Algunas reflexiones y comentarios sobre los teoremas incompletitud de Gödel

J. M. Almira

Universidad de Córdoba
2 de Noviembre de 2022.

El contexto histórico

El contexto histórico

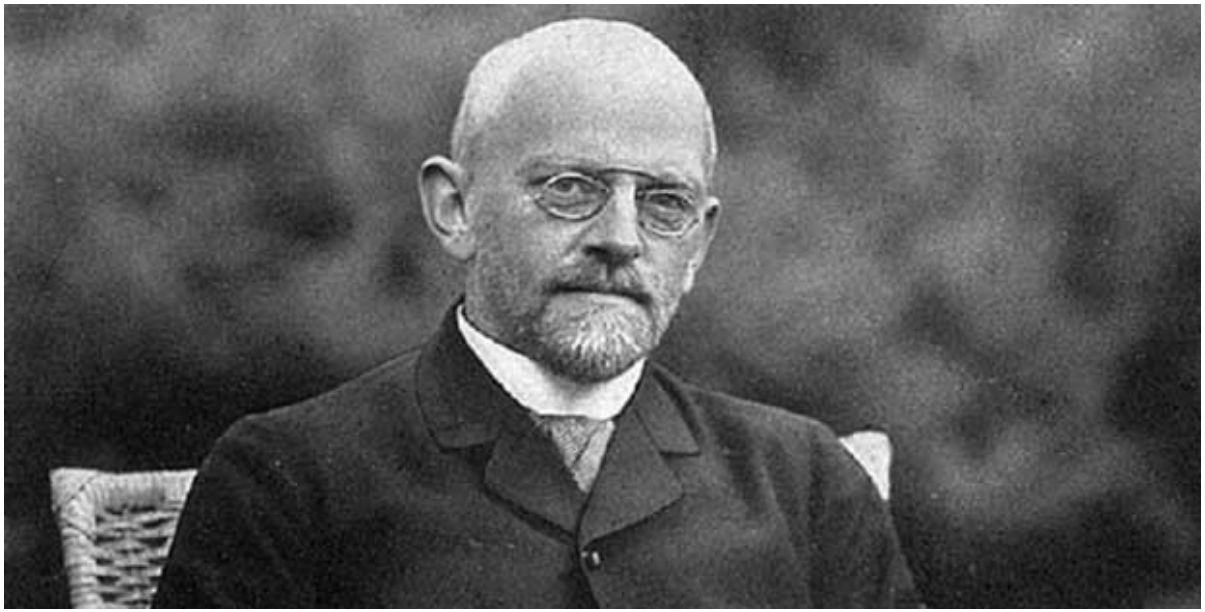
- A finales del siglo XIX, con la Teoría (intuitiva o ingenua) de conjuntos (TC) y con la Aritmetización del Análisis, se había creado una herramienta muy potente que permitía dar fundamento aparentemente sólido a todas las matemáticas conocidas.
- Simultáneamente, el descubrimiento y desarrollo de las Geometrías no-Euclídeas abrió la puerta a la existencia de teorías que describen "realidades" contradictorias y que sin embargo son consistentes (o, en todo caso, relativamente consistentes)
- Esto significa, entre otras cosas, que la propia consistencia de teorías aparentemente bien fundadas, como la aritmética, la geometría plana, etc., es por sí misma un problema no trivial.

El contexto histórico

- Aparecen, de pronto, varias paradojas en el seno de la TC -cuando sus métodos e ideas ya habían calado fuertemente en matemáticas.
- Surgen. por tanto, varias escuelas de pensamiento (Conjustistas, Logicistas, Formalistas, Intuicionistas, etc.) que intentan resolver las paradojas y que fomentan un pensamiento de corte filosófico sobre la naturaleza, objetivo y métodos de las matemáticas.

El contexto histórico

- Uno de los líderes de una de estas nuevas escuelas (el Formalismo) fue D. Hilbert.



El contexto histórico

- Hilbert, tras la publicación por parte de H. Weyl (su mejor alumno) de un texto dedicado a abordar el continuo numérico de forma constructiva, tuvo miedo de que este arrastrara hacia los postulados constructivistas de Brouwer a buena parte de los matemáticos del momento. Era un miedo fundado, pues nadie era tan persuasivo escribiendo matemáticas como Weyl. Si el constructivismo se imponía, entonces habría que renunciar a buena parte de las matemáticas que tanto amaba y, en particular, a muchos de sus propios resultados -y no podía consentirlo. (Desde el inicio de su carrera, cuando resolvió el Problema de Gordan, los métodos usados por Hilbert fueron altamente no constructivos, y utilizaban frecuentemente la TC)
- Por tanto, Hilbert decide iniciar una campaña en favor del formalismo, apoyada por un programa (un proyecto de investigación) dedicado a dar un fundamento constructivo de los métodos habituales en TC. Es decir: quería obtener una demostración constructiva de que la TC formalizada (y en particular, la aritmética formalizada) es una teoría consistente.

El contexto histórico

Programa de Hilbert: objetivos

- Definir, por la vía de una axiomatización apropiada, un *sistema formal* que, con una *interpretación* adecuada, permita expresar todas las matemáticas que existen (en particular, la aritmética y la teoría de conjuntos) Esto se había conseguido ya con la obra *Principia Mathematica* de Russell y Whitehead y con las distintas propuestas de axiomáticas para la TC (ZF, ZF+Choice, etc.) y la aritmética (Peano (para $\mathbb{N}$), Hilbert (para $\mathbb{R}$), etc.)
- Demostrar, con los métodos que permite el sistema formal así definido, la consistencia de dicho sistema formal. Esta prueba sería admitida por la escuela intuicionista y, por tanto, permitiría el uso libre de las herramientas transfinitas de TC.

La idea es que, al reducir todo a símbolos aparentemente despojados de significado, lograríamos evitar las complejidades que en TC surgen con el uso del infinito, con una prueba de tipo combinatorio de la consistencia de las matemáticas. Además, la existencia de diversas interpretaciones para un mismo sistema formal se veía como una ventaja.

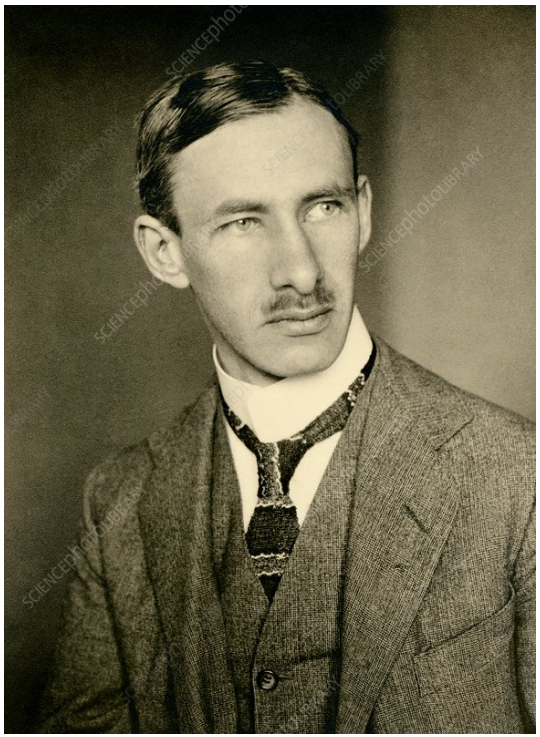
El contexto histórico

La propuesta de Hilbert era abordar el problema de la consistencia por etapas. En una primera etapa se debe ofrecer una demostración absoluta de la consistencia de, por ejemplo, una parte de la lógica (en este caso, la lógica de enunciados). Después se abordaría, en etapas consecutivas, la consistencia (relativa a la consistencia probada en una etapa anterior) de partes cada vez más amplias de la lógica y las matemáticas.

- Lógica de enunciados
- Una forma rudimentaria de la Aritmética elemental
- Aritmética elemental
- Inferencias transfinitas y algunas partes del Análisis
- etc.

El contexto histórico

- El colaborador más importante de Hilbert para el desarrollo de su programa fue P. Bernays. (Aunque también cabe mencionar a W. Ackermann).



HILBERT, BERNAYS, Grundlagen der Mathematik I,II, Springer, 1934,1939.
HILBERT, ACKERMANN, Grundzüge der theoretischen Logik, 1928.

El contexto histórico

- En 1918 Hilbert y Bernays demuestran la consistencia y la completitud semántica de la lógica de enunciados.
- En 1928 Hilbert y Ackermann formulan el "Problema de la decisión"
- En 1930 K. Gödel demuestra que el Cálculo de Predicados de primer orden es semánticamente completo (ya se sabía del calculo de enunciados, pero este otro resultado es mucho más complicado). Se asume que esto representa un impulso fundamental para el avance del Programa de Hilbert.
- En 1931, Gödel demuestra que la aritmética de Peano formalizada contiene proposiciones formalmente indecidibles, por lo que es incompleta. Es más, demuestra que es incompletable y que también es incapaz de probar su propia consistencia.
  Es un mazazo contra el Programa de Hilbert. Gödel tiene solo 25 años...

Los sistemas formales y sus propiedades

El sistema MIU

- Alfabeto: {M,I,U}
- Gramática: Todas las cadenas de letras del alfabeto se consideran palabras bien formadas.
- Axiomas:
  - A1 MI
- Reglas de inferencia:
  - R1 A partir de una cadena del tipo xI podemos pasar a una del tipo xIU
  - R2 A partir de una cadena del tipo Mx podemos pasar a una del tipo Mxx
  - R3 A partir de una cadena del tipo xIIIy podemos pasar a una del tipo xUy
  - R4 A partir de una cadena del tipo xUUy podemos pasar a una del tipo xy

Un teorema de MIU es una cadena que se puede obtener a partir de MI aplicando las reglas de inferencia R1,R2,R3,R4

Los sistemas formales y sus propiedades

Por ejemplo, MUIIU es un teorema de MIU porque podemos derivarlo con los siguientes pasos:

1. MI — Es un axioma.
2. MII — Aplicamos R2 a (1).
3. MIIII — Aplicamos R2 a (2).
4. MIIIIU — Aplicamos R1 a (3).
5. MUIU — Aplicamos R3 a (4).
6. MUIUUIU — Aplicamos R2 a (5).
7. MUIIU — Aplicamos R4 a (6).

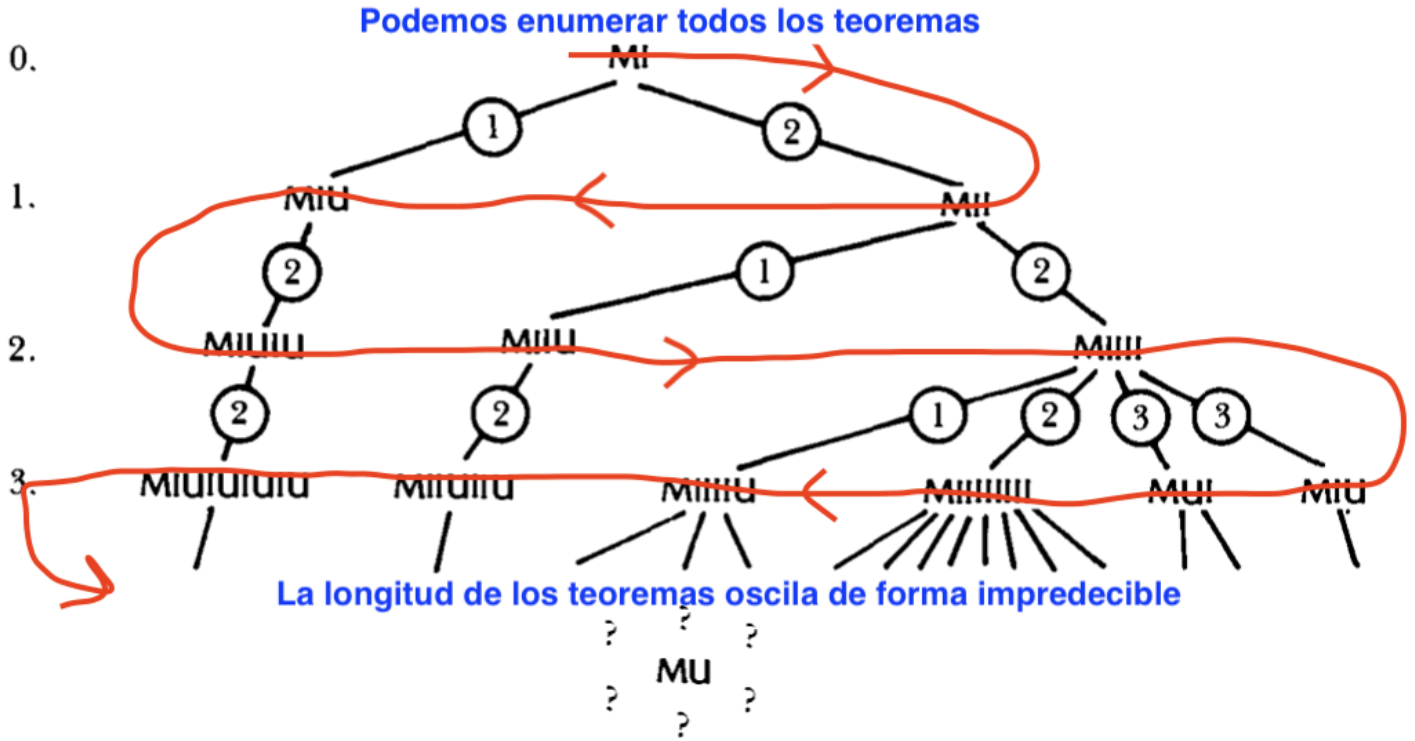
Ejemplo de **metateorema**:

"Todos los teoremas de MIU son cadenas del tipo Mx".

Pregunta: ¿Es MU un teorema de MIU?

Los sistemas formales y sus propiedades

En MIU es posible generar todos los teoremas de manera sistemática:



- Si la cadena C aparece en el grafo, es un teorema de MIU
- Las cadenas C que no están el el grafo no son teoremas de MIU. Pero ¿cómo podemos saber que una cadena no va a aparecer en el futuro, por muy grande que sea la porción de grafo que se ha examinado?

Los sistemas formales y sus propiedades

Si llamamos $n_k = n(C_k)$ al número de I's que aparecen en la k-ésima cadena de una derivación en MIU, entonces es claro que

$$n_0 = 1, n_{k+1} \in \{n_k, 2n_k, n_k - 3\}$$

Por otra parte, la cadena MU no contiene la letra "I" (por lo que $n(\text{MU}) = 0$). Si pudiéramos probar que $n_k \neq 0$ para todo k y toda derivación, veríamos que MU no es un teorema de MIU. Algunas secuencias (n_k) obtenidas de derivaciones son:

$$\begin{array}{cccccccccc}
1 & 2 & 4 & 1 & 2 & 4 & 1 & 2 & \cdots \\
1 & 2 & 4 & 8 & 5 & 2 & 4 & 8 & \cdots \\
1 & 2 & 4 & 8 & 5 & 10 & 7 & 4 & \cdots \\
1 & 2 & 4 & 8 & 5 & 10 & 20 & 17 & \cdots \\
1 & 1 & 2 & 4 & 1 & 2 & 4 & 8 & \cdots
\end{array}$$

Los sistemas formales y sus propiedades

Ahora bien, sabemos que

$$n_0 = 1 \notin 3\mathbb{N}.$$

Y que

$$\text{Si } n_k \notin 3\mathbb{N} \text{ entonces } n_{k+1} \in \{n_k, n_k - 3, 2n_k\} \subset \mathbb{N} \setminus 3\mathbb{N}.$$

De modo que

$$\{n_k\}_{k=0}^{\infty} \subseteq \mathbb{N} \setminus 3\mathbb{N}.$$

En particular,

$$\forall k \in \mathbb{N} : \ n_k \neq 0$$

¡Lo que prueba que MU no es un teorema de MIU!

Los sistemas formales y sus propiedades

El sistema formal TNT (por: "Teoría de Números Tipográfica"):

- Vocabulario Lógico:

$$\neg \quad = \quad \wedge \quad \vee \quad \rightarrow \quad \leftrightarrow \quad \forall \quad \exists \quad x \quad ' \quad :$$

- Vocabulario no lógico:

$$0 \quad S \quad + \quad \cdot$$

0 es una constante, S es una función unaria, $+$ y $\cdot$ son funciones binarias. Usamos $(a+b)$ y $(a \cdot b)$ para denotar $+(a,b)$ y $\cdot(a,b)$, respectivamente.

  - Las sentencias $0, S0, SS0, \cdots, SSS\cdots S0, \cdots$ están bien formadas y denotan los numerales $0, 1, 2, 3, \cdots, n, \cdots$.
  - El signo x denota una variable. Hay una sucesión infinita de variables que se pueden usar y todas surgen a partir de x como $x, x', x'', x''', \cdots$.

Los sistemas formales y sus propiedades

Términos:

- 0 (así como cualquier constante) son términos.
- Si σ, τ son términos, también lo son $S\sigma$, $(\sigma + \tau)$ y $(\sigma \cdot \tau)$.
- Ninguna otra cadena es un término.

Fórmulas bien formadas (WFF)

- Las únicas WFF atómicas son las expresiones del tipo $\sigma = \tau$ donde σ, τ son términos.
- El resto de WFF se obtienen a partir de las WFF atómicas aplicando de forma usual los conectores, cuantificadores, etc.

Nota: Llamamos L_A al lenguaje que acabamos de definir para TNT.
Una sentencia del tipo $\exists x A(x)$ es cierta si para algún número natural n se tiene que $A(\overline{n})$ es cierta, donde $\overline{n} = \mathrm{SS}\cdots\mathrm{S0}$, con n apariciones de S.
Además, $\forall x A(x)$ es cierta si $A(\overline{n})$ es cierta para todo número natural n.

Los sistemas formales y sus propiedades

Axiomas de TNT

- $\forall x : \neg Sx = 0$
- $\forall x : (x + 0) = x$
- $\forall x : \forall x' : (x + Sx') = S(x + x')$
- $\forall x : \forall x' : (x \cdot Sx') = ((x \cdot x') + x)$

Los sistemas formales y sus propiedades

Ejemplos de enunciados de la aritmética formalizados en TNT:

- Conjetura de Goldbach:

$$\forall x : \exists x' : \exists x'' : (SS0 \cdot SSx) = (x' + x'') \wedge P(x') \wedge P(x'')$$

donde $P(x)$ es el predicado:

$$(\exists x' : x = SSx') \wedge (\forall x'' : \forall x''' : \neg(x = (Sx'' + S0) \cdot (Sx''' + S0)))$$

"Todo número par mayor que 2 es suma de dos primos"

- Teorema de Euclides:

$$\forall x : P(x) \rightarrow (\exists x' : \exists x'' : (P(x') \wedge x' = (x + Sx'')))$$

'Existen infinitos primos"

Los sistemas formales y sus propiedades

Reglas de inferencia de TNT (I):

- **Especificación**: Sea u una variable que aparece dentro de la cadena P. Si la cadena $\forall u : P$ es un teorema, entonces P también lo es, y también lo son todas las cadenas obtenidas de P reemplazando u, en cualquiera de sus apariciones, con cualquier término dado. El término que reemplaza a u no debe contener una variable que se cuantifique en P.
- **Generalización**: Sea P un teorema en el que aparece la variable libre u. Entonces $\forall u : P$ es un teorema.

Los sistemas formales y sus propiedades

Reglas de inferencia de TNT (II):

- **Intercambio**: Si u es una variable, $\forall u : \neg$ y $\neg\exists u :$ son intercambiables dentro de cualquier teorema.
- **Existencia**: Permite sustituir un término τ que aparece en una cadena por una variable x , añadiendo antes $\exists x :$ de modo apropiado. Por ejemplo, podemos pasar de $\forall x : \neg Sx = 0$ a $\exists x' : \forall x : \neg Sx = x'$
- **Introducción y eliminación de S**: Se puede pasar de $\tau = \delta$ a $S\tau = S\delta$ (y viceversa), donde τ, δ son términos.
- **Igualdad**: La igualdad es simétrica y transitiva. (Es decir, si $P = Q$ entonces $Q = P$; además, si $P = Q$ y $Q = R$, entonces $P = R$)

Los sistemas formales y sus propiedades

Reglas de inferencia de TNT (III):

- **Inducción**: Si

$$P(0)$$

y

$$\forall x : (P(x) \to P(x+1))$$

entonces

$$\forall x : P(x)$$

Esta regla puede también asumirse en la forma de un "esquema de axiomas":

$$\forall P : ((P(0) \wedge \forall x : (P(x) \to P(x+1))) \to (\forall x : P(x)))$$

Es decir, es una lista infinita de axiomas -uno, para cada P- que tienen una estructura común. (Es por esto que se admiten infinitos axiomas en un sistema formal)

Los sistemas formales y sus propiedades

Definición

Llamamos sistema formal (o teoría formalizada) T a la entidad formada por

- *Un lenguaje $\mathcal{L}$ completamente formalizado (es decir, un vocabulario y unas reglas gramaticales para la formación de palabras (i.e., fórmulas)). La gramática de $\mathcal{L}$ determina las fórmulas bien formadas (WFF), etc.*
- *Un conjunto (no necesariamente finito) de axiomas. (Cuando intervienen infinitos axiomas, se hace en la forma de "esquemas de axiomas").*
- *Un conjunto (finito y, de hecho, lo más reducido posible) de reglas de inferencia.*

Los axiomas, que se expresan en el lenguaje $\mathcal{L}$, representan la base de la que se parte o sobre la que se apoya el sistema formal para derivar todos sus resultados (o teoremas). Las reglas de inferencia determinan las formas de razonamiento permitidas en el sistema.

Asumimos que todas las teorías formales T que consideremos están efectivamente axiomatizadas.

Los sistemas formales y sus propiedades

Definición (Teorema = Consecuencia sintáctica)

Dado un sistema formal T, la sentencia A es un teorema de T (también llamado "consecuencia sintáctica" de T) y lo denotamos por

$$T \vdash A$$

si A es la última fórmula de una derivación formal en T que parte de los axiomas de T.

Definición (Consistencia)

El sistema formal T es inconsistente si permite derivar un teorema del tipo $A \wedge \neg A$. Cuando no es inconsistente (es decir, si no permite derivar contradicciones), decimos que es consistente.

Definición (Completitud (sintática))

El sistema formal T es sintácticamente completo (también se dice "negación completo") si para toda sentencia A se tiene que o bien $T \vdash A$ o bien $T \vdash \neg A$.

Los sistemas formales y sus propiedades

El ideal formalista consiste en identificar una teoría matemática con un sistema formal apropiado que la describa completamente. En tal caso, todas las afirmaciones de la teoría deberían admitir una formulación en el sistema formal y, además, todas ellas deberían admitir una prueba o una refutación (es decir, una prueba de su negación).
Si tal es el caso, podremos identificar la **verdad** de una afirmación con su **demostrabilidad** y afirmar que una sentencia es cierta precisamente cuando existe una prueba de ella dentro del sistema.
Sin embargo, este ideal se ha mostrado tremendamente escurridizo. Los primeros ejemplos de que un conjunto determinado de axiomas -por más que uno crea que son los apropiados para definir una teoría- no siempre fijan la verdad de forma inexorable los tenemos en la geometría de Euclides y en las geometrías no-Euclídeas que, sin tener una aparente conexión con la realidad física, mostraron su validez gracias al uso de **modelos** apropiados.

Los sistemas formales y sus propiedades

Definición (Verdad = Consecuencia semántica)

Decimos que una sentencia A de T es verdadera (es una consecuencia semántica de T) y lo denotamos por

$$T \models A$$

si es cierta en todos los posibles modelos de T.

Definición (Coherencia)

Diremos que el sistema formal T es coherente si todos sus teoremas son verdad en todos los modelos que existan de T. (Es decir, el sistema es coherente si en él es imposible derivar resultados que son falsos para algún modelo del sistema)

Definición (Completitud semántica)

Diremos que el sistema formal T es semánticamente completo (o adecuado) si todas las verdades del sistema son teoremas.

Los sistemas formales y sus propiedades

¿Por qué preocuparnos por los lenguajes formales? Porque el lenguaje cotidiano está repleto de redundancias y ambigüedades, por no mencionar las sentencias que sencillamente carecen de condiciones claras de verdad. De modo que, cuando tratamos con argumentos complejos, ayuda someterlos a un lenguaje artificial adecuado, expresamente diseñado para librarnos de oscuridades, en el que la forma revela la estructura lógica.

Los sistemas formales y sus propiedades

¿Por qué preocuparnos por las deducciones formales? Porque los argumentos informales frecuentemente utilizan premisas que se han suprimido e inferencias falaces. Sencillamente, es demasiado fácil engañar. Redactar los argumentos como deducciones formales en uno u otro estilo obliga a ser honestos, pues debemos detallar todas las premisas y las reglas de inferencia que invocamos durante la deducción. Y la honestidad es la mejor de las políticas.

Los sistemas formales y sus propiedades

De modo que al unir el objetivo de claridad perfecta e inferencia honesta de los lógicos al proyecto de los matemáticos de regimentar una teoría en un conjunto preciso de axiomas, podemos apreciar el concepto de una teoría formal axiomatizada como un ideal compuesto.
En todo caso, es claro que podemos contemplar toda teoría completamente formalizada como un cálculo sobre cuyo alcance podemos hacernos preguntas.

Los sistemas formales y sus propiedades

Teorema (Bernays-Hilbert, 1918; Post, 1920)

La lógica de enunciados es consistente y sintácticamente y semánticamente completa.

Teorema (Gödel, 1930)

El cálculo de predicados de primer orden es semánticamente completo.

Teorema (Gödel, 1931; Primer teorema de incompletitud)

Si un sistema formal T contiene suficiente aritmética y es consistente, entonces es sintácticamente incompleto e incompletable.

Teorema (Gödel, 1931; Segundo teorema de incompletitud)

Si un sistema formal T contiene suficiente aritmética entonces es incapaz de probar su propia consistencia.

Los sistemas formales y sus propiedades

- La consistencia de la lógica de enunciados se deduce de que todos los axiomas son tautologías y “ser tautología"permanece invariante frente a la aplicación de todas las reglas de derivación. Por tanto, es imposible derivar $P \wedge \neg P$.
- La completitud semántica se basa en un truco que permite transformar los cálculos realizados para obtener la tabla de verdad de un enunciado en una deducción del mismo (ver, e.g., M. Garrido "Lógica simbólica").
- Gracias a los apartados anteriores, en lógica de enunciados se garantiza la equivalencia entre ser un teorema y ser una tautología (semánticamente válido). Como hay algoritmos que calculan los valores de verdad de un enunciado, concluimos que hay algoritmos para conocer mecánicamente si un enunciado es o no un teorema: la teoría es decidible.

Gödel realizó una demostración de la completitud semántica del cálculo de predicados en su tesis doctoral, en 1930. Es un resultado mucho más complejo que el correspondiente a la lógica de enunciados y se consideró un paso importante para el programa de Hilbert.

Los sistemas formales y sus propiedades

Un pequeño inciso: ¡Hay que tener mucho cuidado con los modelos!

Teorema (Löwenheim-Skolem)

Si una teoría formalizada (de primer orden) T tiene un modelo infinito, entonces tiene también un modelo numerable.

- Por ejemplo, ¡existe un modelo numerable de la teoría de números reales e incluso para la teoría elemental de conjuntos! (A pesar de que sabemos que $\mathbb{R}$ no es numerable).
- Esto no es, en realidad, contradictorio. Cuando decimos que $\mathbb{R}$ y $\mathbb{N}$ tienen cardinales distintos, significa que no existen aplicaciones sobreyectivas $f : \mathbb{N} \to \mathbb{R}$. En un modelo numerable de los números reales, lo que sucederá es que -sean lo que sean las aplicaciones en dicho modelo, así como $\mathbb{N}$ y $\mathbb{R}$-, no habrá suficientes como para que podamos encontrar una de ellas sobreyectiva que lleve $\mathbb{N}$ a $\mathbb{R}$.
- En todo caso queda claro que no debemos restringir nuestro pensamiento sobre un sistema formal al modelo concreto que lo haya motivado, porque es seguro que hay otros muy distintos.

El primer teorema de incompletitud de Gödel

Teorema (Gödel, 1931; 1^{er} teorema de incompletitud, v. semántica)

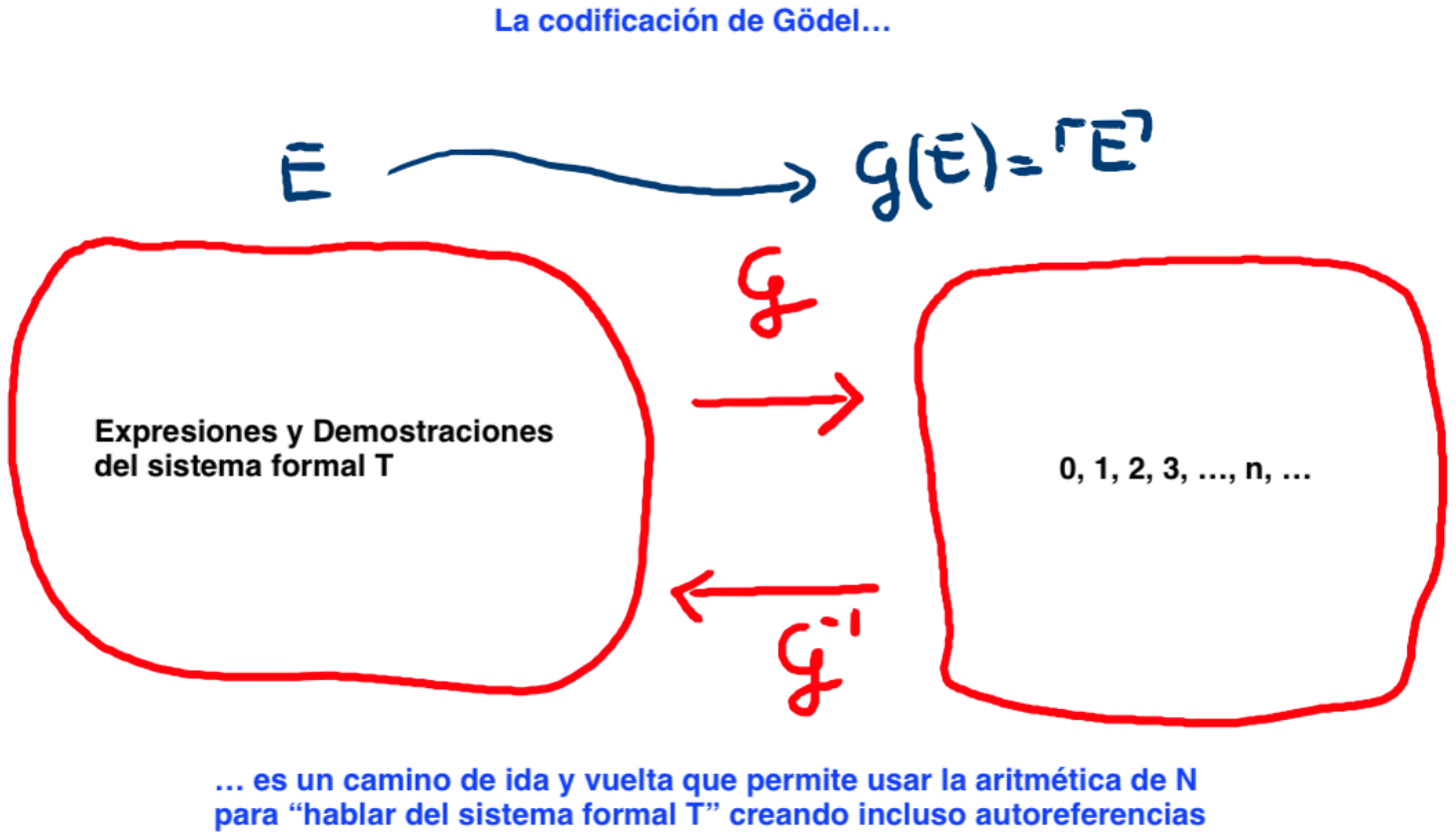
Si un sistema formal T contiene el lenguaje de la aritmética básica y es coherente, entonces es semánticamente incompleto e incompletable.

Teorema (Gödel, 1931; 1^{er} teorema de incompletitud, v. sintáctica)

Si un sistema formal T contiene suficiente aritmética y es consistente, entonces es sintácticamente incompleto e incompletable.

- Por "contiene suficiente aritmética" se entiende que en él se puede modelar una versión débil de la aritmética conocida como "aritmética de Robinson". En particular, el sistema formal TNT (que modela la aritmética de Peano) verifica esta hipótesis.
- La formulación inicial del resultado requería una hipótesis más técnica que la mera consistencia (llamada ω-consistencia), pero el resultado fue reducido a la formulación actual por J. B. Rosser en 1936.

La numeración de Gödel: Codificación



La numeración de Gödel: Codificación

- La clave de las pruebas de Gödel está en establecer un "diccionario" que "traduce" expresiones del sistema formal T a números enteros y relaciones (o fórmulas) aritméticas en fórmulas de T que hablan sobre su propia capacidad expresiva.
- El primer éxito de Hilbert se basó en la construcción de un diccionario entre el álgebra y la geometría

$$\begin{aligned} \mathbb{I} \to V(\mathbb{I}) &= \{x \in \mathbb{K}^d : p(x) = 0 \ \forall p \in \mathbb{I}\} \\ S \subseteq \mathbb{K}^d \to I(S) &= \{p \in \mathbb{K}[x_1, \cdots, x_d] : p(x) = 0 \ \forall x \in S\} \\ I(V(\mathbb{I})) &= \sqrt{\mathbb{I}} \end{aligned}$$

y el fracaso de su programa formalista también pasó por la construcción de otro diccionario -esta vez entre la lógica y la aritmética.

La numeración de Gödel: Codificación

Consideremos el lenguaje de la aritmética L_A, sobre el cual se formula un sistema formal o teoría T. Vamos a asociar, a cada expresión E de L_A, un número $\ulcorner E \urcorner$.

A cada uno de los símbolos lógicos de L_A, así como al símbolo S (de sucesor), al 0, y las operaciones $+$ y $\times$ se les asocia (de un modo determinado de antemano) un número impar:

$\neg$	$=$	$\wedge$	$\vee$	$\rightarrow$	$\leftrightarrow$	$\forall$	$\exists$	$($	$)$	0	S	$+$	$\cdot$
1	3	5	7	9	11	13	15	17	19	21	23	25	27

Por ejemplo,

$$\ulcorner \forall \urcorner = 13$$

Las variables que puede usar L_A forman un conjunto numerable, por lo que las podemos denotar por v_k, $k = 0, 1, \cdots$. Por tanto, forzamos la siguiente asociación:

$$\ulcorner v_k \urcorner = 2(k+1) \qquad k = 0, 1, \cdots$$

La numeración de Gödel: Codificación

Sea

$$\pi_0 = 2 < \pi_1 = 3 < \pi_2 = 5 < \cdots < \pi_n < \cdots$$

la sucesión (infinita) de los números primos. Sea $E = s_0 s_1 s_2 s_3 \cdots s_k$ una expresión de L_A formada por la concatenación de los $k+1$ símbolos s_i, cada uno de los cuales puede ser bien un símbolo lógico o bien una variable v_s. Como ya hemos determinado de forma unívoca los valores $\ulcorner s_i \urcorner$, ahora podemos definir:

$$\ulcorner E \urcorner = \pi_0^{\ulcorner s_0 \urcorner} \cdot \pi_1^{\ulcorner s_1 \urcorner} \cdot \ldots \cdot \pi_k^{\ulcorner s_k \urcorner}$$

Por ejemplo, si consideramos la expresión $0 = 0$, como $\ulcorner 0 \urcorner = 21$ y $\ulcorner = \urcorner = 3$, tendremos que

$$\ulcorner 0 = 0 \urcorner = 2^{21} \cdot 3^3 \cdot 5^{21} = 27 \cdot 10^{21} = 27000000000000000000000$$

La numeración de Gödel: Codificación

Otro ejemplo:

$$
\begin{aligned}
& \ulcorner \exists v_0 (S0 + v_0) = SS0 \urcorner \\
= \;& 2^{15} \cdot 3^2 \cdot 5^{17} \cdot 7^{23} \cdot 11^{21} \cdot 13^{25} \cdot 17^2 \cdot 19^{19} \cdot 23^3 \cdot 29^{23} \cdot 31^{23} \cdot 37^{21} \\
= \;& 165339461420929924684207256578163675371314067548468419430427893776485938068817053144316918483613086674234894085792104913297766263141880618965324612386434455137189307412669758681975914177923741935771496275000000000000000
\end{aligned}
$$

La numeración de Gödel: Codificación

Consideremos ahora una demostración D dentro de L_A. Esta estará formada por una secuencia (finita) de expresiones

$$E_0, E_1, \cdots, E_t$$

de L_A.

- Todas ellas serán expresiones bien formadas.
- La expresión E_{i+1} debe deducirse de las expresiones anteriores $E_0, \cdots, E_i$ aplicando alguna regla de inferencia apropiada.
- E_t debe ser la expresión del teorema que se quiere probar.

Pues bien:

$$\ulcorner D \urcorner = \pi_0^{\ulcorner E_0 \urcorner} \cdot \pi_1^{\ulcorner E_1 \urcorner} \cdot \ldots \cdot \pi_t^{\ulcorner E_t \urcorner}$$

La numeración de Gödel: Codificación

Asociadas a la numeración de Gödel podemos construir numerosas funciones aritméticas computables -que es una forma de decir que sus valores se pueden calcular aplicando un algoritmo, que quedan descritas por un programa informático (aunque Gödel obviamente no empleó estos términos, pues no existían):

$$\begin{aligned} \text{len}(n) &= \text{número de factores primos de } n \\ \text{exf}(n,i) &= \text{es el exponente de } \pi_i \\ &\quad \text{en la descomposición en factores primos de } n \end{aligned}$$

La numeración de Gödel: Codificación

- Si n es el número de Gödel de una fórmula, entonces es el producto de los primeros $\mathrm{len}(n)$ primos elevados a potencias apropiadas.
- Es más, si $n = \ulcorner E \urcorner$, donde $E = s_0 \cdots s_t$, entonces

$$t = \mathrm{len}(n) - 1 \text{ y } \ulcorner s_i \urcorner = \mathrm{exf}(n, i),\ i = 0, 1, \cdots, \mathrm{len}(n) - 1.$$

Es decir:

$$n = \pi_0^{\mathrm{exf}(n,0)} \cdot \pi_1^{\mathrm{exf}(n,1)} \cdot \ldots \cdot \pi_{\mathrm{len}(n)-1}^{\mathrm{exf}(n,\mathrm{len}(n)-1)}$$

- Finalmente, si $n = \ulcorner D \urcorner$ es el número de Gödel de una sucesión de expresiones $D = E_0, \cdots, E_h$, entonces

$$h = \mathrm{len}(n) - 1 \text{ y } \ulcorner E_i \urcorner = \mathrm{exf}(n, i),\ i = 0, 1, \cdots, \mathrm{len}(n) - 1$$

La numeración de Gödel: Codificación

Resumiendo: Hay algoritmos para:

- Calcular los números de Gödel de expresiones y de demostraciones,
- Dado un número natural, recuperar la expresión o secuencia de expresiones que representa. (Y si no representa nada, confirmarlo)

Estos algoritmos se utilizan para:

- Construir funciones (y relaciones) aritméticas que nos hablan del comportamiento lógico del sistema formal T (y, en general, del lenguaje formal L_A)
- La utilidad de dichas fórmulas y relaciones se basa en dos cuestiones técnicas: la representación y la diagonalización.

La numeración de Gödel: Codificación

- $$\mathrm{Term}(n) = \begin{cases} 1 & \text{si } n = \ulcorner E \urcorner, \text{ para algún término } E \\ 0 & \text{en otro caso} \end{cases}$$

- $$\mathrm{Wff}(n) = \begin{cases} 1 & \text{si } n = \ulcorner E \urcorner, \text{ para una fórmula bien formada } E \\ 0 & \text{en otro caso} \end{cases}$$

- $$\mathrm{Dem}(m,n) = \begin{cases} 1 & \text{si } m \text{ es el número de Gödel de una demostración} \\ & \text{de un teorema de TNT cuyo número de Gödel es } n \\ 0 & \text{en otro caso} \end{cases}$$

La numeración de Gödel: Codificación

Obsérvese que dados m,n:

- Hay un algoritmo para rescatar en L_A el significado de m y de n.
- Dicho algoritmo nos muestra el texto asociado a m en todo detalle y por tanto, comprobar si dicho texto se corresponde con una prueba (en TNT) del texto que representa n, es trivial -se hace mecánicamente.

Por tanto:

$$\text{Dem}(m,n) \text{ es computable}$$

La numeración de Gödel: Representación

Teorema (Representación)

Existe una fórmula de L_A, que denotamos por Dem_T, tal que

$$T \vdash Dem_T(\overline{m}, \overline{n})$$

si y solo si m es el número de Gödel de una demostración de un teorema de TNT cuyo número de Gödel es n

Por tanto, si $\mathrm{Dem}_T(x)$ denota la fórmula

$$\exists y \mathrm{Dem}_T(y, x)$$

entonces se tiene que

$$\text{Si } T \vdash A \text{ entonces } T \vdash \mathrm{Dem}_T(\overline{\ulcorner A \urcorner})$$

La numeración de Gödel: Diagonalización

Teorema (Diagonalización)

Sea $A(x)$ una fórmula de T con una única variable libre x. Entonces se puede construir una sentencia D tal que

$$T \vdash D \leftrightarrow A(\overline{\ulcorner D \urcorner}).$$

Si aplicamos este teorema a la fórmula

$$A(x) = \neg \mathrm{Dem}(x)$$

obtenemos que existe una fórmula G_T tal que

$$T \vdash G_T \leftrightarrow \neg \mathrm{Dem}(\overline{\ulcorner G_T \urcorner}).$$

En particular, G_T será verdad sii no es un teorema de T.

Demostración del primer teorema de Gödel

Analicemos la fórmula:

$$T \vdash G_T \leftrightarrow \neg \mathrm{Dem}(\ulcorner G_T \urcorner).$$

Supongamos que

$$T \vdash G_T$$

Entonces, por construcción de Dem, tendremos:

$$T \vdash \mathrm{Dem}(\ulcorner G_T \urcorner)$$

Pero, por otra parte, tendremos también que

$$T \vdash \neg \mathrm{Dem}(\ulcorner G_T \urcorner)$$

porque G_T y $\neg \mathrm{Dem}(\ulcorner G_T \urcorner)$ son sintácticamente equivalentes en T. Esto viola la consistencia de T. Por tanto, no es cierto que $T \vdash G_T$ y G_T no es un teorema de T. ¡Y además sabemos que es cierto precisamente porque dice de si mismo no ser un teorema de T! En particular, $\neg G_T$ es falso y, por tanto, si asumimos que T es coherente (solo prueba verdades), entonces $\neg G_T$ tampoco es un teorema de T.

Demostración del primer teorema de Gödel

- El primer teorema de Gödel no solo garantiza la incompletitud (sintáctica y semántica) de la aritmética de Peano, también garantiza su incompletabilidad -al añadir unos pocos axiomas, nunca se obtiene una teoría completa.
- Existen, sin embargo, versiones débiles de la aritmética de las que sí se sabe que dan lugar a teorías completas, etc.
- También existen otros tipos de teorías (por ejemplo, la geometría plana y la teoría de números reales) con buenas propiedades.

Segundo teorema de Gödel

Teorema (Gödel, 1931; Segundo teorema de incompletitud)

Si un sistema formal T contiene suficiente aritmética entonces es incapaz de probar su propia consistencia.

Consideremos la fórmula

$$\text{Cons } T \equiv \neg\text{Dem}(\ulcorner 0 = 1 \urcorner).$$

Ella representa la consistencia de T porque afirma que es imposible derivar una falsedad (por ejemplo, $0 = 1$) en T. El primer teorema de Gödel nos dice que

$$\text{Cons } T \Rightarrow \neg\exists z : \text{Dem}(z, \ulcorner G_T \urcorner)$$

Es decir

$$\text{Cons } T \Rightarrow G_T$$

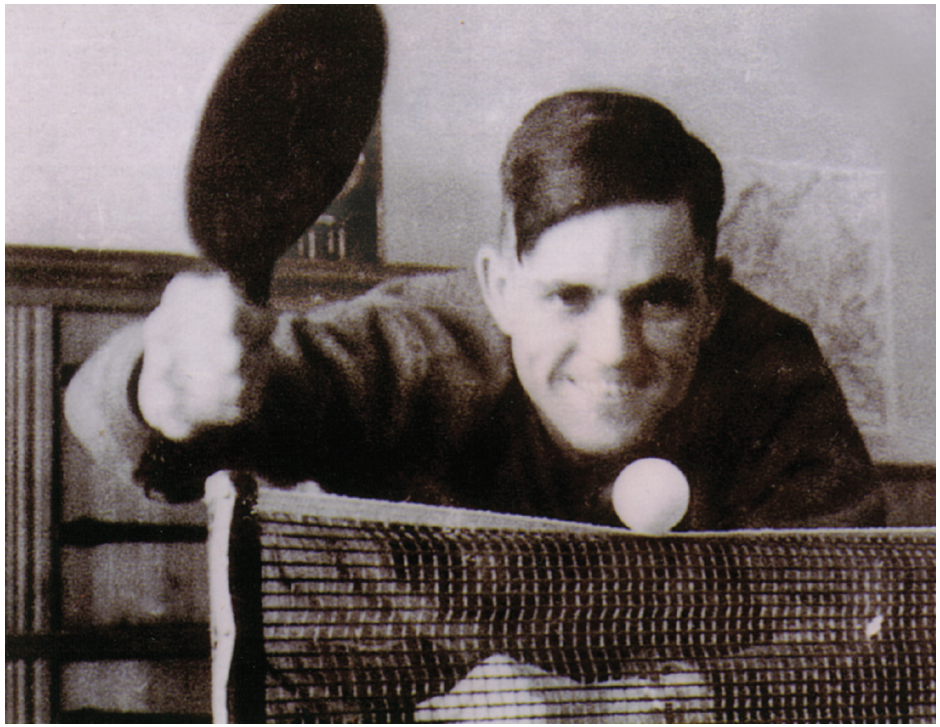
Por tanto, Cons T no es demostrable en T porque ello conduciría a una prueba de G_T en T, lo cual sabemos es imposible.

Segundo teorema de Gödel

- El segundo teorema de Gödel no afirma la imposibilidad de demostrar la consistencia de, por ejemplo, la aritmética. Solo afirma que dicha prueba no puede ser deducida dentro del mismo sistema formal que caracteriza la aritmética.
- Es decir: cabe la posibilidad de demostraciones de consistencia basadas en otros sistemas formales diferentes. (Normalmente, se afirma que son "más fuertes" y se minimiza el valor de la prueba precisamente porque se asume que el nuevo sistema formal es más complejo que el original y, por tanto, su propia consistencia es más dudosa)
- Incluso cabe aún la posibilidad de una demostración finitista (fuera de la aritmética) de la consistencia de la aritmética.

Segundo teorema de Gödel

- Gentzen (1936) (y Ackermann, 1940) demostraron la consistencia de la aritmética usando una versión débil del principio de inducción transfinita. En mi opinión dichas pruebas sí que tienen el valor de apuntar en la dirección de que la consistencia de la aritmética es más verosímil que la inconsistencia -y no solo por motivos psicológicos sino porque disponemos de argumentos sólidos-



Segundo teorema de Gödel

“ A continuación me gustaría llamar su atención sobre un punto frecuentemente descuidado, a saber, el hecho de que el esquema de Hilbert para los fundamentos de las matemáticas sigue siendo muy interesante e importante a pesar de mis resultados negativos. Lo que ha sido probado es sólo que el objetivo epistemológico específico que Hilbert tenía en mente no puede lograrse. Este objetivo era probar la consistencia de los axiomas de las matemáticas clásicas sobre la base de pruebas tan concretas e inmediatamente convincentes como la aritmética elemental. Sin embargo, al ver la situación desde un punto de vista puramente matemático, las pruebas de consistencia sobre la base de presupuestos metamatemáticos más fuertes adecuadamente elegidos (como han sido dadas por Gentzen y otros) son igualmente interesantes, y conducen a conocimientos muy importantes sobre la estructura de las matemáticas desde el punto de vista de la teoría de la demostración. ”

Kurt Gödel a Constance Reid

Algunas reflexiones adicionales: lógica y computación

Existen varias formas de abordar el concepto de algoritmo

- Gödel usó el concepto de función recursiva
- Turing introdujo las máquinas de Turing
- Church creó el λ-cálculo.

Las tres ideas son equivalentes. La **tesis de Church-Turing** afirma que todos estos conceptos (que sabemos son equivalentes entre sí) son la formalización correcta de lo que podemos entender por **algoritmo computable**.

Algunas reflexiones adicionales: lógica y computación

- Una función $f : \mathbb{N} \to \mathbb{N}$ se dice computable si existe un algoritmo A que la computa. Es decir, para todo $n \in \mathbb{N}$,
  - $f(n) = A(n)$ si A produce una salida al introducir n
  - $f(n)$ no existe -no está definida- si A no produce salida (porque entra en bucle o da error) al introducir n.
- Una propiedad (o relación) numérica es decidible si la función característica asociada

$$c_P(n) = \begin{cases} 1 & \text{si } P(n) \text{ es cierto} \\ 0 & \text{en otro caso} \end{cases}$$

  es computable.
- El conjunto $\Delta \subseteq \mathbb{N}$ es decidible si su función característica

$$c_\Delta(n) = \begin{cases} 1 & \text{si } n \in \Delta \\ 0 & n \notin \Delta \end{cases}$$

  es computable.
- $\Delta \subseteq \mathbb{N}$ es efectivamente numerable si $\Delta = \emptyset$ o existe una función sobreyectiva $f : \mathbb{N} \to \Delta$ computable.

Algunas reflexiones adicionales: lógica y computación

Teorema

Todo conjunto decidible es efectivamente numerable.

Teorema

Si Δ y $\Delta^c = \mathbb{N} \setminus \Delta$ son ambos efectivamente numerables, entonces Δ es decidible.

Teorema

$\Delta \subseteq \mathbb{N}$ es efectivamente numerable si y solo si es el dominio de un cierto algoritmo numérico. (Es decir, el conjunto de inputs admisibles para un determinado algoritmo).

Algunas reflexiones adicionales: lógica y computación

Nota: Fijado un lenguaje de programación universal (como C), resulta que solo es posible construir un conjunto numerable de algoritmos. Por tanto:

Teorema

La familia de los conjuntos efectivamente numerables es numerable. Por tanto, existen subconjuntos de $\mathbb{N}$ que no son efectivamente numerables. Estos conjuntos tampoco son decidibles.

Algunas reflexiones adicionales: lógica y computación

Definición

Una propiedad numérica P queda expresada por la wff $\varphi(x)$ sii se tiene que:

- *Si $P(n)$ es cierto entonces $\varphi(\overline{n})$ es cierto.*
- *Si $P(n)$ es falso entonces $\neg\varphi(\overline{n})$ es cierto.*

Definición

La teoría T captura la propiedad numérica P con la wff $\varphi(x)$ sii se tiene que:

- *Si $P(n)$ es cierto entonces $T \vdash \varphi(\overline{n})$.*
- *Si $P(n)$ es falso entonces $T \vdash \neg\varphi(\overline{n})$.*

Algunas reflexiones adicionales: lógica y computación

Definición

La teoría T captura la relación numérica R con la wff $\varphi(x,y)$ sii se tiene que:

- *Si mRn es cierto entonces $T \vdash \varphi(\overline{m},\overline{n})$.*
- *Si mRn es falso entonces $T \vdash \neg\varphi(\overline{m},\overline{n})$.*

Definición

*La teoría T es **suficientemente fuerte** si captura todas las relaciones numéricas R que son decidibles.*

Este concepto es natural porque si queremos construir un sistema formal para la aritmética, este no debería mermar nuestras capacidades de cómputo. Lo que sabemos definir con un algoritmo debería ser representable en la teoría y lo que sabemos comprobar con un algoritmo deberíamos poder deducirlo en la teoría. La TNT es un ejemplo de teoría suficientemente fuerte. De ahí que sea verdad el Teorema de representación.

Algunas reflexiones adicionales: lógica y computación

Teorema

Ninguna teoría consistente, efectivamente axiomatizada, y suficientemente fuerte puede ser decidible.

Esto demuestra que si T es una teoría propiamente formalizada, consistente y capaz de demostrar suficiente aritmética, entonces no existe un algoritmo para determinar si una afirmación de la teoría es o no un teorema. En particular, esto resuelve -en sentido negativo- el Encheisdungproblem (propuesto por Hilbert y Ackermann en 1928) para la aritmética de Peano (TNT).

Algunas reflexiones adicionales: lógica y computación

Demostración. Asumimos que T es consistente, efectivamente axiomatizada, suficientemente fuerte y decidible.

- Como la teoría está efectivamente axiomatizada podemos generar una lista $\{\varphi_n(x)\}_{n=1}^{\infty}$ con todas las wff $\varphi(x)$ de T.
- Como la teoría es suficientemente fuerte la lista anterior captura todas las propiedades decidibles.
- Definimos la siguiente propiedad: n tiene la propiedad D sii $T \vdash \neg\varphi_n(\overline{n})$.
- Como T es decidible, D es decidible.
- Como D es decidible, φ_d captura D para cierto d
- Si d tiene la propiedad D, entonces $T \vdash \neg\varphi_d(\overline{d})$ (por def. de D) y $T \vdash \varphi_d(\overline{d})$ (porque φ_d captura D).
- Por tanto, T no puede ser consistente.

El problema de la parada

(Problema de la parada (Halting Problem), Turing, 1936)

Se desea saber si existe un algoritmo S que acepta como entrada todo par (A, I)*, donde A denota un algoritmo e I una entrada para A, y produce la salida*

$$S(A,I) = \begin{cases} 1 & \textit{si el algoritmo A finaliza cuando se introduce la entrada I} \\ 0 & \textit{en otro caso} \end{cases}$$

Además, se requiere que el algoritmo S siempre finalice, independientemente de la entrada (A, I) *suministrada.*

El problema de la parada

Supongamos que el sistema formal T es lo suficientemente fuerte como para razonar sobre algoritmos. En particular, dado un par (A, I), asumimos que existe una proposición p en T que afirma que el algoritmo A termina cuando se le proporciona la entrada I.

El problema de la parada

Supongamos que el sistema formal T es lo suficientemente fuerte como para razonar sobre algoritmos. En particular, dado un par (A, I), asumimos que existe una proposición p en T que afirma que el algoritmo A termina cuando se le proporciona la entrada I.

Proposición

Si T es completo y consistente, el problema de la parada tiene una solución positiva. Por tanto, si dicho problema tiene una solución negativa, entonces todo sistema formal consistente T que tenga cierto poder expresivo será necesariamente incompleto.

- Basta enumerar todas las pruebas en T (algo que no es difícil de hacer) y usar el algoritmo S que sigue todas estas pruebas en orden y comprueba si son una prueba de p o una prueba de no-p.
- La completitud de T garantiza que este algoritmo termina, lo que resuelve el problema de la parada en sentido positivo.

El problema de la parada

Teorema (Turing, Church)

Ningún algoritmo resuelve el problema de la parada.

Demostración: Supongamos que dicho algoritmo existe. Entonces será ejecutado por una máquina de Turing, que denotamos por **Halt**.

$$\mathbf{Halt}(T,t) = \begin{cases} 1 & \text{si } T \text{ finaliza cuando } t \text{ es la entrada proporcionada a } T \\ 0 & \text{en el resto de casos} \end{cases}$$

Construimos ahora otra máquina de Turing que llamamos **Paradox**.

$$\mathbf{Paradox}(P) = \begin{cases} \text{bucle infinito} & \text{si } \mathbf{Halt}(P,P) = 1 \\ \text{finaliza} & \text{en el resto de casos} \end{cases}$$

El problema de la parada

Teorema (Turing, Church)

Ningún algoritmo resuelve el problema de la parada.

Demostración:

- Si $\mathbf{Halt}(\mathbf{Paradox}, \mathbf{Paradox}) = 1$ entonces $\mathbf{Paradox}(\mathbf{Paradox})$ finaliza y, por tanto,

$$\mathbf{Halt}(\mathbf{Paradox}, \mathbf{Paradox}) = 0$$

  -lo que contradice la hipótesis.

- Pero si $\mathbf{Halt}(\mathbf{Paradox}, \mathbf{Paradox}) = 0$ entonces $\mathbf{Paradox}(\mathbf{Paradox})$ entra en un bucle infinito, por lo que

$$\mathbf{Halt}(\mathbf{Paradox}, \mathbf{Paradox}) = 1$$

  -otra contradicción.

Por tanto, **Paradox** no puede existir, y lo mismo sucede con **Halt**.

El problema de la parada

- El mismo argumento, pero escrito en términos de programas -en lugar de máquinas de Turing- fue incluido en

  E.W. Dijkstra, “A short introduction to the art of programming”, 1971

  manuscrito que escribió en 1971, cuando ya había hecho algunas contribuciones fundamentales a la programación.

El problema de la parada

- Tras exponer dicho argumento, concluyó:

  "La moraleja de esta historia es que es una parte intrínseca del deber de todos los que desean componer algoritmos proporcionar una prueba de que el texto de su programa realmente representa un algoritmo adecuado [Es decir: debe ofrecer una prueba de que su algoritmo finaliza.] ”
- Esto representó una motivación fundamental para el desarrollo de las áreas de Construcción y Verificación formal de programas.
- Dijkstra creó la "separación de preocupaciones” al argumentar que es necesario demostrar tanto que los programas finalizan como que responden correctamente a las especificaciones que los definen. Además, incluyó en sus objetivos el estudio de algoritmos no deterministas, que son relevantes por ejemplo en programación concurrente, sistemas operativos, etc.


J. M. ALMIRA, J.C. SABINA DE LIS, *Hilbert. Matemático fundamental,* Vol. 31 de 'La Matemática en sus personajes", Nivola, 2007.


E. ALONSO, *Sócrates en Viena. Una biografía intelectual de Kurt Gödel,* Montesinos, 2007.


G.S. BOOLOS, P. BURGUESS, R.C. JEFFREY, *Computability and logic,* Cambridge University Press, 2002.


J. W. DAWSON, *Logical Dilemmas: The Life and Work of Kurt Gödel,* CRC Press, 2005.


T. FRAZÉN, *Gödel's theorem. An incomplete guide to its use and abuse,* A.K. Peters, 2005.


H. J. GENSLER, *Gödel's theorem simplified,* University Press of America, 1984.


D. R. HOFSTADTER, *Gödel, Escher, Bach. Un eterno y grácil bucle,* Tusquets, 1987.


G. LOLLI, *La máquina y las demostraciones,* Alianza Universidad, 1991.


LORENZ HALBEISEN, REGULA KRAPF, *Gödel's Theorems and Zermelo's Axioms. A Firm Foundation of Mathematics*, Birkhäuser, 2020.


G. MARTÍNEZ, G. PIÑEIRO, *Gödel ∀ (para todos)*, Destino, 2010.


E. NAGEL, J.R. NEWMAN, *El teorema de Gödel,* Tecnos, 1994.


J. PLA I CARRERA, *El teorema de Gödel: un análisis de la verdad matemática,* RSME, 2013.


N. SHANKAR, Metamathematics, machines, and Gödel's proof, Cambridge University Press, 1994.


P. SMITH, *Gödel Without (Too Many) Tears*, Logic Matters, 2020.


J. MOSTERÍN, *Los lógicos*, Espasa, 2000.