldots., S^{a_m}0, z) \longleftrightarrow z = S^c0$. Sea $b_i = H_i(a)$ y escribimos $b = (b_1, \ldots, b_m)$, con esta notación $c = F(a) = G(b)$ entonces por hipótesis

$$(AP) \vdash \psi(S^b0, z) \longleftrightarrow z = S^c0 \text{ y } (AP) \vdash \varphi_i(S^a0, y_i) \longleftrightarrow y_i = S^{b_i}0,$$

  por el Teorema de Completitud 1.15 con lo que efectivamente se cumple lo buscado.

- (Prueba R3) Sea $G : \mathbb{N}^{n+1} \to \mathbb{N}$ tal que para todo $a \in \mathbb{N}^n$ existe $b \in \mathbb{N}$ con $G(a, b) = 0$. Definimos $F : \mathbb{N}^n \to \mathbb{N}$ como $F(a) = \mu b(G(a, b) = 0)$. Supongamos que G es (AP)-representada por $\varphi(x_1, \ldots, x_n, y, z)$. Veamos que la fórmula

$$\psi(x_1, \ldots, x_n, y) := \varphi(x_1, \ldots, x_n, y, 0) \wedge \forall w(w < y \to \neg\varphi(x_1, \ldots, x_n, w, 0))$$

  representa a F. Sea $a \in \mathbb{N}^n$ y sea $b = F(a)$, entonces $G(a, i) \neq 0$ para todo $i < b$ y $G(a, b) = 0$. Por ello, $(AP) \vdash \varphi(S^a0, S^b0, z) \longleftrightarrow z = 0$ y, para cada $i < b$, $G(a, i) \neq 0$ y $(AP) \vdash \varphi(S^a0, S^b0, z) \longleftrightarrow z = S^{G(a,i)}0$. Con esto, se cumple que la fórmula ψ representa a F ya que por el Teorema de Completitud 1.15 se satisface $(AP) \vdash \psi(S^a, y) \longleftrightarrow y = S^b0$.

□

Capítulo 4

Numeración de Gödel

Decimos que una $\mathfrak{L}$-teoría T está cerrada bajo demostrabilidad si cumple que si $T \vdash \sigma$ entonces $\sigma \in T$. Con esto, el concepto de teoría presenta una dualidad a la hora de formularse de una manera más práctica:

- Dado un conjunto Σ de $\mathfrak{L}$-sentencias, podemos definir la teoría como el conjunto $\Sigma^{\vdash} = \{\sigma : \Sigma \vdash \sigma\}$. Nos podemos referir a este conjunto como los Teoremas de Σ. Además, por el curso de Lógica Matemática sabemos que $\Sigma^{\vdash}$ está cerrada bajo demostrabilidad.
- Por otra parte, dada una $\mathfrak{L}$-estructura $\mathcal{A}$, obtenemos el conjunto llamado Teoría de $\mathcal{A}$ como $\mathrm{Th}(\mathcal{A}) = \{\sigma : \mathcal{A} \models \sigma\}$. Observamos además que esta definición hace a la teoría de $\mathcal{A}$ sea completa.

Lo que nosotros tratamos de buscar ahora es probar que la teoría de la arirmética de Peano, (AP), es incompleta. Pero para ello, nos encontramos con el problema de lo que, a priori, puede parecer la baja expresividad de este lenguaje. En los capítulos anteriores hemos definido lo que es «computable» para nosotros y también hemos probado que todo lo computable se puede representar como una (AP)-fórmula gracias al Teorema de Representabilidad (3.5). Así, las principales cuestiones del Teorema de Incompletitud quedan encerradas en el campo del lenguaje de la aritmética. Solo nos queda resolver este problema desde el marco propio de la aritmética.

Sin embargo, hablar de conceptos metalógicos como la incompletitud parece quedar lejos de la capacidad expresiva de las operaciones y relaciones más triviales, es por eso que todavía queda un paso más, un nivel más profundo al que descender, y es el de ser capaces de codificar cada fórmula lógica de la aritmética como un número biunívocamente dado que esto nos permitirá, solo mediante los números naturales, expresar todas las fórmulas lógicas expresables en nuestro lenguaje.

Es esto lo que denominamos codificación de Gödel o numeración de Gödel. Para ello, necesitaremos definir lo que son los números simbólicos y recordar los números secuenciales proporcionados por la función β de Gödel en 2.30. Recordamos que denotamos por $L = \{0, S, +, \cdot, <\}$ al lenguaje de la aritmética.

Definición 4.1. (Número simbólico) Definimos para cada símbolo

$$s \in L \cup \{=, \neg, \wedge, \vee, \exists, \forall\} \cup \{x_0, x_1, x_2, \ldots\}$$

su número simbólico SN como:

- $\mathrm{SN}(s) = 2i$ si $s = x_i$.
- $\mathrm{SN}(s)$ es un número impar natural distinto para cada símbolo en $L \cup \{=, \neg, \wedge, \vee, \exists, \forall\}$.

Ejemplo 4.2. Una asignación de números simbólicos válidos sobre L sería asignar a cada variable un número par, asignar $1, 3, 5, 7, 9, 11$ a $=, \neg, \wedge, \vee, \exists, \forall$ respectivamente y, para los símbolos propios de L, asignamos 13 a 0, 15 a S, 17 a $+$, 19 a $\cdot$ y 21 a $<$.

Definición 4.3. (Número de Gödel) Para un L-término t, definimos el número de Gödel $\ulcorner t \urcorner$ recursivamente como

$$\ulcorner t \urcorner = \begin{cases} \langle SN(c) \rangle & \text{si } t = c \\ \langle SN(x_i) \rangle & \text{si } t = x_i \\ \langle SN(F), \ulcorner t_1 \urcorner, \ldots, \ulcorner t_n \urcorner \rangle & \text{si } t = F(t_1, \ldots, t_n) \end{cases}$$

Para una L-fórmula φ definimos su número de Gödel $\ulcorner \varphi \urcorner$ recursivamente como

$$\ulcorner \varphi \urcorner = \begin{cases} \langle \mathrm{SN}(=), \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner \rangle & \text{si } \varphi = (t_1 = t_2) \\ \langle \mathrm{SN}(R), \ulcorner t_1 \urcorner, \ulcorner t_2 \urcorner, \ldots, \ulcorner t_n \urcorner \rangle & \text{si } \varphi = R(t_1, t_2, \ldots, t_n) \\ \langle \mathrm{SN}(\neg), \ulcorner \psi \urcorner \rangle & \text{si } \varphi = \neg\psi \\ \langle \mathrm{SN}(\vee), \ulcorner \varphi_1 \urcorner, \ulcorner \varphi_2 \urcorner \rangle & \text{si } \varphi = (\varphi_1 \vee \varphi_2) \\ \langle \mathrm{SN}(\wedge), \ulcorner \varphi_1 \urcorner, \ulcorner \varphi_2 \urcorner \rangle & \text{si } \varphi = (\varphi_1 \wedge \varphi_2) \\ \langle \mathrm{SN}(\exists), \ulcorner x \urcorner, \ulcorner \psi \urcorner \rangle & \text{si } \varphi = \exists x \psi \\ \langle \mathrm{SN}(\forall), \ulcorner x \urcorner, \ulcorner \psi \urcorner \rangle & \text{si } \varphi = \forall x \psi \end{cases}$$

Ejemplo 4.4. Consideramos la asignación de números simbólicos dada por el Ejemplo 4.2. Veamos como se desarrollaría el número de Gödel hasta su representación secuencial del primer axioma de Peano, que podemos expresar como $\forall x_1(\neg(Sx_1 = 0))$.

$$\ulcorner \forall x_1(\neg(Sx_1 = 0)) \urcorner = \langle \mathrm{SN}(\forall), \ulcorner x_1 \urcorner, \ulcorner (\neg(Sx_1 = 0)) \urcorner \rangle = \langle \mathrm{SN}(\forall), \mathrm{SN}(x_1), \langle \mathrm{SN}(\neg), \ulcorner (Sx_1 = 0)) \urcorner \rangle \rangle =$$

$$= \langle \mathrm{SN}(\forall), \mathrm{SN}(x_1), \langle \mathrm{SN}(\neg), \langle \mathrm{SN}(=), \ulcorner Sx_1 \urcorner, \ulcorner 0 \urcorner \rangle \rangle \rangle =$$

$$= \langle \mathrm{SN}(\forall), \mathrm{SN}(x_1), \langle \mathrm{SN}(\neg), \langle \mathrm{SN}(=), \langle \mathrm{SN}(S), \mathrm{SN}(x_1) \rangle, \langle \mathrm{SN}(0) \rangle \rangle \rangle \rangle = \langle 11, 2, \langle 3, \langle 1, \langle 15, 2 \rangle, \langle 13 \rangle \rangle \rangle \rangle.$$

Observación. Sabemos por el Lema 2.31 que la función $a \mapsto \langle a \rangle$ es computable y la numeración de Gödel para una fórmula hace uso de esta función por lo que toda $L-$fórmula o $L-$término dentro del lenguaje de la aritmética, siguiendo un proceso similar al ejemplo anterior, lleva a una expresión que

es computable. Esto se aplica, por ejemplo, a los axiomas de igualdad, tautologías y otras $L-$fórmulas o $L-$términos que sigan la estructura presente en la Definición 4.3.

De esta manera tenemos que cada fórmula se expresa como un número secuencial de forma única por el Teorema 2.28, expresándose como el número secuencial obtenido de la secuencia dada por los números simbólicos que aparezcan y los términos y fórmulas que aparezcan en ella recursivamente. Esta es la principal virtud de la numeración de Gödel. No es tan importante saber qué número codifica cada fórmula, sino tener en cuenta la propiedad de que cada fórmula puede ser representada como consecuencia de la función β de Gödel como un número natural de forma unívoca. En el siguiente Lema veremos cómo algunos subconjuntos de $\mathbb{N}$ caracterizados por la numeración de Gödel son computables:

Lema 4.5. *Los siguientes subconjuntos de* $\mathbb{N}$ *son computables:*

1. Vble :=$\{\ulcorner x \urcorner : x$ *es una variable*$\}$
2. Term :=$\{\ulcorner t \urcorner : t$ *es un L-término*$\}$
3. Afor :=$\{\ulcorner \varphi \urcorner : \varphi$ *es una L-fórmula atómica*$\}$
4. For :=$\{\ulcorner \varphi \urcorner : \varphi$ *es un L-fórmula*$\}$

Demostración. Realizaremos cada uno de los casos.

1. Si $a \in$ Vble es porque $a = \langle 2b \rangle$ para algún $b < a$. Y sabemos que $a \mapsto \langle a \rangle$ es computable por el Lema 2.31.
2. Si $a \in$ Term entonces hay tres opciones. Si $a \in$ Vble el resultado se obtiene como en 1. Si $a = \langle SN(c) \rangle$ donde c es un símbolo de constante, es computable pues $SN(c)$ es un número impar y por el Lema 2.31 sabemos que la función $a \mapsto \langle a \rangle$ es computable. Finalmente, si $a = \langle SN(F), \ulcorner t_1 \urcorner, \ldots, \ulcorner t_n \urcorner \rangle$ dónde F es un símbolo de función del lenguaje L y $t_1, \ldots, t_n$ $L-$términos con números de Gödel menores que a, luego es computable también porque lo es la función $a \mapsto \langle a \rangle$ por el Lema 2.31 y se invoca a sí misma sobre números menores por lo que podemos hacer uso de la Proposición 2.34.
3. Si $a \in$ Afor entonces hay dos opciones para la fórmula atómica que codifica, o es de la forma $(t_1 = t_2)$ o es de la forma $R(t_1, \ldots, t_n)$ donde R es una relación n-aria. En cualquier caso se resuelve haciendo uso del apartado 2 sobre la codificación de términos también por la Proposición 2.34 y el Lema 2.31 como se indica en el apartado 2.
4. Si $a \in$ For entonces todas las opciones para la fórmula se pueden demostrar teniendo en cuenta el apartado 2 y el 3 sobre la codificación de términos y fórmulas junto con la Proposición 2.34 y el Lema 2.31 como se indica en el apartado 2.

□

Gracias a esta numeración podemos definir ahora las siguientes funciones que serán necesarias para llevar a cabo la demostración. La primera es la función Sub, que se comportará dentro del margen de la codificación de Gödel como lo haría la sustitución en términos y fórmulas. El objetivo de esta será poder representar aritméticamente esta operación lógica:

Proposición 4.6. *La función* $Sub : \mathbb{N}^3 \to \mathbb{N}$ *definida recursivamente*

$$Sub(a,b,c) = \begin{cases} c & \text{Si } a = b \text{ y } a \text{ es el número de Gödel de una variable} \\ \langle a_0, Sub(a_1,b,c), \ldots, Sub(a_n,b,c)\rangle & \text{Si } a = \langle a_0,\ldots,a_n\rangle \text{ con } n>0 \text{ y } SN(\exists) \neq a_0 \neq SN(\forall) \\ \langle SN(\exists), a_1, Sub(a_2,b,c)\rangle & \text{Si } a = \langle SN(\exists), a_1, a_2\rangle \text{ y } a_1 \neq b \\ \langle SN(\forall), a_1, Sub(a_2,b,c)\rangle & \text{Si } a = \langle SN(\forall), a_1, a_2\rangle \text{ y } a_1 \neq b \\ a & \text{En otro caso} \end{cases}$$

es computable y satisface que para t *término y* φ *fórmula*

$$Sub(\ulcorner t\urcorner, \ulcorner x\urcorner, \ulcorner \tau\urcorner) = \ulcorner t(\tau/x)\urcorner \text{ y } Sub(\ulcorner \varphi\urcorner, \ulcorner x\urcorner, \ulcorner \tau\urcorner) = \ulcorner \varphi(\tau/x)\urcorner.$$

donde $\varphi(\tau/x)$ *denota la sustitución de* τ *por* x *en la fórmula* φ.

Demostración. El hecho de que sea computable se desprende de la propiedad de la definición por casos del Lema 2.12 y de que la función β lo es por el Teorema 2.28 para los casos 2, 3 y 4. Para los casos 1 y 5 basta con ver que es equivalente a la función coordenada que es, por (R1), computable.

Para la segunda parte procederemos por inducción sobre la complejidad del término y la fórmula. Distinguimos los casos:

- El caso $t = \alpha$ constante se deduce por

$$\ulcorner t(\tau/x)\urcorner = \ulcorner \alpha(\tau/x)\urcorner = \ulcorner \alpha\urcorner = \mathrm{Sub}(\ulcorner \alpha\urcorner, \ulcorner x\urcorner, \ulcorner \tau\urcorner).$$

- Para el caso en el que $a = b$ y a es el número de Gödel una variable (esto se da cuando $a = \langle 2a_0\rangle$ para algún $a_0 < a$), tenemos que solo puede ser $t = x$, luego

$$\ulcorner t(\tau/x)\urcorner = \ulcorner x(\tau/x)\urcorner = \ulcorner \tau\urcorner = \mathrm{Sub}(\ulcorner t\urcorner, \ulcorner x\urcorner, \ulcorner \tau\urcorner).$$

- Para el caso en el que $a = \langle a_0,\ldots,a_n\rangle$ con $n > 0$ y $SN(\exists) \neq a_0 \neq \mathrm{SN}(\forall)$ entonces podemos tomar aquí las expresiones como $\varphi = (t_1 = t_2)$, $\varphi = R(t_1,\ldots,t_n)$, $\varphi = \neg\psi$, $\varphi = \psi \wedge \chi$, y $t = F(t_1,\ldots,t_n)$. Todos estos casos se resuelven de forma análoga, usando la hipótesis

de inducción sobre las sustituciones previas. Veamos el caso $\varphi = R(t_1, \ldots, t_n)$:

$$\ulcorner \varphi(\tau/x) \urcorner = \ulcorner R(t_1, \ldots, t_n)(\tau/x) \urcorner = \ulcorner R(t_1(\tau/x), \ldots, t_n(\tau/x)) \urcorner = \langle \mathrm{SN}(R), \ulcorner t_1(\tau/x) \urcorner, \ldots, \ulcorner t_n(\tau/x) \urcorner \rangle \underset{=}{H.I.}$$

$$\underset{=}{H.I.} \langle \mathrm{SN}(R), \mathrm{Sub}(\ulcorner t_1 \urcorner, \ulcorner x \urcorner, \ulcorner \tau \urcorner), \ldots, \mathrm{Sub}(\ulcorner t_n \urcorner, \ulcorner x \urcorner, \ulcorner \tau \urcorner)) \rangle = \mathrm{Sub}(\ulcorner R(t_1, \ldots, t_n) \urcorner, \ulcorner x \urcorner, \ulcorner \tau \urcorner) =$$

$$= \mathrm{Sub}(\ulcorner \varphi \urcorner, \ulcorner x \urcorner, \ulcorner \tau \urcorner).$$

- Los casos $a = \langle \mathrm{SN}(\exists), a_1, a_2 \rangle$ y $a_1 \neq b$ y $a = \langle \mathrm{SN}(\forall), a_1, a_2 \rangle$ y $a_1 \neq b$ son análogos, siendo las únicas configuraciones posibles $\varphi = \exists y \psi$ y $\varphi = \forall y \psi$ respectivamente. Probaremos el del primer caso ya que el segundo es análogo, teniendo en cuenta que por hipótesis $x \neq y$:

$$\ulcorner \varphi(\tau/x) \urcorner = \ulcorner (\exists y \psi)(\tau/x) \urcorner = \ulcorner \exists y (\psi(\tau/x)) \urcorner = \langle \mathrm{SN}(\exists), \ulcorner y \urcorner, \ulcorner \psi(\tau/x) \urcorner \rangle \underset{=}{H.I.}$$

$$\underset{=}{H.I.} \langle \mathrm{SN}(\exists), \ulcorner y \urcorner, \mathrm{Sub}(\ulcorner \psi \urcorner, \ulcorner x \urcorner, \ulcorner \tau \urcorner) \rangle = \mathrm{Sub}(\ulcorner \exists y \psi \urcorner, \ulcorner x \urcorner, \ulcorner \tau \urcorner) = \mathrm{Sub}(\ulcorner \varphi \urcorner, \ulcorner x \urcorner, \ulcorner \tau \urcorner).$$

□

Corolario 4.7. *La función Sub es representable.*

Demostración. Se obtiene de que es computable por la Proposición 4.6 y entonces por el Teorema de Representabilidad 3.5 es representable. □

También podemos definir la función para identificar los números naturales según la función sucesor:

Proposición 4.8. *La función Num :* $\mathbb{N} \to \mathbb{N}$ *definida como* $Num(a) = \ulcorner S^a 0 \urcorner$ *es computable.*

Demostración. Se puede escribir $\mathrm{Num}(0) = \ulcorner 0 \urcorner$ y $\mathrm{Num}(a+1) = \langle \mathrm{SN}(S), \mathrm{Num}(a) \rangle$, que sabemos son computables porque la función $a \mapsto \langle a \rangle$ es computable Lema 2.31. □

Corolario 4.9. *La función Num es representable.*

Demostración. Se obtiene de que es computable por la Proposición 4.8 y entonces por el Teorema de Representabilidad 3.5 es representable. □

Podemos combinar estas dos funciones para definir la siguiente función que será útil en la demostración del Teorema de Incompletitud.

Proposición 4.10. *Dada una L-fórmula* $\varphi(x, y)$*, la función*

$$(a, b) \longmapsto Sub(a, \ulcorner x \urcorner, Num(b))$$

es computable.

Demostración. El resultado se da porque las funciones Sub y Num son computables por la Proposición 4.6 y la Proposición 4.8. son computables junto con que también lo es la función $a \mapsto \langle a \rangle$ por el Lema 2.31. □

Por último, necesitamos definir el conjunto de demostraciones de un conjunto de sentencias.

Definición 4.11. Sea Σ un conjunto de $L-$sentencias. Definimos

$$\ulcorner \Sigma \urcorner = \{\ulcorner \sigma \urcorner : \sigma \in \Sigma\}$$

y decimos que Σ es computable si $\ulcorner \Sigma \urcorner$ es computable.

Observación. Con esta definición entonces podemos decir que (AP) es computable.

Definición 4.12. El conjunto Prf_Σ de números de Gödel de las demostraciones de Σ es

$$\mathrm{Prf}_\Sigma = \{\langle \ulcorner \varphi_1 \urcorner, \ldots, \ulcorner \varphi_n \urcorner \rangle : \varphi_1, \ldots, \varphi_n \text{ es una demostración de } \Sigma\}.$$

Proposición 4.13. *Si Σ es computable, entonces Prf_Σ también es computable.*

Demostración. Un número natural $a \in \mathrm{Prf}_\Sigma$ si y solo si se da que $Seq(a)$ y $\mathrm{long}(a) \neq 0$, y además para cada $i < \mathrm{long}(a)$ el elemento $\beta(a, i)$ corresponde a una fórmula φ_i siguiendo una de las siguientes posibilidades.

- Es una tautología, un axioma de igualdad, una sentencia de Σ o una instancia de $\exists$-introducción.
- Existen $j, k < i$ tales que la fórmula φ_i se deduce de las fórmulas correspondientes a $\beta(a, j)$ y $\beta(a, k)$ mediante Modus Ponens.
- Existe $j < i$ tal que φ_i se deduce de la fórmula correspondiente a $\beta(a, j)$ mediante el axioma de $\exists$-sustitución.

Sabemos por hipótesis que Σ es computable y también por el lema 2.33 que pertenecer a Seq es computable. Para los casos restantes tenemos que su numeración de Gödel es computable pues la función $a \mapsto \langle a \rangle$ lo es por el lema 2.31 y debemos aplicar inducción sobre la complejidad de la fórmula teniendo en cuenta la notación dada anteriormente para la implicación $(\varphi_j \to \varphi_i) = \neg(\varphi_j \wedge \neg\varphi_i)$. □

Corolario 4.14. *Prf_Σ es representable.*

Demostración. Se obtiene de que es computable por la Proposición 4.13 y entonces por el Teorema de Representabilidad 3.5 es representable. □

Capítulo 5

El Teorema de Incompletitud

En este capítulo probaremos, finalmente, el Teorema de Incompetitud de Gödel.

Con todas las herramientas que hemos estado desarrollando hasta ahora seremos capaces de llevar a cabo esta demostración. Tomaremos en lo que sigue un lenguaje aritmético extendido finito, es decir, $L_{ext} \supseteq L = \{0, S, +, \cdot, <\}$ finito y Σ un conjunto de L_{ext}-sentencias. Esencialmente, el Teorema de Incompletitud nos dice que para este conjunto de sentencias, si es consistente y computable, siempre será incompleto.

Para probar esto, Gödel planteó la siguiente idea basada en crear una fórmula autorreferencial. ¿Qué pasaría si pudieramos encontrar una fórmula que afirmase «yo no soy demostrable»? Si la afirmación es cierta, no es demostrable y, por tanto, no podría ser una teoría completa. Si la afirmación fuese falsa, tendría que ser demostrable, pero entonces sería cierto; alcanzanado así una contradicción. Parece obvio entonces que si somos capaces de encontrar dicha fórmula, el teorema cae por su propio peso. Sin embargo, encontrar esta fórmula no es para nada trivial y es gracias a las propiedades de computabilidad, el Teorema de Representabilidad y la numeración de Gödel; que dentro del marco de la aritmética vamos a poder establecer una traducción directa de estas ideas a la lógica de primer orden.

Antes de probar el Teorema de Incompletitud, necesitaremos el Lema del Punto Fijo que nos dará la primera aproximación a esta fórmula deseada.

Lema 5.1. *(Lema del Punto Fijo) Supongamos un conjunto de L_{ext}-sentencias $\Sigma \supseteq (AP)$. Entonces para cada L_{ext}-fórmula $\rho(y)$ existe una L_{ext}-sentencia σ tal que $\Sigma \vdash \sigma \leftrightarrow \rho(S^{\ulcorner \sigma \urcorner}0)$.*

Demostración. Por la Proposición 4.10 la función $(a, b) \longmapsto \mathrm{Sub}(a, \ulcorner x \urcorner, \mathrm{Num}(b))$ es computable. Luego, por el Teorema de Representabilidad (3.5), es Σ-representable. Sea $\mathrm{sub}(x_1, x_2, y)$ la fórmula que la representa. Sea ahora x una variable que no aparece en $\mathrm{sub}(x_1, x_2, y)$. Se tiene entonces que para todo $a, b \in \mathbb{N}$:

$$\Sigma \vdash sub(S^a 0, S^b 0, y) \leftrightarrow y = S^c 0 \tag{5.0.1}$$

donde $c = \mathrm{Sub}(a, \ulcorner x \urcorner, \mathrm{Num}(b))$. Sea ahora $\rho(y)$ una L_{ext}-fórmula. Definimos $\theta(x) := \exists y(\mathrm{sub}(x, x, y) \wedge$

$\rho(y)$) y sea $m = \ulcorner\theta\urcorner$. Si tomamos ahora $\sigma := \theta(S^m0/x) = \theta(S^m0)$ y $n = \ulcorner\sigma\urcorner$. Probamos que, con esta construcción, $\Sigma \vdash \sigma \leftrightarrow \rho(S^n0)$.

Tenemos que, usando el Lema 4.6

$$n = \ulcorner\sigma\urcorner = \ulcorner\theta(S^m0/x)\urcorner = \mathrm{Sub}(\ulcorner\theta\urcorner, \ulcorner x\urcorner, \ulcorner S^m0\urcorner) = \mathrm{Sub}(m, \ulcorner x\urcorner, \mathrm{Num}(m)).$$

Entonces, por (5.0.1),

$$\Sigma \vdash \mathrm{sub}(S^m0, S^m0, y) \leftrightarrow y = S^n0.$$

Por otro lado, tenemos que

$$\sigma = \theta(S^m0) = \exists y(\mathrm{sub}(S^m0, S^m0, y) \wedge \rho(y)).$$

Juntando todo esto tenemos que, sustituyendo y por el Teorema de Completitud (1.15),

$$\Sigma \vdash \sigma \leftrightarrow \exists y(y = S^n0 \wedge \rho(y)),$$

es decir, $\Sigma \vdash \sigma \leftrightarrow \rho(S^n0)$. □

Teorema 5.2. *(Teorema de Incompletitud de Gödel) Supongamos un conjunto de L_{ext}-sentencias $\Sigma \supseteq (AP)$ consistente y computable. Entonces existe una L-fórmula $\varphi(x)$ tal que $(AP) \vdash \varphi(S^m0)$ para cada $m \in \mathbb{N}$, pero $\Sigma \nvdash \forall x\varphi(x)$.*

Demostración. Consideremos la relación Pr_Σ 2-aria definida como $\mathrm{Pr}_\Sigma(m, n)$ ocurre si m es el número de Gödel de la demostración de una $L_{ext}-$sentencia con número de Gödel n en Σ.

Como Σ es computable, por la Proposición 4.13 sabemos entonces que Pr_Σ es computable. Así por el Teorema de Representabilidad (3.5), Pr_Σ es representable en (AP). Sea $\mathrm{pr}_\Sigma(x, y)$ la L-formula que la representa en (AP), y por tanto en Σ. Como por hipótesis Σ es consistente, tenemos que para todo $m, n \in \mathbb{N}$:

$$\mathrm{Pr}_\Sigma(m, n) \text{ implica que } \Sigma \vdash \mathrm{pr}_\Sigma(S^m0, S^n0)$$

$$\neg\mathrm{Pr}_\Sigma(m, n) \text{ implica que } \Sigma \vdash \neg\mathrm{pr}_\Sigma(S^m0, S^n0)$$

Consideremos ahora la L-fórmula $\rho(y) = \forall x\neg\mathrm{pr}_\Sigma(x, y)$. Por el Lema del Punto Fijo (5.1), obtenemos que existe una L-sentencia σ tal que $(AP) \vdash \sigma \leftrightarrow \rho(S^{\ulcorner\sigma\urcorner}0)$. Como $(AP) \subseteq \Sigma$, deducimos que $\Sigma \vdash \sigma \leftrightarrow \rho(S^{\ulcorner\sigma\urcorner}0)$, es decir,

$$\Sigma \vdash \sigma \leftrightarrow \forall x\neg\mathrm{pr}_\Sigma(x, S^{\ulcorner\sigma\urcorner}0) \tag{5.0.2}$$

Probamos ahora el siguiente aserto: $\Sigma \nvdash \sigma$. Supongamos por reducción al absurdo que $\Sigma \vdash \sigma$ y sea m el número de Gödel de la demostración de σ en Σ. De esta forma, se satisface la relación $\mathrm{Pr}_\Sigma(m, \ulcorner\sigma\urcorner)$. Entonces, por la expresión (5.0.2) tenemos que $\Sigma \vdash \forall x\neg\mathrm{pr}_\Sigma(x, S^{\ulcorner\sigma\urcorner}0)$. Por el Teorema de Completitud 1.15 y sustituyendo en S^m0 tenemos que $\Sigma \vdash \neg\mathrm{pr}_\Sigma(S^m0, S^{\ulcorner\sigma\urcorner}0)$. Por otro lado, si $\mathrm{Pr}_\Sigma(m, S^{\ulcorner\sigma\urcorner}0)$

entonces obtenemos $\Sigma \vdash \mathrm{pr}_\Sigma(S^m 0, S^{\ulcorner \sigma \urcorner} 0)$, lo cual implica que Σ es inconsistente, contradicción.

Con esto se prueba el aserto. Tomemos entonces $\varphi(x) := \neg \mathrm{pr}_\Sigma(x, S^{\ulcorner \sigma \urcorner} 0)$ como la fórmula que buscamos. Se cumple, pues,

(i) $(AP) \vdash \varphi(S^m 0)$ para cada m. Como $\Sigma \nvdash \sigma$, no existe m que sea el número de Gödel de la prueba de σ en Σ. Así $\neg \mathrm{Pr}_\Sigma(m, \ulcorner \sigma \urcorner)$ para cada m, lo que implica que $\Sigma \vdash \neg \mathrm{pr}_\Sigma(S^m 0, S^{\ulcorner \sigma \urcorner} 0)$ ya que Pr_Σ es $(AP)-$representable. Es decir, $(AP) \vdash \varphi(S^m 0)$.

(ii) $\Sigma \nvdash \forall x \varphi(x)$. Esto se obtiene por Modus Ponens de que $\Sigma \nvdash \sigma$ y (5.0.2).

□

Con esto queda probado el Teorema de Incompletitud, siendo esta fórmula φ la que estamos buscando. Haciendo un análisis literal de su construcción, $\varphi(x)$ afirmaría que x no es número de Gödel de la demostración en Σ de la fórmula σ. Al demostrar que $(AP) \vdash \varphi(S^m 0)$ para cada m estamos expresando que la aritmética de Peano demuestra que m no codifica ninguna demostración para σ, luego sería cierto que ningún valor codifica una demostración para σ. Por otro lado, que $\Sigma \nvdash \forall x \varphi(x)$ nos dice que en una extensión finita de la aritmética, no se puede demostrar precisamente esta propiedad.

Es decir, que la aritmética de Peano es incompleta.

Bibliografía

[1] Lou van den Dries. *Mathematical Logic.* 2019.

[2] Daniel Palacín. *Apuntes de Lógica Matemática.* Curso 2020-2021.

[3] Martin Hils, François Loeser. *A First Journey through Logic. 2019.*

[4] Universidad de Sevilla. *Lógica Matemática.* 2011-2012.

[5] Anatolli Grinsphan. Discrete Mathematics. https://www.math.drexel.edu/ tolya/cantorpairing.pdf