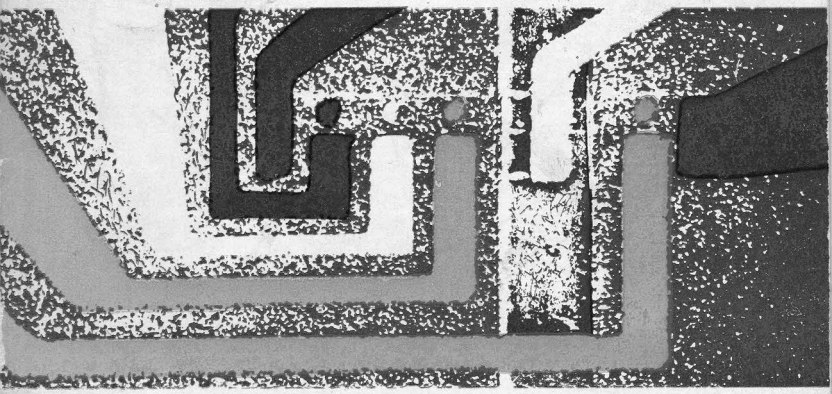
RODRIGO VALENZUELA C.

LIMITACIONES MATEMATICAS DE LOS METODOS DE COMPUTACION

CONSEJO DE RECTORES DE LAS UNIVERSIDADES CHILENAS



12

FASCICULOS
PARA LA COMPRENSION DE
LA CIENCIA,
LAS HUMANIDADES
Y LA TECNOLOGIA

EDITORIAL UNIVERSITARIA

MATEMATICA

CIENCIAS EXACTAS Y NATURALES

12 FASCICULOS
PARA LA COMPRENSION DE LA CIENCIA, LAS HUMANIDADES Y LA TECNOLOGIA

Esta Colección

se publica bajo la dirección del

Consejo de Rectores de

las Universidades Chilenas

Departamento Académico

IMPRESO EN CHILE / PRINTED IN CHILE

LIMITACIONES MATEMÁTICAS DE LOS MÉTODOS DE COMPUTACIÓN

RODRIGO VALENZUELA C.

EDITORIAL UNIVERSITARIA

Texto compuesto con fotomatrices
Photon Perpetua

Se terminó de imprimir esta 1ª edición en los talleres de
EDITORIAL UNIVERSITARIA,
San Francisco 454, Santiago de Chile,
en el mes de septiembre de 1978
5.000 ejemplares

Proyectó la edición *Mauricio Amster*
Cubierta de *Eddy Carmona, Fernando Polvorín, Ximena Ulibarri*

INDICE

Para
VERENA

NOTA PRELIMINAR

Con el gran desarrollo de los computadores digitales de los últimos años, el estudio de los procedimientos de computación automáticos ha adquirido mucha importancia. Sin embargo, aun antes de este desarrollo, se desenvolvió una teoría matemática sobre lo que era computable en principio por tales computadores, es decir, computable sin tomar en cuenta las limitaciones tecnológicas que pueda haber en un período determinado. Así, lo que no es computable en principio, no lo será nunca, aunque los avances tecnológicos sean enormes.

En este volumen, se presenta esta teoría matemática de la computabilidad. Se usa para ello, un artículo muy importante y completo de E. Post, aparecido en 1944. Como en todos los volúmenes de la serie de matemática, se acompaña una extensa explicación de los principales conceptos. Esto se debe al carácter muy técnico de los artículos originales en matemática, lo que los hace imposibles de entender directamente para los no iniciados. La introducción de Rodrigo Valenzuela cumple admirablemente el propósito de hacer el artículo comprensible.

ROLANDO CHUAQUI

PROLOGO

Los aportes más profundos de este siglo a la matemática provienen de la investigación de sus fundamentos, destacándose particularmente el estudio de los problemas de decisión de la lógica, de la matemática o de partes de la matemática.

Los grandes logros en este campo han sido esencialmente obra de los matemáticos Church, Gödel, Kleene, Post, Rosser y Turing, cada cual autor de publicaciones fundamentales. Por estar escrito en un estilo brillante e informal que se adecúa perfectamente a la finalidad de divulgación que persigue este trabajo, he escogido presentar la primera parte de un artículo de Post titulado »Conjuntos recursivamente enumerables de enteros positivos y sus problemas de decisión«. Introduce la teoría de funciones recursivas y, a la vez, su relación con el teorema de incompletitud de Gödel.

La introducción y el primer capítulo aseguran que el lector comprenda los problemas de interés general que se encuentran detrás de un lenguaje aparentemente técnico y árido. Se aprovecha la ocasión para destacar algunas características del quehacer matemático. El segundo capítulo contiene las primeras cuatro secciones del artículo de Post.

INTRODUCCION

1. METODOS GENERALES PARA RESOLVER CLASES DE PROBLEMAS

Más que resolver un problema aislado, es a menudo de gran utilidad, en matemática, encontrar un método general y simple cuyo uso permita resolver cualquiera de toda una clase de problemas. Así, por ejemplo, más que resolver el problema »¿Cuánto es tres por cinco?« interesa encontrar un método general que permita resolver fácilmente cualquier problema de la clase de problemas de la forma »¿Cuánto es *m* por *n*?«. Encontrado dicho método su simple aplicación nos indicará no sólo cuánto es tres por cinco sino también cuánto es ocho por trece, doscientos quince por veintitrés, y cualquier producto que nos pueda interesar. Tal es el método para multiplicar que se nos enseña en el colegio. Otros métodos generales para resolver problemas de determinadas clases son la regla de tres, la regla para resolver sistemas de ecuaciones lineales, la regla para resolver ecuaciones de segundo grado, la regla para derivar polinomios, la fórmula para encontrar la circunferencia del círculo dado el radio, la fórmula para encontrar la superficie de un triángulo dadas la base y la altura,

etcétera... La importancia de estos métodos generales es consecuencia tanto de la amplitud de su ámbito de aplicación como de lo fácil que resulta aplicarlos.

Pero fuera de las clases de problemas mencionados en el párrafo anterior, hay otras de sumo interés que tal vez el lector no ha considerado. Por ejemplo, ¿existirá por ahí, a la espera de ser descubierto, un método general de fácil aplicación que sirva para resolver *cualquier problema de la matemática?* Tome conciencia el lector de lo que esto significaría. Sería algo así como disponer de una fórmula cuya simple y mecánica aplicación nos permitiría resolver cualquier problema matemático. Con toda seguridad dicho método general se podría traducir en un programa que, alimentando a una computadora, le permitiría resolver cualquier problema de la matemática. Y quedarían cesantes los matemáticos.

Si esta inquietud parece demasiado ambiciosa, o precaviendo la posibilidad de que no se encuentre o no exista dicho método general, se puede reducir el alcance de la interrogante a otra cuestión también de gran interés. Recordará el lector la estructura axiomática que tiene la geometría euclideana que aprendió en el colegio. Se fijan de antemano ciertos axiomas, que son enunciados que no necesitan

demostración por suponerse verdaderos, y a partir de ellos se deducen otros enunciados que se llaman teoremas. Dar forma axiomática a una rama de la matemática tiene la gran ventaja de permitir la individualización clara de los supuestos a partir de los cuales se hacen las deducciones. Más aún, tal como modernamente se estructura un sistema axiomático, permite también precisar las reglas de deducción que se emplean y el lenguaje en que se hacen las deducciones. Este proceder con las cartas sobre la mesa evita que la validez de las deducciones se vea afectada por el uso inconsciente de supuestas verdades o de métodos de deducción no aceptados.

Ahora bien, dado que en los sistemas axiomáticos se precisa el lenguaje usado, los axiomas y las reglas de deducción, es razonable formular la siguiente pregunta acerca de algún sistema axiomático determinado: ¿Existe un método general tal que, ante cualquier afirmación hecha en el lenguaje del sistema, me diga si dicha afirmación se deduce o no de los axiomas? O lo que es igual: ¿Existe un método general que, ante cualquier afirmación formulada en el lenguaje de este sistema axiomático, me diga si dicha afirmación es o no un teorema del sistema?

Tanta es la matemática que se puede cubrir

bajo el paraguas de un sistema axiomático que, para efectos prácticos, disponer de tales métodos generales para algunos sistemas axiomáticos equivaldría a disponer de un método general para resolver cualquier problema de la matemática.

Estas inquietudes ya las planteó en el siglo diecisiete el matemático y filósofo Leibniz, aunque con alcances que rebasan el ámbito de la matemática. Leibniz asignó dos grandes tareas a la ciencia y a la filosofía: 1) descubrir un lenguaje universal en el cual se pueda formular cualquier problema, interrogante o afirmación de la ciencia, y sin ambigüedad alguna *(characteristica universalis)*, y 2) descubrir un método general para manipular las afirmaciones de dicho lenguaje de un modo tal que queden a la luz sus interrelaciones, su significado y su validez *(ars combinatoria)*. En lo referente a su propio ámbito, la matemática del siglo veinte y fines del diecinueve ha hecho frente a ambas tareas. En cuanto a la primera, ha descubierto un lenguaje preciso en el cual se pueden formular todas las afirmaciones y demostraciones de la matemática y en cuyos términos se puede dar un criterio para distinguir fácilmente entre una demostración correcta y una incorrecta. En cuanto a la segunda, o sea, en cuanto a la búsqueda de un método general para decidir ante cualquier afirmación de este lenguaje si ella es verdadera o falsa o si ella es o no teorema de un sistema axiomático, ha llegado a conclusiones que constituyen algunos de los resultados más sorprendentes de la matemática de este siglo. El lector encontrará dichas conclusiones y sus fundamentos en la sección 2 del texto de Emil Post, transcrito más abajo.

2. NATURALEZA FILOSOFICA DE LAS INTERROGANTES PLANTEADAS

Nos hemos planteado las siguientes interrogantes: ¿Existe un método general para resolver cualquier problema de la matemática? ¿Existe, para tal o cual sistema axiomático determinado, un método general que permita decir de cualquier afirmación en el lenguaje del sistema si es o no teorema? Y de estas preguntas es posible derivar otras: ¿Existen tales métodos generales de solución (que distinguen entre lo que es y lo que no es teorema) para algunos sistemas axiomáticos y no para otros? En tal caso, ¿qué características debe tener un sistema axiomático para que tenga método general de solución? Siempre en igual caso, ¿existe un método general tal que ante cualquier

sistema axiomático me diga si dicho sistema tiene o no método general de solución?

La lista podría extenderse indefinidamente, pero aquí nos interesa destacar lo siguiente. Estas interrogantes no son interrogantes matemáticas sino interrogantes sobre la matemática. No estamos preguntando si los números o si las figuras geométricas tienen tales o cuales propiedades. Estamos preguntando si la matemática tiene tales o cuales propiedades. No estamos estudiando los números desde el punto de vista de la matemática, sino que estamos estudiando la matemática desde un punto de vista superior que podríamos llamar de la filosofía de la matemática. Asimismo, por ejemplo, si nos preguntamos, ¿existe un conjunto de axiomas de los cuales se pueda deducir toda verdad física?, no estamos formulando una pregunta de física (sobre el universo) sino una pregunta de filosofía (sobre la física).

Pero existe una gran diferencia entre las posibilidades de hacer frente a un problema de filosofía de la matemática y las de hacer frente a un problema de filosofía de otra ciencia como, por ejemplo, la física. Un problema de filosofía de la física jamás podrá ser enfrentado o resuelto con herramientas de la física. En cambio muchos problemas de la filosofía de la matemática sí se pueden enfrentar y resolver con herramientas matemáticas. Entre ellos, los que nos ocupan. Estos problemas filosóficos se pueden asociar a problemas matemáticos equivalentes. Una vez resuelto el equivalente matemático, con herramientas matemáticas, se extraen las conclusiones filosóficas que correspondan. Es decir, la matemática es la única ciencia que se puede estudiar a sí misma. La »metafísica« no es una rama de la física como lo son la mecánica o la electricidad. En cambio, la metamatemática sí es una rama de la matemática al igual que el álgebra o la geometría.

3. MODELOS

Pero ¿cómo se puede pasar del plano filosófico al matemático y después regresar? El lector podrá observarlo directamente en los capítulos que siguen, pero cabe señalar que no se trata de una característica propia y exclusiva de la relación entre la matemática y su filosofía, sino más bien de la naturaleza del quehacer matemático.

A menudo los problemas matemáticos tienen su origen fuera de la matemática: un problema físico, un problema biológico, un problema filosófico, etcétera... El matemático

hace frente al problema con la ayuda de un modelo. Un modelo se construye abstrayendo del problema original (con una mezcla de arbitrariedad y buen criterio) algunos aspectos que se consideran importantes, dejándose de lado los demás aspectos por no considerarse importantes. En seguida, el matemático estudia la simplificación de la realidad que es el modelo. Por último, regresa a la realidad original (física, biológica, filosófica) atribuyéndole aquello que ha descubierto en el modelo. El procedimiento será exitoso si el modelo efectivamente es reflejo de las características importantes del problema original.

Es posible que el lector, presa de alguna desconfianza frente a los números, mire con escepticismo todo intento de aplicar matemática a la solución de otros problemas que el simple cómputo de cantidades. De ser así, serias dudas debe tener acerca de la coherencia de los capítulos que siguen. Para su tranquilidad, recordemos que la creencia de que la matemática es tan sólo »la ciencia que trata de la cantidad« (como la define el Diccionario de la Lengua Española de la Real Academia) no es más que un mito por no decir una calumnia. Lo esencial del quehacer matemático no está en cuantificar o en trabajar con cantidades, sino en resolver problemas mediante la construcción

y/o el trabajo con modelos. La cuantificación de un problema no es más que un caso muy particular de este quehacer. Que un problema no se pueda cuantificar no significa que la matemática no le sea aplicable. Significa, simplemente, que para estudiar dicho problema no se puede acudir a modelos »cuantificados« sino debe acudirse a otros. Es así como en los capítulos siguientes el lector verá mucha matemática y nada de cantidades.

4. RESUMEN

Nos preguntamos si existen métodos generales para resolver problemas de determinadas clases. La interrogante no constituye un problema matemático sino, más bien, un problema sobre la matemática. No obstante, mediante la construcción de un modelo podemos hacer uso de la propia matemática para darle respuesta.

En el capítulo que sigue esbozamos la construcción de un modelo, mostrando al lector el vínculo que existe entre el problema original y el problema matemático equivalente. Con ello el lector quedará en condiciones de comprender el alcance real de los problemas planteados y/o resueltos en el texto de Emil Post, cuya traducción damos en el capítulo final.

1. PROBLEMAS DE DECISION DE CONJUNTOS DE ENTEROS POSITIVOS

Llamemos *algoritmo* a cualquier conjunto de instrucciones que indique en forma precisa y sin exigir labor creativa alguna, los pasos a seguir para resolver cualquier problema de una determinada clase de problemas. Así, por ejemplo, todos conocemos un algoritmo para sumar. Llamemos *efectivamente computable* a toda clase de problemas para cuya solución exista un algoritmo. Por ejemplo, es computable la clase de problemas de la forma »¿Cuánto es m + n?«.

Detengámonos en este último ejemplo. Sea U el conjunto de todas las afirmaciones de la forma »m + n = p«, y sea V el conjunto de todas las afirmaciones de U que son verdaderas. Consideremos ahora la clase de los problemas de la siguiente forma: »¿Pertenece a V esta afirmación de U?«. Esta clase de problemas será computable si existe un algoritmo que para cualquier elemento de U me diga si está o no ese elemento en V. Es fácil ver que si esta clase de problemas es efectivamente computable, lo es la del último ejemplo y viceversa. Porque si dispongo de un algoritmo para saber

si una afirmación de U está en V, y necesito saber cuánto es m+ n, hago lo siguiente: aplico mi algoritmo para saber si m + n = 1 está en V; si no está, pruebo m + n = 2; si tampoco, veo m + n = 3; y así sucesivamente hasta que encontraré un número q tal que m+ n = q estará en V, momento en el cual responderé a la pregunta original diciendo que m + n es q. A la inversa, si dispongo de un algoritmo para saber cuánto es m + n, para cualquier par de números m y n, y necesito saber si la afirmación m + n = p está en V, me bastará aplicar mi algoritmo para saber cuánto es m+ n y después compararé el resultado con p. Si son iguales, la afirmación está en V; si son distintos, no está en V.

Podemos concluir que estas dos clases de problemas son *efectivamente equivalentes* en el sentido que de un algoritmo para una se obtiene un algoritmo para la otra y viceversa. Luego una es efectivamente computable si y sólo si lo es la otra.

En general, tal como en el caso de este ejemplo, toda clase de problemas es efectivamente equivalente a otra clase de problemas de la forma: »¿Está en V este elemento de U?«, donde U es un conjunto y V es un subconjunto de U. Por lo tanto, como justamente nos interesa la calidad de efectivamente computable que

puede tener o no tener una clase de problemas, no es necesario estudiar *todas* las clases de problemas, ya que nos será suficiente estudiar sólo aquellas clases de la forma »dado este elemento de U, decir si está en V«. Cualquiera otra clase de problemas será equivalente a alguna de éstas.

Retomemos el ejemplo del conjunto U de las afirmaciones de la forma »m + n = p«, y el conjunto V de las afirmaciones de U que son verdaderas. Podemos colocar todas las afirmaciones de U en una lista numerada, por ejemplo de la siguiente manera:

LISTA

número	*elemento de U*
1	0+ 0= 0
2	0+ 0= 1
3	0+ 1= 0
4	1+ 0= 0
5	0+ 1= 1
6	1+ 0= 1
7	1+ 1= 0
8	0+ 0= 2
9	0+ 2= 0
10	2+ 0= 0
11	1+ 1= 1
12	0+ 1= 2
13	0+ 2= 1

número	*elemento de U*
14	1+ 0= 2
15	1+ 2= 0
16	2+ 0= 1
17	2+ 1= 0
18	0+ 0= 3
.	.
.	.
.	.

Claramente, la lista es infinita. Pero lo importante es que existe un método para generarla[1], de modo que si se da un entero positivo cualquiera, n, fácilmente se genera la lista para encontrar la afirmación de U correspondiente al lugar n. A la inversa, si se da una afirmación de U, basta generar la lista hasta que aparezca dicha afirmación y así saber a qué lugar corresponde.

Ahora, sea N el conjunto de todos los enteros positivos, y sea C el conjunto de aquellos enteros positivos que en la lista corresponden a afirmaciones que están en V. Así, por ejemplo, 1, 5 y 6 pertenecen al conjunto C. Conside-

[1] El lector que no esté dispuesto a hacer acto de fe, puede hacer frente al problema de ingenio de saber qué afirmación de U corresponde en la lista al número 19, qué afirmación al 20, y pronto descubrirá el método para generar la lista.

remos la clase de problemas de la forma: »¿Está n en C?«. Es fácil ver que esta clase de problemas es efectivamente equivalente a la clase de los problemas de la forma »¿Pertenece a V esta afirmación de U?«. De hecho, si disponemos de un algoritmo para saber si cualquier n está o no en C, y deseamos saber si a + b =c está en V, hacemos lo siguiente: generamos la lista hasta que aparezca a + b = c; vemos el número que le corresponde por su lugar en la lista; aplicamos nuestro algoritmo a dicho número para saber si está en C. Si está en C, concluimos que a + b = c está en V. Si no está en C, concluimos que a + b = c no está en V. A la inversa, si tenemos un algoritmo para saber si cualquier afirmación de U está en V, y queremos saber si un número está en C, hacemos lo siguiente: generamos la lista hasta alcanzar dicho número; aplicamos nuestro algoritmo a la correspondiente afirmación de U para saber si ella está o no en V. Si está, concluimos que el número está en C. Si la afirmación no está en V, concluimos que el número no está en C.

En general, tal como en este ejemplo, toda clase de problemas es efectivamente equivalente a otra clase de problemas de la forma »¿Está n en C?«, donde C es un conjunto de números enteros positivos. Por lo tanto, como

lo que nos interesa es precisamente la calidad de efectivamente computable que puede tener o no tener una clase de problemas, nos bastará estudiar sólo aquellas clases de problemas de la forma »¿Está n en C?«, donde C es un conjunto de números enteros positivos.

Sea C un conjunto de números enteros positivos. Llamemos *método de decisión* para C, a todo algoritmo que nos diga, para todo entero positivo n, si está o no está en C. Diremos que C es *efectivamente computable* si tiene un método de decisión. Llamaremos el *problema de decisión* de C al siguiente problema: encontrar un método de decisión para C. En caso que C no sea efectivamente computable, diremos que su problema de decisión es *insoluble.*

Uno de nuestros problemas originales, planteado a modo de ejemplo en la introducción, era descubrir, en caso que existiera, un método general (es decir, un algoritmo) para resolver cualquier problema de la matemática. Ahora sabemos que dicho problema es equivalente al siguiente: sea U el conjunto de todas las afirmaciones que se pueden formular en el lenguaje matemático; sea V el conjunto de las afirmaciones de U que son verdaderas; entonces, encontrar un algoritmo que nos diga, para cualquier afirmación de U, si ella está o no está en V. Y por último, haciendo una lista, con al-

gún método de generación, de todas las afirmaciones de U, sea C el conjunto de enteros positivos correspondientes en dicha lista a V. Entonces nuestro problema original es efectivamente equivalente al problema de decisión de C.

Ahora, por ejemplo, si logramos demostrar que es soluble el problema de decisión de cualquier conjunto de enteros positivos, habremos demostrado que, en particular, es soluble el problema de decisión de C. Pero entonces habremos demostrado que existe un algoritmo para resolver cualquier problema de la matemática.

Es aconsejable que el lector reconsidere lo expuesto las veces que sea necesario para comprender como el problema de carácter filosófico acerca de la existencia de un método general para resolver todo problema matemático, comienza a transformarse en un problema matemático acerca de propiedades de conjuntos de números enteros positivos.

Finalizamos esta sección con un par de ejemplos de conjuntos efectivamente computables y de métodos de decisión para ellos.

Ejemplo 1

Sea C el conjunto de los números pares positivos.

Intuitivamente hablando, es evidente que C es efectivamente computable: claramente existe un método general para saber si un entero positivo cualquiera está o no está en C. Así, por ejemplo, no dudamos ni un instante en decir que 2, 4, 6, 48 y 326 están en C, mientras que 5, 9 y 43 no están.

Un método de decisión para C es el siguiente: »Dado un entero positivo, examinar el último dígito. Si éste es 0, 2, 4, 6 u 8, entonces el entero dado está en C. En caso contrario, el entero no está en C«.

Como se puede ver, el método de decisión, siendo por definición un algoritmo, es un conjunto de instrucciones precisas cuya aplicación no requiere de labor creativa alguna.

Habiendo encontrado un método de decisión para C podemos concluir que C es efectivamente computable.

Ejemplo 2

Sea C el conjunto de los enteros positivos, n, tales que el desarrollo decimal de pi contiene una sucesión de al menos n sietes.

A este conjunto no se le conoce método de decisión, pero es fácil ver que tiene uno y que, por lo tanto, es efectivamente computable. Hay dos posibilidades:

1) Que en el desarrollo decimal de pi existan sucesiones de sietes arbitrariamente largas.

2) Que exista un número entero positivo k tal que la sucesión más larga de sietes que aparezca en el desarrollo decimal de pi sea de largo k.

En el primer caso, un método de decisión para C sería el siguiente: »Dado cualquier entero positivo, decir que está en C«.

En el segundo caso, un método de decisión para C sería el siguiente: »Dado un entero positivo, compararlo con k. Si es menor o igual a k, entonces el entero dado está en C. Si es mayor que k, entonces dicho entero no está en C«. Como necesariamente una de las dos posibilidades es la real, entonces uno de los dos métodos de decisión es método de decisión para C.

En este ejemplo puede ver el lector que es posible demostrar que un conjunto tiene método de decisión, y que, por ende, es efectivamente computable, sin necesidad de encontrarle un método de decisión.

2. CONJUNTOS EFECTIVAMENTE ENUMERABLES DE ENTEROS POSITIVOS

De entre todos los conjuntos de enteros positivos, tienen especial interés aquellos que es posible *generar* mediante un algoritmo. Algunos ejemplos aclararán su naturaleza.

Ejemplo 1

a) algoritmo:

»1. Iniciar una lista escribiendo 1.

2. Sumar 2 al último número de la lista, y agregar a la lista el resultado«.

b) conjunto generado: claramente, este algoritmo indica una forma de generar los números impares 1, 3, 5, 7...

Ejemplo 2

a) algoritmo:

»1. Iniciar la lista colocando primero 1 y después 2.

2. Sumar los últimos dos números que aparezcan en la lista y agregar a la lista el resultado«.

b) conjunto generado: 1, 2, 3, 5, 8, 13, 21, 34, 55...

Ejemplo 3

a) algoritmo:

»1. Iniciar la lista colocando 1, 3 y 5.

2. Si los últimos tres números que aparecen en la lista son pares, agregar los tres impares más pequeños que aún no estén en la lista.

3. Si los últimos tres números de la lista son

impares, agregar los tres pares más pequeños que aún no estén en la lista«.

b) conjunto generado: 1, 3, 5, 2, 4, 6, 7, 9, 11, 8, 10, 12..., es decir los enteros positivos.

Llamemos *efectivamente enumerable* a todo conjunto de enteros positivos que se pueda generar mediante un algoritmo.

Dejamos al lector que por sí solo se convenza de lo siguiente: todo conjunto efectivamente computable es efectivamente enumerable.

Llamemos *método de generación* a todo algoritmo que genera un conjunto de enteros positivos.

Decíamos al comenzar esta sección, que los conjuntos efectivamente enumerables de enteros positivos son de especial interés entre los conjuntos numéricos. Esto se debe en gran medida a lo siguiente: recordemos que toda cuestión acerca de si es o no efectivamente computable una clase de problemas matemáticos, es equivalente al problema de decisión de un conjunto de enteros positivos; pues ocurre que para muchas clases de problemas matemáticos que son especialmente importantes, los conjuntos correspondientes de enteros positivos son efectivamente enumerables; por lo tanto, especial importancia tienen los conjuntos efectivamente enumerables de enteros positivos y sus problemas de decisión.

Así, por ejemplo, para *cualquier* sistema axiomático ocurre lo siguiente. Hagamos corresponder al conjunto de todos los enteros positivos el conjunto de todos los enunciados que pueden formularse en el lenguaje de dicho sistema. Esto se puede hacer generando una lista numerada con todos los enunciados que pueden formularse en el lenguaje del sistema. De acuerdo con esta correspondencia, los teoremas del sistema axiomático corresponden a un conjunto C de enteros positivos, y el problema de decisión de C es equivalente al problema de saber si existe un algoritmo para distinguir a los teoremas de entre los enunciados del lenguaje. Pues bien, C es efectivamente enumerable. Es decir, a cada sistema axiomático corresponde un conjunto efectivamente enumerable de enteros positivos cuyo problema de decisión equivale al problema de encontrar un algoritmo que permita distinguir entre los teoremas y los demás enunciados del sistema. Entonces, por ejemplo, si logramos demostrar tan sólo que todo conjunto efectivamente enumerable es efectivamente computable, habremos demostrado que para todo sistema axiomático existe un algoritmo que distingue entre teoremas y otros enunciados.

También se destacan los conjuntos efectivamente enumerables por el siguiente hecho: existe un algoritmo que permite generar todos los métodos de generación. En cambio no podemos decir algo análogo de los conjuntos efectivamente computables: sucede que no existe algoritmo con el cual generar todos los métodos de decisión. No volveremos sobre esta última circunstancia, pero nos detendremos un momento en la descripción de un algoritmo con el cual generar todos los métodos de generación. El hecho de que exista un tal algoritmo tiene gran importancia y será usado frecuentemente en el texto de Post.

Los métodos de generación son instrucciones que se expresan en algún lenguaje. Es posible escoger un lenguaje y precisar en base a él concepto de método de generación, de modo tal que se disponga de un algoritmo que nos permita distinguir entre las sucesiones finitas de enunciados del lenguaje que son métodos de generación y las sucesiones finitas de enunciados que no lo son. Hecho esto, la situación será la siguiente:

(A) El lenguaje estará compuesto sobre la base de un alfabeto finito.

(B) El lenguaje tendrá reglas de sintaxis precisas que nos darán un algoritmo para distinguir de entre las sucesiones finitas

de símbolos del alfabeto aquellas que sean enunciados gramaticalmente correctos.

(C) Como ya se ha dicho, gracias a una definición precisa de lo que es un método de generación, dispondremos de un algoritmo para distinguir entre sucesiones finitas de enunciados correctos que son métodos de generación y sucesiones finitas de enunciados correctos que no lo son.

Consecuencialmente podremos generar todos los métodos de generación de la siguiente manera:

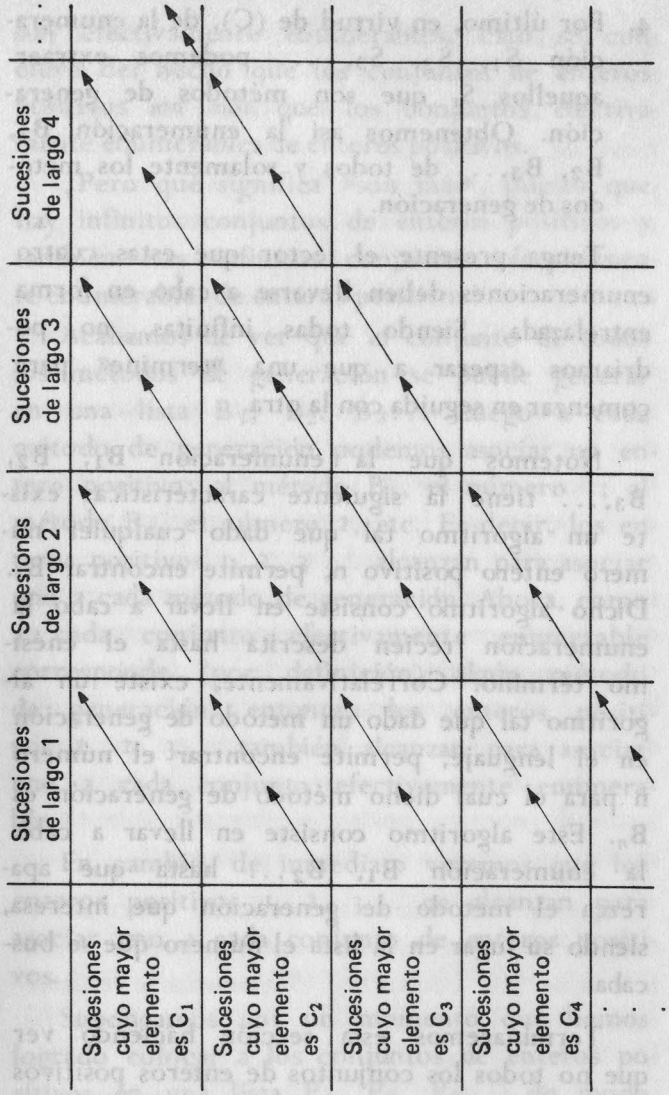
1. Como consecuencia de (A) podemos enumerar (generar), en orden alfabético, todas las expresiones del lenguaje compuestas de un solo símbolo del alfabeto. En seguida, todas las compuestas de dos símbolos. Después, todas las compuestas de tres. Así sucesivamente, enumeraríamos todas las expresiones (es decir, sucesiones finitas de símbolos) del lenguaje. Por ejemplo, si el alfabeto fuera »a, b, c«, la enumeración que hemos descrito comenzaría así: a, b, c, aa, ab, ac, ba, bb, bc, ca, cb, cc, aaa, aab, aac, aba... En términos generales, sea E_1, E_2, E_3, ... la enumeración de todas las expresiones del lenguaje.
2. Ahora, gracias a (B), de la enumeración

E_1, E_2, ... podemos extraer aquellas expresiones que son enunciados gramaticalmente correctos. Sea C_1, C_2, C_3... la enumeración resultante. En ellas están comprendidos todos y solamente los enunciados gramaticalmente correctos.

3. Usando la enumeración C_1, C_2, C_3... podemos enumerar todas las sucesiones finitas de enunciados correctos. Esto puede hacerse como se indica en la figura de la página 35.

Digamos que un enunciado correcto C_n »es mayor« que un enunciado correcto C_m, en caso que C_n aparezca después que C_m en la enumeración C_1, C_2, C_3 ...

Primero ordenamos lexicográficamente (atendiendo a los subíndices i de los C_i) las sucesiones de largo 1 cuyo mayor elemento es C_1. A continuación colocamos, siempre en orden lexicográfico, las sucesiones de largo 2 cuyo mayor elemento es C_1. Y así vamos completando la enumeración que nos interesa, siguiendo el orden indicado por las flechas en la figura. Se va formando de este modo la siguiente lista de sucesiones finitas de enunciados correctos: C_1, C_2, C_1C_1, C_3, C_1C_2, C_2C_1, C_2C_2, $C_1C_1C_1$, C_4, C_1C_3, C_2C_3, C_3C_1, C_3C_2, C_3C_3... Denotemos con S_1, S_2, S_3, ... esta enumeración.



4. Por último, en virtud de (C), de la enumeración S_1, S_2, S_3, ... podemos extraer aquellos S_i que son métodos de generación. Obtenemos así la enumeración B_1, B_2, B_3,... de todos y solamente los métodos de generación.

Tenga presente el lector que estas cuatro enumeraciones deben llevarse a cabo en forma entrelazada. Siendo todas infinitas, no podríamos esperar a que una »termine« para comenzar en seguida con la otra.

Notemos que la enumeración B_1, B_2, B_3,... tiene la siguiente característica: existe un algoritmo tal que dado cualquier número entero positivo n, permite encontrar B_n. Dicho algoritmo consiste en llevar a cabo la enumeración recién descrita hasta el enésimo término. Correlativamente, existe un algoritmo tal que dado un método de generación en el lenguaje, permite encontrar el número n para el cual dicho método de generación es B_n. Este algoritmo consiste en llevar a cabo la enumeración B_1, B_2... hasta que aparezca el método de generación que interesa, siendo su lugar en la lista el número que se buscaba.

Terminaremos esta sección haciendo ver que no todos los conjuntos de enteros positivos

son efectivamente enumerables. Esto se concluye del hecho que los conjuntos de enteros positivos *son más* que los conjuntos efectivamente enumerables de enteros positivos.

¿Pero qué significa »son más«, puesto que hay infinitos conjuntos de enteros positivos y también hay infinitos conjuntos efectivamente enumerables de enteros positivos?

Acabamos de ver que el conjunto de todos los métodos de generación se puede generar en una lista B_1, B_2, B_3... Luego a cada método de generación podemos asociar un entero positivo: al método B_1, el número 1; al método B_2, el número 2, etc. Es decir, los enteros positivos 1, 2, 3 ... alcanzan para asociar uno a cada método de generación. Ahora, como a cada conjunto efectivamente enumerable corresponde (por definición) algún método de generación, entonces los enteros positivos 1, 2, 3... también alcanzan para asociar uno a cada conjunto efectivamente enumerable.

En cambio, de inmediato veremos que los enteros positivos 1, 2, 3... *no* alcanzan para asociar uno a cada conjunto de enteros positivos.

Supongamos por un momento que hemos logrado colocar a los conjuntos de enteros positivos en una lista P_1, P_2, P_3... de modo

que a cada conjunto asociamos un entero positivo. A P_1 asociamos el 1; a P_2 el 2; etcétera. En tal caso, necesariamente ha quedado fuera de nuestra lista el conjunto Q de enteros positivos que definimos a continuación: sea n un entero positivo cualquiera; entonces n pertenece a Q si y sólo si n no pertenece a P_n. Notemos que Q difiere de todos los conjuntos de la lista P_1, P_2, P_3... Luego Q es un conjunto de enteros positivos que se ha quedado sin número asociado, puesto que ya usamos todos los números enteros positivos para asociarlos a los conjuntos de la lista. Podemos concluir que, contrario a lo supuesto momentáneamente, es imposible lograr colocar a todos los conjuntos de enteros positivos en una lista de modo que a cada conjunto asociemos un entero positivo: siempre quedará fuera de la lista y, por ende, sin número asociado, un conjunto Q. Es decir, cualquiera que sea el modo en que tratemos de asociar un entero positivo a cada conjunto de enteros positivos, siempre sobrará al menos un conjunto de enteros positivos.

Queda así justificada nuestra afirmación original: los conjuntos de enteros positivos son más que los conjuntos efectivamente enumerables de enteros positivos; y esto a pesar de haber infinitos tanto de los unos como de los otros. El número de conjuntos efectivamente

enumerables de enteros positivos se denota con la expresión $\aleph_0$; y el número de conjuntos de enteros positivos se denota con la expresión $2^{\aleph_0}$.

3. GRADOS DE INSOLUBILIDAD

Recordemos ahora (habíamos encargado al lector la comprobación del hecho) que todo conjunto efectivamente computable es efectivamente enumerable. Luego, los conjuntos que no son efectivamente enumerables, tampoco son efectivamente computables. En la última sección acabamos de ver que existen conjuntos de enteros positivos que no son efectivamente enumerables. Por lo tanto, podemos concluir que existen conjuntos de enteros positivos que no son efectivamente computables. Los llamaremos *incomputables* o *insolubles.*

Ahora bien, a menudo es posible comparar dos conjuntos insolubles para encontrar que uno de ellos es »más difícil« o »más insoluble« que el otro. Veamos qué pueden significar estas expresiones.

Sean A y B conjuntos insolubles. Supongamos que existe un algoritmo que ante cualquier número entero positivo n, me permite encontrar un número entero positivo m de manera

tal que n está en A si y sólo si m está en B. Notemos que, en tal caso, si de alguna manera pudiéramos saber dado cualquier entero positivo, si está o no está en B, entonces podríamos saber, dado cualquier entero positivo, si está o no está en A. En efecto, para saber si un entero n está en A, haríamos lo siguiente: aplicando el algoritmo, obtendríamos m; en seguida, veríamos si m está o no está en B; de acuerdo con eso, sabríamos si n está o no está en A. Es decir, si bien es cierto que tanto A como B son insolubles, también sería cierto que *si* pudiéramos resolver B, entonces podríamos resolver A. Dada esta situación, decimos que el problema de decisión de A es reducible al problema de decisión de B. O, más simplemente, que *A es reducible a B.*

Ahora bien, si un conjunto insoluble A es reducible a un conjunto insoluble B, es razonable considerar que B es »más difícil« o »más insoluble« que A, puesto que *si* encontráramos un método para resolver B, ese mismo método nos daría la solución para A. En cambio un método para resolver A no nos daría necesariamente un método para resolver B... podríamos decir por ser B 'más difícil'.

De estas comparaciones nace el concepto de *grado de insolubilidad.* Sean A y B insolubles. Si A es reducible a B y B es reducible a A,

decimos que A y B tienen el mismo grado de insolubilidad. Si A es reducible a B, pero B no es reducible a A, decimos que B tiene un mayor grado de insolubilidad. Bien puede ocurrir que ni A sea reducible a B, ni B a A; en tal caso decimos que A y B son incomparables.

Resumamos lo dicho en los últimos párrafos: hemos deducido que existen conjuntos insolubles de enteros positivos; y hemos señalado que entre los conjuntos insolubles, hay unos que pueden considerarse más insolubles que otros. ¿Y qué interés tiene hacer estas distinciones entre los conjuntos insolubles? No olvidemos que a cada clase de problemas matemáticos corresponde un conjunto de enteros positivos; en caso de que el conjunto correspondiente a una clase de problemas sea incomputable, su grado de insolubilidad nos dirá mucho acerca de la estructura y complejidad de esa clase de problemas. Ilustremos este hecho con un interesante ejemplo.

Dado cualquier[1] sistema axiomático para la aritmética, sea V el conjunto de enteros positivos correspondientes a los enunciados

[1] En realidad, lo dicho vale para todo sistema axiomático que cumpla con *ciertos requisitos.* Pero se trata de requisitos muy naturales que, para todo efecto práctico, no restringen la generalidad de la afirmación, y en cuyo detalle no nos es necesario detenernos.

verdaderos que se pueden formular en el lenguaje del sistema, y sea T el conjunto de enteros positivos correspondientes a los teoremas del sistema. Entonces se puede demostrar que *V y T tienen distinto grado de insolubilidad.* De esto se desprende que V y T son conjuntos diferentes, de lo cual se concluye lo siguiente: es imposible construir un sistema axiomático cuyos teoremas comprendan todas y solamente las verdades de la aritmética. Como puede ver el lector, de la diferencia en los grados de insolubilidad de dos conjuntos numéricos, se concluye una importante limitación de los sistemas axiomáticos.

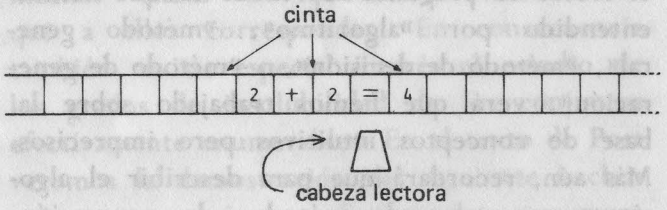
Recordemos ahora que los conjuntos efectivamente enumerables tienen especial interés por ser particularmente interesante muchas clases de problemas matemáticos que a ellos corresponden. En consecuencia, también reviste especial interés el estudio de los grados de insolubilidad de los conjuntos efectivamente enumerables. En el texto de Post veremos la demostración del siguiente hecho: existe un conjunto efectivamente enumerable cuyo grado de insolubilidad es el mayor de entre todos los conjuntos efectivamente enumerables. Dicho de otro modo, existe un conjunto efectivamente enumerable a cuyo problema de decisión se puede reducir el problema de decisión de cualquier conjunto efectivamente enumerable.

4. FUNCIONES RECURSIVAS, CONJUNTOS RECURSIVOS, CONJUNTOS RECURSIVAMENTE ENUMERABLES

El lector ha visto cómo hemos reemplazado los problemas sobre la matemática planteados inicialmente, por problemas acerca de conjuntos de enteros positivos. Pero en esta selección de conceptos matemáticos precisos que tomen el lugar de los conceptos no matemáticos iniciales, es decir, en esta construcción de un modelo matemático de los problemas originales, no hemos abordado aún el concepto básico que da sentido a todos los demás: a saber, el concepto de algoritmo. Si en este momento el lector se pregunta acerca de lo que hemos entendido por »algoritmo«, »método general«, »método de decisión«, o »método de generación«, verá que hemos trabajado sobre la base de conceptos intuitivos pero imprecisos. Más aún, recordará que para describir el algoritmo generador de métodos de generación, fue necesario *suponer* que ya disponíamos de una definición precisa de »método de generación«. En esta sección nos ocuparemos de completar en este aspecto fundamental nuestro modelo.

Definiremos, pero sin hacer una construcción formal, unos objetos llamados Máquinas de Turing. En base a estos objetos daremos una definición precisa de »algoritmo«. En seguida definiremos una clase de funciones: aquellas computables por máquinas de Turing (o, equivalentemente, por algoritmos). Finalmente, daremos un concepto preciso de »conjunto efectivamente computable« y de »conjunto efectivamente enumerable«.

Una máquina de Turing es una caja negra (entiéndase una máquina cuyo mecanismo interno no nos interesa, preocupándonos sólo su comportamiento externo) acoplada a una cinta infinita. La cinta está dividida en células. La cinta corre ante la cabeza lectora de la máquina, la cual puede leer una célula a la vez.



La máquina dispone de un alfabeto predeterminado. La máquina puede realizar las siguientes operaciones:

1. Escribir un símbolo en la célula que está leyendo, si esta célula se encuentra en blanco.

2. Borrar el símbolo que aparece en la célula que está leyendo.
3. Mover la cabeza lectora una célula a la izquierda.
4. Mover la cabeza lectora una célula a la derecha.

Exactamente cuál de estas operaciones realizará la máquina en un momento dado depende de:

1. El símbolo que se encuentre en la célula que está leyendo.
2. La configuración mecánica interna de la máquina en ese instante. Se dice también »el estado« en que se encuentra la máquina.
3. Lo que ordenen las instrucciones que la máquina está obedeciendo.

La máquina puede encontrarse en cualquiera de una cantidad finita de estados. Puede modificar el estado en que se encuentra, en obediencia de una instrucción.

La máquina comienza a trabajar con a lo más un número finito de células de la cinta impresa. Las demás células están en blanco. Lo impreso constituye los datos a los cuales la máquina aplicará las instrucciones. Lo que queda en la cinta cuando la máquina termina de operar (si es que termina) es el resultado de la computación.

Cada instrucción tiene la siguiente forma: »Encontrándose en tal estado y leyendo tal símbolo, realizar tal operación y colocarse en tal estado«. Luego, cada instrucción puede abreviarse en una cuádrupla de la forma:

(estado, símbolo, operación, estado)

Veamos un ejemplo.

ALFABETO: 1, B. (Aquí B significa célula en blanco).

ESTADOS: q_0, q_1, q_2, q_3.

INSTRUCCIONES:

$q_0 1 D q_1$ Interpretación: encontrándose en estado q_0 y leyendo 1, moverse una célula a la derecha y pasar al estado q_1.

$q_1 1 B q_2$ Interpretación: encontrándose en estado q_1 y leyendo 1, escribir B (o sea, borrar) y pasar al estado q_2.

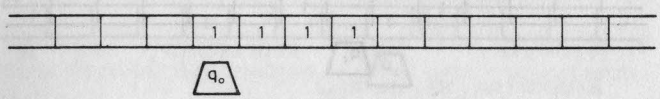
$q_2 B D q_1$ Interpretación: encontrándose en estado q_2 y leyendo B, moverse una célula a la derecha y pasar al estado q_1.

$q_1 B 1 q_3$ Interpretación: encontrándose en estado q, y leyendo B, escribir 1 y pasar al estado q_3.

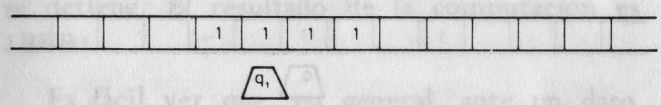
Nota. Se acata la siguiente convención: La máquina comienza a trabajar encontrándose en el estado q_0 y leyendo el símbolo (distinto de B que significa célula en blanco) impreso más a la izquierda.

Veamos qué hace esta máquina ante el dato 1111. Hará con la cinta lo siguiente:

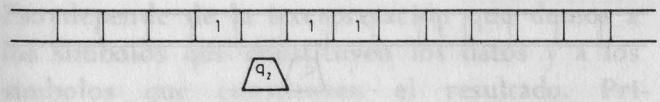
1. La máquina comienza leyendo el símbolo de más a la izquierda, encontrándose en estado q_0.



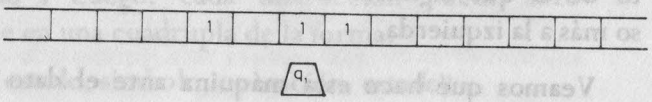
2. Obedece la primera instrucción, única aplicable al caso de encontrarse la máquina en estado q_0 leyendo 1. Queda



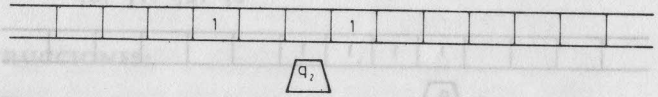
3. Obedece la segunda instrucción, única aplicable al caso de encontrarse la máquina en estado q_1 leyendo 1. Queda



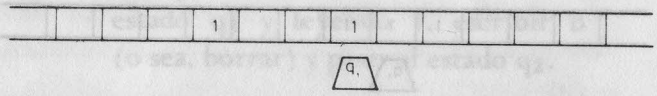
4. Obedece la tercera instrucción, única aplicable al caso de encontrarse la máquina en estado q_2 leyendo B. Queda.



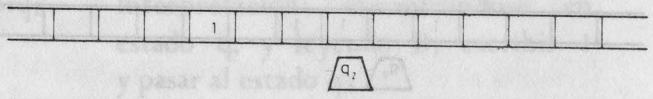
5. Obedece la segunda instrucción, única aplicable al caso de encontrarse la máquina en estado q_1 leyendo 1. Queda



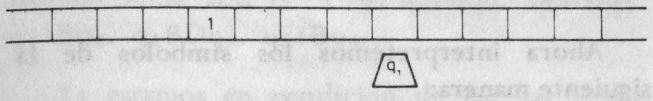
6. Obedece la tercera instrucción, única aplicable al caso de encontrarse la máquina en estado q_2 leyendo B. Queda



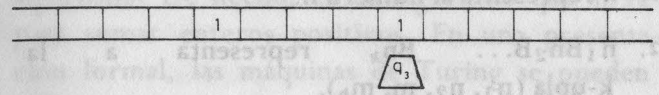
7. Obedece la segunda instrucción, única aplicable al caso de encontrarse la máquina en estado q_1 leyendo 1. Queda



8. Obedece la tercera instrucción, única aplicable al caso de encontrarse la máquina en estado q_2 leyendo B. Queda



9. Obedece la cuarta instrucción, única aplicable al caso de encontrarse la máquina en estado q_1 leyendo B. Queda



Notemos que ahora no quedan instrucciones por obedecer puesto que ninguna es aplicable al caso de encontrarse la máquina en estado q_3 leyendo 1. Por lo tanto, la máquina se detiene. El resultado de la computación es 1BBB1.

Es fácil ver que, en general, ante un dato 11...1 (n veces 1) la máquina arroja como resultado 1B...BB1 (n-1 blancos entre dos 1).

¿Y qué es lo que computa esta máquina? Eso depende de la interpretación que demos a los símbolos que constituyen los datos y a los símbolos que constituyen el resultado. Pri-

meramente acordemos la siguiente notación: para todo entero positivo n, $\bar{n}$ denotará una sucesión de n + 1 1's seguidos. Por ejemplo, $\bar{5}$ denota 111111 y $\bar{0}$ denota 1.

Ahora interpretemos los símbolos de la siguiente manera:

Datos (es decir, los símbolos que se encuentran en la cinta al comenzar a trabajar la máquina):

1. $\bar{n}$ representa al número n.
2. $\bar{n}_1 B \bar{n}_2 B \ldots B \bar{n}_k$ representa a la k-upla (n_1, n_2, m, m_k).

Resultados (es decir, los símbolos que se encuentran en la cinta al terminar la computación):

La sucesión de símbolos en la cinta representa al número de 1's que contiene.

Así, por ejemplo, el dato 111B111B11 representa el trío (2, 2, 1); el resultado 111B-111B11 representa el número ocho; el dato 1 representa el número cero; la cinta en blanco no representa dato alguno, pero representa el resultado cero.

Con esta interpretación resulta claro que la máquina de Turing de nuestro ejemplo computa

la función constante f(x) = 2. Asimismo podrá el lector comprobar que la máquina de Turing descrita por las siguientes cuádruplas computa la función suma f(x, y) = x + y; $q_0 1 B q_0$, $q_0 B D q_1$, $q_1 1 B q_2$, $q_1 B D q_3$, $q_3 1 B q_4$.

Ya estamos en condición de dar las primeras definiciones. Llamaremos *algoritmo* a todo conjunto de cuádruplas que defina una máquina de Turing. Así, por ejemplo, $q_0 1 B q_0$, $q_0 B D q_1$, $q_1 1 B q_2$, $q_1 B D q_3$, $q_3 1 B q_4$ es un algoritmo. De hecho, es un algoritmo que sirve para sumar enteros positivos. En una presentación formal, las máquinas de Turing se pueden *definir* como conjuntos de cuádruplas; luego, en rigor, los conceptos de algoritmo y máquina de Turing se identifican.

Llamaremos *función recursiva* a toda función que pueda computarse mediante una máquina de Turing. Así, sobre la base de los ejemplos que hemos dado, sabemos que la función constante f(x) = 2 y la función suma f(x, y) = x + y, son funciones recursivas. Suponiendo que la noción de máquina de Turing sea una adecuada versión matemática del concepto intuitivo de algoritmo, entonces las funciones recursivas constituyen una precisión adecuada del concepto intuitivo de función efectivamente computable, es decir, de función para el

cálculo de cuyos valores existe un algoritmo (en sentido intuitivo del término).

Por *conjunto recursivo* entenderemos todo conjunto que tenga función característica recursiva. Es decir, un conjunto A es recursivo si y sólo si la función f definida como sigue es recursiva: si n pertenece a A, $f(n) = 1$; si n no pertenece a A, $f(n) = 2$. Nuevamente, si el concepto de máquina de Turing refleja adecuadamente el concepto intuitivo de algoritmo, entonces el concepto de conjunto recursivo es una adecuada versión precisa del concepto de conjunto efectivamente computable.

Finalmente, definiremos *conjunto recursivamente enumerable* como todo conjunto que sea el recorrido o imagen de una función recursiva. Es decir, A es recursivamente enumerable si y sólo si existe una función recursiva f tal que ocurre lo siguiente: n pertenece a A si y sólo si $f(x) = n$ para algún entero positivo x. Nótese que en estas circunstancias la función recursiva f *genera* el conjunto A en el sentido de que A está compuesto precisamente por los números $f(1)$, $f(2)$, $f(3)$,... Luego, una vez más, si las máquinas de Turing son un buen modelo del concepto intuitivo de algoritmo, entonces los conjuntos recursivamente enumerables son un buen modelo de los conjuntos efectivamente enumerables.

Finalmente hemos completado nuestro modelo. Las funciones recursivas, los conjuntos recursivos y los conjuntos recursivamente enumerables constituyen una colección de funciones y conjuntos numéricos determinados con absoluta precisión: son aquellas funciones y aquellos conjuntos que se pueden definir de un modo claramente determinado y sobre la base del concepto formal de máquina de Turing. Con una combinación de arbitrariedad y buen criterio, como decíamos en un comienzo, hemos incorporado a estas funciones y a estos conjuntos a nuestro modelo sosteniendo que son, ni más ni menos, precisamente aquellas funciones y aquellos conjuntos que en términos intuitivos se consideran efectivamente computables o efectivamente enumerables, según corresponda.

Construido ya el modelo, el matemático se introduce de lleno en la segunda etapa de su labor: estudiar matemáticamente el modelo. Este es tal vez el quehacer matemático más practicado. De hecho, ni siquiera es necesario conocer cuáles fueron las motivaciones originales, para poder estudiar fructíferamente un modelo. Pero quien no ha perdido la perspectiva, tendrá siempre presente las motivaciones originales, y también las nuevas motivaciones que el propio modelo pueda sugerir, con el objeto de enfrentar a la realidad de la cual partió, las

conclusiones que extraiga del estudio del modelo.

El lector que tenga presente lo dicho en el párrafo anterior, lo dicho sobre modelos en la introducción y, en definitiva, todo el recorrido que ha hecho desde la introducción hasta el párrafo anterior, comprenderá fácilmente que detrás de un título de apariencia técnica, especializada y hasta pequeña como »Conjuntos recursivamente enumerables de enteros positivos y sus problemas de decisión«, se encuentra un estudio riguroso de problemas cuyo interés es general y cuya comprensión está al alcance de todos. Y lo mismo, aunque ello no conste de lo expuesto, vale para la matemática en general: detrás de una apariencia técnica y especializada se hace frente a inquietudes profundas de interés general y fácil comprensión.

Capítulo Segundo

Conjuntos recursivamente enumerables de enteros positivos y sus problemas de decisión

EMIL POST

Introducción. Desarrollos recientes de la lógica simbólica tienen importancia considerable para la matemática tanto respecto de su filosofía como de su práctica. Que los matemáticos generalmente desatiendan la importancia de esta labor de Gödel, Church, Turing, Kleene, Rosser y otros en cuanto afecta las materias de su propio interés se debe en parte al formalismo diferente, prohibitivo y extraño en que esta labor ha tomado cuerpo. No obstante, sin este formalismo, esta labor creadora de nuevas perspectivas perdería la mayor parte de su poder de convicción. Pero aparte de cuestiones sobre importancia, estos formalismos traen a la matemática un concepto matemático nuevo y preciso, el de función recursiva general de Herbrand - Gödel - Kleene, o sus equivalentes comprobados en los desarrollos de Church y Turing. Es el propósito de esta disertación demostrar por vía de ejemplo que este concepto es susceptible de ser desarrollado e incorporado a una teoría matemática, tal como el concep-

to de grupo se ha desarrollado e incorporado a una teoría de grupos. Más aún, que despojada de su formalismo, tal teoría admite un desarrollo intuitivo, que puede ser seguido, si no incluso impulsado, por un matemático, aun si es lego en este campo formal. Es este desarrollo intuitivo de una limitada porción de una subteoría de la deseada teoría general, que presentamos en esta disertación. Debemos enfatizar que, con las pocas excepciones explícitamente indicadas, hemos obtenido demostraciones formales de todos los teoremas, por ende matemáticos, que aquí se desarrollan informalmente. Pero la verdadera matemática involucrada se encuentra en el desarrollo informal. Pues en toda ocasión la »demostración« informal se obtuvo primero; una vez obtenida, convertirla en demostración formal, resultó un quehacer rutinario.

No reproduciremos aquí la definición formal de *función recursiva de enteros positivos.* Un ejemplo simple de tal función es un polinomio arbitrario $P(x_1, \ldots, x_n)$, con, digamos, coeficientes enteros no negativos, y no idénticamente cero. Si a las x se asignan valores enteros positivos arbitrarios expresados, por ejemplo, en notación arábica, los algoritmos de adición y multiplicación en esa notación nos permiten calcular el valor entero positivo correspondiente del polinomio. Es decir, $P(x_1, \ldots, x_n)$ es una *función efectivamente computable de enteros positivos.* La importancia del concepto técnico función recursiva emana de los abrumadores indicios en el sentido de ser dicho concepto coextensivo con el concepto intuitivo de función efectivamente computable.

Un conjunto de enteros positivos se dice *recursivamente enumerable* si existe una función recursiva f(x) de una variable entera positiva cuyos valores, para valores enteros positivos de x, constituyen dicho conjunto. La sucesión f(1), f(2), ... se dice entonces una *enumeración recursiva* del conjunto. El concepto intuitivo correspondiente es el de un conjunto *efectivamente enumerable* de enteros positivos. Para prepararnos en parte para nuestro enfoque intuitivo, consideremos los siguientes tres ejemplos de conjuntos recursivamente enumerables de enteros positivos.

(a): $1^2, 2^2, 3^2, \ldots$

(b): $1, 2, 2^{1 \cdot 2}, 2^{1 \cdot 2 + 2^{1+2}}, \ldots$

(c): $1^2, 2^2, 3^2, \ldots$

$1^3, 2^3, 3^3, \ldots$

$1^4, 2^4, 3^4, \ldots$

. . . . . .

. . . . . .

. . . . . .

En el primer ejemplo, el conjunto está dado por una enumeración recursiva suya mediante la función recursiva x^2. En el segundo ejemplo, el conjunto está generado en una sucesión lineal, cada nuevo elemento obteniéndose efectivamente de los elementos generados anteriormente, en este caso elevando 2 a la suma de los elementos precedentes. El conjunto es efectivamente enumerable, ya que el n-ésimo elemento de la sucesión puede encontrarse, dado n, regenerando la sucesión a través de sus primeros n elementos. En el tercer ejemplo, más bien imaginamos los enteros positivos 1, 2, 3,... generados en su orden natural, y, a medida que cada entero positivo n se genera, un proceso correspondiente se echa a andar que genera n^2, n^3, n^4,... todos éstos al pertenecer al conjunto. De hecho, el método corriente para demostrar que un conjunto enumerable de conjuntos enumerables es enumerable, da una enumeración efectiva del conjunto.

Varios ejemplos más debieran ofrecerse para dar a entender la idea que tiene el autor de *conjunto generado,* en este momento de enteros positivos. Baste decir que cada elemento del conjunto se escribe en algún momento, quedando constancia de su pertenencia al conjunto, como consecuencia de procedimientos efectivos predeterminados. Se entiende que una vez que un elemento se ubica en el conjunto, ahí queda. En otra ocasión el autor ha hecho referencia a una generalización que puede reformularse *todo conjunto generado de enteros positivos es recursivamente enumerable.* Para efectos de comparación esto puede desmembrarse en dos afirmaciones: todo conjunto generado es efectivamente enumerable, todo conjunto efectivamente enumerable de enteros positivos es recursivamente enumerable. La primera de estas afirmaciones es aplicable a conjuntos generados de expresiones simbólicas arbitrarias; se ve fácilmente que sus inversos son verdaderos. El concepto expuesto y la generalización nos resultarán muy útiles en nuestro desarrollo intuitivo. Pero cuando a menudo digamos, explícita o implícitamente, »tal conjunto de enteros positivos es generado, y por lo tanto recursivamente enumerable«, para nuestros propósitos ello significará tan sólo »intuitivamente se ha mostrado que tal conjunto es generado; se puede demostrar que es recursivamente enumerable«. Asimismo ocurrirá en el caso de otras identificaciones de conceptos formales correspondientes que han sido definidos matemáticamente.

En algunos puntos de nuestro desarrollo informal debemos apoyarnos en el desarrollo

formal. Este último es otro formalismo más debido al autor pero que ha sido demostrado equivalente al de función recursiva general. Bastará dar el equivalente de »conjunto recursivamente enumerable de enteros positivos« en este desarrollo.

Un entero positivo n se puede representar de la manera más primitiva como una sucesión 11...1 de n rayas. Por motivos de trabajo, introducimos la letra b, y consideramos »cadenas« de 1's y b's tales como 11b1bb1. Una operación sobre tales cadenas como ser »b1bP produce P1bb1«, la llamaremos una operación normal. Esta operación normal particular sólo es aplicable a cadenas que comienzan con b1b, y la cadena derivada se obtiene de la cadena dada quitando el b1b inicial y agregando1bb1 al final. Así, b1bb deviene b1bb1. »gP produce Pg'« es la forma de una operación normal arbitraria. Un sistema en forma normal, o sistema normal, está dado por una cadena inicial A de 1's y b's, y un conjunto finito de operaciones normales »g_iP produce Pg_i'«, i = 1, 2,..., m. Las cadenas derivadas del sistema son A y todas las cadenas que se pueden obtener de A mediante aplicaciones reiteradas de las m operaciones. Cada sistema normal define unívocamente un conjunto, posiblemente vacío, de enteros positivos, a

saber los enteros representados por aquellas cadenas de 1's solamente. En seguida, puede demostrarse que todo conjunto recursivamente enumerable es el conjunto de enteros positivos definido por algún sistema normal, y viceversa. *Aquí, como más abajo, extendemos arbitrariamente el concepto de recursivamente enumerable para incluir el conjunto vacío.* Por *base* B de un sistema normal, y del conjunto recursivamente enumerable de enteros positivos que define, entendemos la sucesión de letras y símbolos que aquí representamos con

A;g_1P produce P$g'_1, \ldots, g'_m$.

Interpretando correctamente, B determina el sistema normal, y el conjunto recursivamente enumerable de enteros positivos, en cuestión. Cada base no es más que una sucesión finita de los símbolos 1, b, P, la coma, el punto y coma y las letras de la palabra »produce«. El conjunto de bases es, por lo tanto, infinito enumerable, y puede ser generado en una sucesión de elementos distintos

O: $B_1, B_2, B_3, \ldots$

Como cada B_i define un único conjunto recursivamente enumerable de enteros positivos y cada tal conjunto está definido por al menos un B_i, O es también una ordenación de to-

dos los conjuntos recursivamente enumerables de enteros positivos, aunque cada conjunto de hecho aparecerá infinitas veces en O. Podemos entonces decir, en términos clásicos, que mientras hay $2^{\aleph_0}$ conjuntos arbitrarios de enteros positivos, hay tan sólo $\aleph_0$ conjuntos recursivamente enumerables.

Por el *problema de decisión* de un conjunto dado de enteros positivos entendemos el problema de efectivamente determinar para un entero positivo dado arbitrariamente, si está o no está en el conjunto. Mientras, en cierto sentido, la teoría de conjuntos recursivamente enumerables de enteros positivos es potencialmente tan amplia como la teoría de funciones recursivas generales, los problemas de decisión de tales conjuntos constituyen una clase muy especial de problemas de decisión. No obstante, son importantes, como se muestra en los siguientes ejemplos especiales y generales.

Uno de los problemas planteados por Hilbert en su disertación de París, de 1900, es el problema de determinar para una ecuación diofántica arbitraria con coeficientes enteros racionales, si tiene o no tiene una solución entera racional. Si las variables de una ecuación diofántica se escogen de un conjunto infinito enumerable dado de variables, es claro que el

conjunto de las ecuaciones diofánticas es infinito enumerable. De hecho ellas pueden colocarse efectivamente en correspondencia uno a uno con el conjunto de enteros positivos. Como dada cualquier ecuación diofántica y cualquier asignación de valores enteros racionales a sus variables, es posible determinar efectivamente si la ecuación queda satisfecha para dichos valores, es posible generar el conjunto de las ecuaciones diofánticas con soluciones enteras racionales. Los enteros correspondientes, en esta correspondencia uno a uno, pueden entonces también generarse y, de hecho, constituyen un conjunto recursivamente enumerable de enteros positivos. Y bajo dicha correspondencia, el problema de Hilbert se transforma en el problema de decisión de ese conjunto recursivamente enumerable.

Las oraciones de una lógica simbólica arbitraria, constituyen un conjunto generado A de lo que podemos llamar complejos de símbolos, o fórmulas. Suponemos que A es un subconjunto de un conjunto infinito generado E de complejos de símbolos, que en un caso puede ser el conjunto de enunciados con sentido de la lógica, en otro el conjunto de todos los complejos de símbolos de una determinada simbología. El problema de decisión de la lógica, más precisamente su problema de deducibi-

lidad, es entonces el problema de determinar para un miembro arbitrario de E si está o no está en A. Aceptando que todo conjunto generado es efectivamente enumerable, los miembros de E pueden efectivamente colocarse en correspondencia uno a uno con el conjunto de los enteros positivos. Los enteros positivos correspondientes a miembros de A constituyen entonces un conjunto generado y, por ende, bajo nuestra generalización, recursivamente enumerable de enteros positivos. Y bajo dicha correspondencia el problema de decisión de la lógica simbólica se transforma en el problema de decisión de este conjunto recursivamente enumerable de enteros positivos.

Estrechamente vinculado al concepto técnico de conjunto recursivamente enumerable de enteros positivos, está aquel de conjunto *recursivo* de enteros positivos. Es un conjunto para el cual existe una función recursiva f(x) tal que f(x) es, digamos, 2 cuando x es un entero positivo perteneciente al conjunto, 1 cuando x es un entero positivo no perteneciente al conjunto. Podemos también hacer de esto la definición de ser *recursivamente soluble* el problema de decisión del conjunto. Puesto que 2 y 1 pueden considerarse como los dos posibles valores de verdad, verdadero, falso, de la afirmación »el entero positivo x está en el conjunto«,

y la definición de conjunto recursivo equivale a que este valor de verdad sea recursivamente calculable para todos los enteros positivos x. Entonces, si función recursiva es coextensivo con computabilidad efectiva, solubilidad recursiva es coextensivo con solubilidad en sentido intuitivo. En particular, el problema de decisión de un conjunto recursivamente enumerable sería soluble o insoluble según el conjunto sea o no sea recursivo. En términos más generales que en nuestras dos ilustraciones, a través del mecanismo más preciso de las representaciones de Gödel, una amplia variedad de problemas de decisión y otros problemas se transforman en problemas sobre enteros positivos; y que esos problemas sean o no solubles en sentido intuitivo, equivale a que sean o no recursivamente solubles en sentido técnico preciso.

El teorema clásico de Gödel sobre la incompletitud y extendibilidad de lógicas simbólicas, casi lo llevó a la insolubilidad recursiva de una generalización del problema antedicho de Hilbert. Church formuló explícitamente el concepto de insolubilidad recursiva, y llegó a la insolubilidad de varios problemas; naturalmente, demostró que eran recursivamente insolubles. El problema antedicho de Hilbert clama por una demostración de insolubilidad. Al igual que las demostraciones clásicas de inso-

lubilidad, estas demostraciones son de insolubilidad mediante instrumentos dados. La novedad del caso actual es que estos instrumentos, en efecto, parecen ser los únicos instrumentos a disposición del hombre.

Relacionada a la cuestión de solubilidad o insolubilidad de problemas, está aquella de la reducibilidad o no reducibilidad de un problema a otro. Así, si el problema P_1 ha sido reducido al problema P_2, una solución de P_2 de inmediato da una solución de P_1, mientras que si se demuestra que P_1 es insoluble, P_2 también debe ser insoluble. Para problemas insolubles el concepto de reducibilidad lleva al concepto de *grado de insolubilidad,* siendo dos problemas insolubles del mismo grado de insolubilidad si cada uno es reducible al otro, siendo uno de menor grado de insolubilidad que otro, si es reducible a ese otro, pero sin ser ese otro reducible al primero, siendo de grados incomparables de insolubilidad si ninguno es reducible al otro. Un problema básico de la teoría de conjuntos recursivamente enumerables es el problema de determinar los grados de insolubilidad de sus problemas de decisión insolubles. Pronto veremos que para tales problemas existe ciertamente un grado que es el más alto de insolubilidad. Todo nuestro desarrollo se centra en gran

medida en la sola cuestión acerca de si hay, entre estos problemas, un grado de insolubilidad menor que aquél, o si bien todos tienen el mismo grado de insolubilidad. Ahora en su trabajo sobre *lógicas ordinales,* Turing presenta como cuestión lateral una formulación que puede ser expresada como la formulación general de la »reducibilidad recursiva« de un problema a otro, y demuestra un resultado que fácilmente se generaliza a la conclusión de que para cualquier problema insoluble »dado recursivamente« hay otro de grado más alto de insolubilidad. Si bien este teorema no nos ayuda en nuestra búsqueda de aquel menor grado de insolubilidad, su formulación hace preciso nuestro problema. Sigue siendo un problema al término de este trabajo. Pero en el camino obtenemos varios resultados particulares, y hacia el final logramos una idea de las dificultades del problema general.

I. CONJUNTOS RECURSIVOS Y CONJUNTOS RECURSIVAMENTE ENUMERABLES

La relación entre ambos conceptos queda de manifiesto en el siguiente

TEOREMA. *Un conjunto de enteros positivos es recursivo si y sólo si tanto el conjunto como su com-*

plemento con respecto al conjunto de todos los enteros positivos son recursivamente enumerables.

Para mayor simplicidad, supongamos que tanto el conjunto S como su complemento $\overline{S}$ son infinitos. Entonces, si S es recursivo existe un método efectivo para determinar si un entero positivo cualquiera n está o no está en S. Generemos los enteros positivos 1, 2, 3,... en su orden natural y, a medida que se genera cada entero positivo, determinamos si está o no está en S. Cada vez que descubramos que un entero positivo está en S, escribámoslo en una lista que contendrá los elementos de S. Así, hemos establecido un procedimiento efectivo para enumerar efectivamente los elementos de S. Luego, S es recursivamente enumerable. Asimismo se puede demostrar que $\overline{S}$ es recursivamente enumerable.

A la inversa, sean S y $\overline{S}$ recursivamente enumerables, y sea $n_1, n_2, n_3, \ldots$ una enumeración recursiva de S, y $m_1, m_2, m_3, \ldots$ una enumeración recursiva de $\overline{S}$. Dado un entero positivo n, generemos en orden $n_1, m_1, n_2, m_2, n_3, m_3$, y así sucesivamente, comparando cada número con n. Como n tiene que estar en S o bien en $\overline{S}$, en un número finito de pasos encontraremos un n_i o un m_j idéntico a n, y consecuencialmente descubriremos

si n está en S o en $\overline{S}$. Así hemos establecido un método efectivo para determinar si un entero positivo cualquiera está o no está en S. Luego, S es recursivo.

COROLARIO. *El problema de decisión de un conjunto recursivamente enumerable es recursivamente soluble si y sólo si su complemento es recursivamente enumerable.*

Pues en tal caso y sólo en tal caso es recursivo dicho conjunto recursivamente enumerable.

Fácilmente se demuestra que la suma lógica y el producto lógico de dos conjuntos recursivamente enumerables es recursivamente enumerable, que el complemento de un conjunto recursivo, la suma lógica, y por lo tanto el producto lógico, de dos conjuntos recursivos es recursivo.

Claramente, *cualquier conjunto finito de enteros positivos es recursivo.* Pues si $n_1, n_2, \ldots, n_v$ son dichos enteros, podemos determinar si n está o no está en el conjunto mediante su comparación directa con $n_1, n_2, \ldots, n_v$. Asimismo ocurre con cualquier conjunto cuyo complemento sea finito. Para conjuntos infinitos arbitrarios tenemos el siguiente resultado de Kleene. *Un conjunto infinito de enteros positivos es recursivo si y sólo si es susceptible de*

enumeración recursiva sin repeticiones y en orden de magnitud. En efecto, si n_1, n_2, n_3, ... es una enumeración recursiva de S sin repeticiones y en orden de magnitud, todos los n_i más allá del enésimo deben ser mayores que n. Luego podemos determinar si n está o no está en S, generando los primeros n miembros de dicha enumeración recursiva de S, y viendo si n es o no es uno de ellos. A la inversa, si S infinito es recursivo, la enumeración recursiva de sus elementos que dimos en la demostración de nuestro primer teorema, es una enumeración de los elementos de S sin repetición y en orden de magnitud.

Una consecuencia directa de la primera mitad del último resultado es el siguiente

TEOREMA. *Todo conjunto recursivamente enumerable infinito contiene un conjunto infinito recursivo.*

Pues si n_1, n_2, n_3, ... es una enumeración recursiva de un conjunto infinito S, para cada n_i debe existir, en esta sucesión, un $n_j > n_i$ posterior. Luego, generemos los elementos n_1, n_2, n_3, ... en orden, y sea $m_1 = n_1$, $m_2 = n_{i_2}$, el primer n_i mayor que n_1, $m_3 = n_{i_3}$, el primer n_i posterior a n_{i_2} y mayor que n_{i_2}, y así sucesivamente. La sucesión

m_1, m_2, m_3, ... es entonces una enumeración recursiva de un subconjunto de S sin repeticiones y en orden de magnitud. Dicho subconjunto es, por lo tanto, infinito y recursivo.

Básico para toda la teoría es el siguiente resultado por el cual debemos dar crédito conjuntamente a Church, Rosser y Kleene

TEOREMA. *Existe un conjunto recursivamente enumerable de enteros positivos que no es recursivo.*

Por nuestro primer teorema esto equivale a la existencia de un conjunto recursivamente enumerable de enteros positivos cuyo complemento no es recursivamente enumerable. Generemos en orden las distintas bases B_1, B_2, B_3, ... de todos los conjuntos recursivamente enumerables de enteros positivos, según las mencionamos en la introducción, y llevemos cuenta de estas bases como la primera, la segunda, la tercera, y así sucesivamente, en esta enumeración O. Al generarse la enésima base B_n, con n= 1, 2, 3, ... echemos a andar los procedimientos mediante los cuales el correspondiente conjunto recursivamente enumerable es generado, y toda vez que n sea generado por B_n, coloquemos n en un conjunto U. Como U es un conjunto generado de enteros positivos, es recursivamente enumerable.

Un entero positivo n, entonces, está o no está en U según esté o no esté en el enésimo conjunto recursivamente enumerable en O considerado como una ordenación de todos los conjuntos recursivamente enumerables. Luego n está o no está en $\overline{U}$, el complemento de U, según no esté o esté en el enésimo conjunto de O. Vemos así que $\overline{U}$ difiere de cada conjunto recursivamente enumerable en cuanto a la presencia o ausencia de al menos un entero positivo. Luego $\overline{U}$ no es recursivamente enumerable.

COROLARIO. *Existe un conjunto recursivamente enumerable de enteros positivos cuyo problema de decisión es recursivamente insoluble.*

Tomados individualmente, los conjuntos finitos, o los conjuntos cuyo complemento es finito, son ejemplos más bien triviales de conjuntos recursivos. Por otra parte, si decimos que dos conjuntos de enteros positivos son *abstractamente* iguales en caso que uno pueda transformarse en el otro, mediante una transformación recursiva uno a uno del conjunto de todos los enteros positivos en sí mismo, entonces todos los conjuntos recursivos infinitos con complemento infinito son *abstractamente* iguales. Siendo nuestra teoría esencialmente una teoría abstracta de conjuntos recursivamente enumerables, nuestro interés se centra en conjuntos recursivamente enumerables que no son recursivos. Tales conjuntos, así como sus complementos, son siempre infinitos. No nos ocupamos más de la cuestión de ser dos conjuntos abstractamente iguales, ya que ello no es más que un caso especial de ser cada conjunto reducible uno a uno, al otro.

2. UNA FORMA DEL TEOREMA DE GÖDEL

Dada una base cualquiera B, y un entero positivo n, el par (B, n) se puede usar para representar la proposición, verdadera o falsa, »n está en el conjunto generado por B«. Entrelazando los procedimientos para generar las distintas bases en la sucesión B_1, B_2, B_3, ... y los procedimientos para generar los enteros positivos en la sucesión 1, 2, 3, ... mediante la adición de 1's, podemos efectivamente generar los pares distintos (B, n) en la sola sucesión infinita

$$O': (B_1,1), (B_2,1), (B_1,2), (B_3,1), (B_2,2), (B_1,3), \ldots$$

Por otra parte, el conjunto de todos los pares (B, n) es así un conjunto generado de expresiones al cual llamaremos E. Por otra parte, O' lleva a una correspondencia uno a uno efectiva

entre los miembros de E y el conjunto de los enteros positivos, (B, n) correspondiendo a m si (B, n) es el emésimo miembro de O'. Podemos llamar a m la representación de Gödel de (B, n). Dado un subconjunto generado de E, las representaciones de Gödel de sus miembros constituirán un conjunto generado de enteros positivos, y viceversa. Así, en el primer caso podemos generar los miembros del subconjunto de E y, al generarse cada par (B, n), encontrar su representación de Gödel m regenerando O'. El conjunto de estos m's es así un conjunto generado. Asimismo en el caso inverso. Por lo tanto, si formalmente decimos que un subconjunto de E es recursivamente enumerable, si el conjunto de representaciones de Gödel de sus miembros es recursivamente enumerable, podemos concluir que todo subconjunto generado de E es recursivamente enumerable y, naturalmente, viceversa. Análogamente ocurre con una definición formal similar de subconjunto recursivo de E.

Si bien E es tan sólo el conjunto de pares (B, n), puede interpretarse como el conjunto de enunciados »n está en el conjunto generado por B«. El subconjunto T de E constituido por aquellos pares (B, n) para los cuales n *está* en el conjunto generado por B, puede entonces interpretarse como el conjunto de afir-

maciones verdaderas que contiene E, mientras que $\overline{T}$, el complemento de T con respecto a E, está constituido por las afirmaciones falsas que hay en E.

Ahora, T mismo puede generarse como sigue. Generemos B_1, B_2, B_3, ... en orden. Al generarse cada B, echemos a andar el procedimiento para generar el conjunto de enteros positivos determinado por B, y, toda vez que se genere de este modo un entero positivo n, escribamos el par (B, n). De esta manera cada (B, n) para el cual n pertenece al conjunto generado por B será anotado, y viceversa. Este conjunto generado de (B, n)'s es entonces T. Luego podemos concluir que *T es recursivamente enumerable.*

Ahora, sea F cualquier subconjunto recursivamente enumerable de $\overline{T}$. Si (B, n) está en F, está en $\overline{T}$ y, por lo tanto, n no está en el conjunto generado por B. Ahora generemos los miembros de F, y al generarse así un (B, n), encontremos el enésimo miembro de O: B_1, B_2, B_3, ... y si B_n es B, coloquemos n en un conjunto de enteros positivos S_0. Como S_0 es así un conjunto generado de enteros positivos, es recursivamente enumerable. Luego debe estar determinado por alguna base B. Sea esta base la v-ésima en O, es decir, sea esta base B_v,

y formemos el par (B_v, v). Ahora, por construcción, S_0 está compuesto por aquellos miembros de F de la forma (B_n, n). Supongamos que (B_v, v) está en F. Entonces, por una parte, siendo falso el enunciado (B_v, v), v no podría estar en el conjunto generado por B_v, es decir (1): v no estaría en S_0. Pero siendo (B_v, v) de la forma (B_n, n), (2): v estaría en S_0. Como nuestra suposición nos ha llevado a una contradicción, se desprende que (B_v, v) *no está en F.* Pero v sólo puede estar en S_0 si (B_v, v) está en F. Luego, v *no está en* S_0. Por último, (B_v, v) como enunciado dice que v está en S_0. El enunciado (B_v, v) es por lo tanto falso, o sea, (B_v, v) *está en* $\overline{T}$.

Luego, para cualquier subconjunto recursivamente enumerable F de $\overline{T}$, siempre existe un par (B_v, v) en $\overline{T}$ que no está en F. Por otra parte, entonces, $\overline{T}$ nunca puede ser F. Luego $\overline{T}$ *no es recursivamente enumerable.* De las definiciones de esta sección y del primer teorema de la sección anterior, se sigue que T, *siendo recursivamente enumerable, no es recursivo.* Por el problema de decisión de T, entendemos el problema de determinar para cualquier miembro de E, si está o no en T. Pero esto puede interpretarse como el problema de decisión de la clase de conjuntos recursivamente

enumerables de enteros positivos, es decir, el problema de determinar para cualquier conjunto recursivamente enumerable, o sea para cualquier base B de un tal conjunto, y cualquier entero positivo n, si n está o no está en el conjunto generado por B. Podemos entonces decir que el problema de decisión para la clase de todos los conjuntos recursivamente enumerables de enteros positivos es recursivamente insoluble, y por lo tanto, muy probablemente, insoluble en sentido intuitivo.

Por otra parte, como (B_v, v) está en $\overline{T}$ pero no en F, T y F conjuntamente nunca pueden cubrir E. Ahora, T, o cualquier subconjunto, recursivamente enumerable T' de T, en conjunto con F, puede llamarse una lógica generada recursivamente, con respecto a la clase de enunciados E. Pues al aparecer (B, n) en T' se nos garantiza la verdad de la proposición »n está en el conjunto generado por B«, mientras que si aparece en F se nos garantiza su falsedad. Podemos entonces decir que *ninguna lógica generada recursivamente, con respecto a E, es completa,* puesto que F sólo nos lleva al (B_v, v) que no está ni en T' ni en F. Es decir, (B_v, v) es indecidible en esta lógica. Más aún, si, dada una base para F, se desarrolla formalmente el argumento que hemos dado, el conjunto recursivamente enumerable S_0 ob-

tenido estará determinado por una base específica B que puede construirse mediante dicho desarrollo formal. Encontrada B podemos generar O: B_1, B_2, B_3,... hasta alcanzar B, y así determinar el v tal que $B = B_3$,... Es decir, dada una base para F, el (B_v, v) que está en $\overline{T}$ pero no en F puede de hecho encontrarse. Si entonces agregamos este (B_v, v) a F, obtenemos un subconjunto recursivamente enumerable F' de $\overline{T}$, más amplio que F. Podemos entonces decir que *toda lógica generada recursivamente con respecto a E, puede extenderse.* A primera vista, estos dos resultados, al desarrollarse formalmente, parecen ser el teorema de Gödel en miniatura. Pero en virtud de la generalidad del concepto técnico de función recursiva general, ellos, muy probablemente, justifican implícitamente la generalización en el sentido de que toda lógica simbólica es incompleta y extendible, respecto de la clase de enunciados que constituyen E. Es inevitable concluir que, aún para un conjunto tan preciso y bien delineado de proposiciones matemáticas, *el pensamiento matemático es y seguirá siendo esencialmente creativo.* Al parecer del autor, esta conclusión traerá inevitablemente como consecuencia un cambio al menos parcial en el rumbo axiomático de fines del siglo diecinueve y comienzos del veinte,

volviéndose a considerar significado y verdad como esenciales a la matemática.

3. EL CONJUNTO COMPLETO K; CONJUNTOS CREATIVOS

Volvamos ahora a la correspondencia efectiva uno a uno entre el conjunto E de los distintos (B, n)'s y el conjunto de los enteros positivos, obtenida por medio de la enumeración efectiva O' de E. Como T es un subconjunto recursivamente enumerable de E, los enteros positivos correspondientes a los elementos de T constituyen un conjunto recursivamente enumerable de enteros positivos, K. Llamaremos a K el *conjunto completo.* Como $\overline{T}$ no es recursivamente enumerable, $\overline{K}$, que está compuesto por los enteros positivos correspondientes a los elementos de $\overline{T}$, no es recursivamente enumerable. Ahora, sea B la base de un subconjunto recursivamente enumerable, ç, de $\overline{K}$. Los elementos de E correspondientes a los miembros de ç constituyen, entonces, un subconjunto recursivamente enumerable F de $\overline{T}$. Encontremos entonces el (B_v, v) en $\overline{T}$ que no está en F y, mediante O', el entero positivo n correspondiente a (B_v, v). Este n será entonces un elemento de $\overline{K}$ que no está en ç.

Ahora, ni disponemos de método general

para saber cuando una base B define un subconjunto recursivamente enumerable de $\overline{K}$. En realidad, el método visto arriba nos da un entero positivo único para cualquier base B de un conjunto recursivamente enumerable ç de enteros positivos. Pero, cuando ç es un subconjunto de $\overline{K}$, n también estará en $\overline{K}$, pero no en ç.

Más aún, incluso la demostración formal de este resultado nos da tan sólo un método efectivo para encontrar n, dado B. Pero este método mismo se puede formalizar de modo que, consecuencialmente, n se obtiene como »función recursiva de B«. Esto puede significar que se puede determinar una función recursiva f(m) tal que $n = f(m)$, con $B = B_m$. Aislemos ahora esta propiedad de K formulando una

DEFINICION. *Un conjunto creativo C es un conjunto recursivamente enumerable de enteros positivos para el cual existe una función recursiva que da un entero positivo único n para cada base B de un conjunto recursivamente enumerable de enteros positivos ç, de modo que si ç es subconjunto de $\overline{C}$, entonces n está en $\overline{C}$ pero no en ç.*

TEOREMA. *Existe un conjunto creativo, a saber el conjunto completo K.*

En realidad, la clase de conjuntos creativos es

infinita, y de hecho muy rica como se ve en los siguientes resultados fáciles de demostrar. Si C es un conjunto creativo y E es un conjunto recursivamente enumerable de enteros positivos, entonces si E contiene a $\overline{C}$, CE es creativo, y si $\overline{C}$ contiene a E, C + E es creativo. Ahora, los resultados de la primera sección nos permiten construir conjuntos creativos de acuerdo con el primer método usando E's que son complementos de subconjuntos recursivos de C. Los resultados del resto de esta sección llevan a construcciones usando el segundo método.

Resulta conveniente hablar en términos de que n, en la definición de conjunto creativo, está determinado por ç en lugar de por la base B de ç. Claramente, *todo conjunto creativo C es un conjunto recursivamente enumerable que no es recursivo.* Pues si $\overline{C}$ fuera recursivamente enumerable, no podría existir un n en $\overline{C}$ que no perteneciera al subconjunto recursivamente enumerable $\overline{C}$ de $\overline{C}$. Por lo tanto, *el problema de decisión de cada conjunto creativo es recursivamente insoluble.* Por otra parte, *el complemento $\overline{C}$ de cualquier conjunto creativo C contiene un conjunto recursivamente enumerable infinito.* Recordemos que todo conjunto finito es recursivo, y luego recursivamente enumerable. Entonces, tomando ç, de la definición de conjunto creativo, como el conjunto vacío, podemos encontrar

el $n = n_1$ en $\overline{C}$ que no está en ç. Ahora tomando ç como el conjunto cuyo único elemento es n_1, $n = n_2$ estará en $\overline{C}$ y será distinto de n_1. Formando ç con n_1 y n_2, $n = n_3$ estará en $\overline{C}$, y será distinto de n_1 y n_2, y así sucesivamente. El conjunto de enteros positivos $n_1, n_2, n_3, \ldots$ es entonces un subconjunto infinito de $\overline{C}$, generado y, por lo tanto, recursivamente enumerable.

Ahora, tomando este subconjunto de $\overline{C}$ como ç, se obtiene un nuevo elemento n_{ω} de $\overline{C}$, y así sucesivamente hacia el transfinito constructivo. Pero este proceso es esencialmente creativo. Pues todo procedimiento mecánico sólo podría producir n's formando un subconjunto ç de $\overline{C}$ que sería generado y, por ende, recursivamente enumerable, y que por lo tanto podría ser trascendido encontrándose un n en $\overline{C}$ que no esté en ç.

4. REDUCIBILIDAD UNO-UNO A K; REDUCIBILIDAD MUCHOS-UNO

Sean S_1 y S_2 dos conjuntos de enteros positivos. Dispondríamos de una de las maneras más simples de reducir el problema de decisión de S_1 al problema de decisión de S_2, si tuviéramos un método efectivo que nos permitiera determinar para cada entero positivo

n un entero positivo m tal que n está, o no está, en S_1 según esté, o no esté, m en S_2. Pues si pudiéramos de algún modo determinar si m está o no está en S_2, entonces podríamos determinar si n está o no está en S_1. Si reemplazamos »método efectivo« por »método recursivo«, diremos, en breve, que S_1 es *reducible muchos-uno* a S_2. Si, más aún, n's distintos siempre llevan a m's distintos, diremos que S_1 es *reducible uno-uno* a S_2. »Método recursivo« aquí puede significar que $m = f(n)$ con $f(n)$ una función recursiva.

TEOREMA. *El problema de decisión de todo conjunto recursivamente enumerable de enteros positivos es reducible uno-uno al problema de decisión del conjunto completo K.*

Pues sea B' una base para un conjunto recursivamente enumerable S'. Entonces la correspondencia efectiva uno a uno entre todos los (B, n)'s y todos los enteros positivos dada por la enumeración efectiva O' de E, el conjunto de todos los (B, n)'s, da un entero positivo único m para cada (B', n), B' fijo, y entonces un m único para cada n, obteniéndose m's distintos para n's distintos. Ahora n está, o no está, en S' según (B', n) esté en T, o en $\overline{T}$, y por lo

tanto según m esté en K, o en $\overline{K}$, de lo cual se desprende nuestro resultado.

Como K mismo es recursivamente enumerable, podemos decir que para los conjuntos recursivamente enumerables de enteros positivos con problemas de decisión recursivamente insolubles, *hay un grado que es el más alto de insolubilidad con respecto a reducibilidad uno-uno,* a saber, el de K.

SUGERENCIAS DE LECTURA

Al lector interesado, sugiero las siguientes lecturas en orden creciente de dificultad:

1. Computation: finite and infinite machines. Marvin Minsky. Prentice Hall, Inc., 1967.
2. Recursively Enumerable Sets of Positive Integers and Their Decision Problems (del cual el lector ha leído cuatro secciones). Emil Post, Bulletin of the American Mathematical Society, 1944, vol. 50, pp. 284-316.
3. Undecidable Theories. Alfred Tarski. North Holland Publishing Company, 1971.
4. Computability and Unsolvability. Martin Davis, McGraw Hill Book Company, 1958.
5. Theory of Recursive Functions and Effective Computability. Hartley Rogers, McGraw Hill Book Company, 1967.