Eduardo Piza Volio

Una solución moderna para el "Entscheidungsproblem"

Abstract. *This paper analyses the famous decision problem of the first order canonical logic $\mathcal{F}_0$ (also called "Entscheidungsproblem") from a modern perspective. To explain a solution to this decision problem, the paper takes advantage of the development reached by recursion theory and semi-Thue productive systems, then of the work done by Post and Kleene in the 40's and by Davis in the 70's, among others. The "Entscheidungsproblem" is set out prior to using this set of resources.*

Key words: Entscheidungsproblem, first order logic, decision problem.

Resumen. *En este trabajo se analiza el famoso problema de decisión de la lógica canónica de primer orden $\mathcal{F}_0$ (también llamado "Entscheidungsproblem") desde una perspectiva moderna. Se aprovecha el desarrollo alcanzado por la teoría de la recursión y de los sistemas productivos semi-Thue, luego de los trabajos de Post y Kleene en los años 40's y de Davis en la década de los 70's, entre otros, para explicar una solución a este problema de decisión. Todo este instrumental empleado es posterior al planteamiento del "Entscheidungsproblem".*

Palabras clave: Entscheidungsproblem, lógica de primer orden, problema de decisión.

1. Introducción

En pocas palabras, el *problema de decisión* para una lógica de primer orden consiste en determinar si una fórmula bien formada dada es o no es un teorema. Se dice que un problema de decisión es *recursivamente resoluble* si existe un procedimiento efectivo (esto es, una lista de instrucciones o un algoritmo que no requiera de ninguna ingeniosidad para ser ejecutado) para resolver el problema de decisión.

Dentro de las instancias de lógicas de primer orden, quizás la más estudiada es la *lógica canónica de primer orden* $\mathcal{F}_0$, también llamada la "Engere Prädikatenkalkül" por Hilbert y Ackermann, mientras que Church la denominaba "el cálculo funcional puro de primer orden". La importancia de $\mathcal{F}_0$ proviene del resultado bien conocido[1] desde principios del siglo XX que establece que si el problema de decisión para $\mathcal{F}_0$ es recursivamente resoluble entonces también lo es el de cualquier lógica de primer orden.

Ya Hilbert y su reconocida Escuela Formalista dominaban intuitivamente este importante resultado, aún antes de la aparición y desarrollo formal de la teoría de la recursión, a mediados del siglo XX. Acerca de este problema Hilbert llegó a declarar que el problema de decisión para $\mathcal{F}_0$ (usualmente referido como el "Entscheidungsproblem") era el problema central de la lógica matemática. Subsecuentemente se realizó mucha investigación dirigida a buscar una solución positiva al "Entscheidungsproblem".

Algunos resultados preliminares para subclases de $\mathcal{F}_0$ arrojaron resultados positivos, alimentando la creencia errónea que el "Entscheidungsproblem" era recursivamente resoluble y que el menor avance en cualquier dirección podría resolver el problema. Sin embargo, en 1936 Church y Turing demostraron

precisamente lo contrario, en forma independiente uno del otro: ¡el "Entscheidungsproblem" es recursivamente irresoluble! Church utilizó la técnica de "λ-cálculo", mientras que Turing introdujo sus célebres "máquinas de Turing" para resolver este problema.

Analizaremos el "Entscheidungsproblem" desde la perspectiva moderna de la teoría de la recursión y de los sistemas productivos semi-Thue, introducidos por Post en la década de los 40's y perfeccionados por Davis en los 70's.

2. Lógicas formales

Trabajaremos con objetos matemáticos abstractos tales como "palabras", "predicados de palabras" y funciones sobre conjuntos de palabras. Emplearemos la siguiente *aritmetización* de estos objetos matemáticos abstractos:

- Si $\Sigma = \{ a_0, a_1, \ldots \}$ es el alfabeto (finito o numerable) de símbolos, a cada símbolo a_i le asociamos el número Gödel $gn(a_i) = 2\,i + 1$.

- Si $u = a_{i0}\, a_{i1} \ldots a_{im}$ es una palabra sobre Σ (esto es, concatenación de símbolos de Σ) le asociamos el número Gödel $gn(u)$ definido por

$$gn(u) := \prod_{k=0}^{m} p_k^{gn(a_{ik})},$$

donde p_k es el k-ésimo número primo ($p_0 = 2$, $p_1 = 3$, $p_2 = 5 \ldots$, etc.).

Un conjunto $\mathcal{U}$ de palabras sobre Σ se dice que es *recursivo* si el conjunto asociado $\mathcal{U}^*$ de todos los números Gödel de palabras de $\mathcal{U}$ es recursivo. Una función f entre conjuntos de palabras, $f : \mathcal{U} \to \mathcal{V}$ se dice que es recursiva si su función asociada f^* es recursiva, donde $f^* : \mathcal{U}^* \to \mathcal{V}^*$ es tal que

$$f^*(gn(u)) = gn(f(u)),\ \forall\ u \in \mathcal{U}\ .$$

Emplearemos libremente la teoría de funciones parcialmente recursivas. Por $\mathcal{R}$ denotamos la clase de funciones parcialmente recursivas. Escribimos

$$\mathcal{R} = \{\ \varphi_i^{(n)} : i \in N - \{0\}\ \},$$

donde $\varphi_i^{(n)}$ denota la i-ésima función parcialmente recursiva n-aria. Empezamos definiendo lo que son los *sistemas lógico formales*, y dentro de ellos los conceptos primitivos siguientes: *axiomas*, *reglas de inferencia*, *consecuencia*, *prueba*, *etapa de una prueba* y *teorema*.

Definición 1. *Por una lógica $\mathcal{L}$ (o sistema lógico formal) entenderemos*:

a. Un conjunto recursivo de palabras $\mathcal{U}$, llamado los axiomas de $\mathcal{L}$.

b. Un conjunto finito de predicados recursivos de palabras, ninguno de los cuales es 1-ario, llamados reglas de inferencia de $\mathcal{L}$.

Cuando $\mathcal{R}$ es una regla de inferencia (n+1)*-aria de $\mathcal{L}$, y $\mathcal{R}(Y, X_1, \ldots, X_n)$ es verdadero, entonces diremos que Y es una consecuencia de $X_1, \ldots, X_n$ en $\mathcal{L}$, por medio de la regla de inferencia $\mathcal{R}$.*

Definición 2. *Una sucesión finita de palabras $X_1, X_2, \ldots, X_m$ es llamada una prueba en la lógica $\mathcal{L}$ si, para cada $1 \le i \le m$, se cumple*:

a. O bien X_i es un axioma de $\mathcal{L}$,

b. O bien existen enteros $j_1, j_2, \ldots, j_k < i$ tales que X_i es una consecuencia lógica de $X_{j1}, X_{j2}, \ldots, X_{jk}$ en $\mathcal{L}$ por medio de alguna de las reglas de inferencia de $\mathcal{L}$.

Cada una de las palabras X_i es llamada una etapa de la prueba.

Definición 3. *Decimos que W es un teorema de $\mathcal{L}$, o que W es demostrable en $\mathcal{L}$, y escribimos*

$$\vdash_{\mathcal{L}} W$$

si existe una prueba en $\mathcal{L}$ cuya etapa final sea W. Esta prueba es entonces llamada una prueba de W en $\mathcal{L}$.

3. Problema de decisión de una lógica formal

Definición 4. *Dada una lógica formal $\mathcal{L}$, al conjunto de todos los números Gödel correspondientes a los teoremas de $\mathcal{L}$ lo denotamos por $T_{\mathcal{L}}$, esto es*:

$$T_{\mathcal{L}} = \{\ gn(W) : \vdash_{\mathcal{L}} W\ \}.$$

Es bien conocido que $T_{\mathcal{L}}$ es un conjunto *recursivamente enumerable*, esto es, existe un procedimiento efectivo para ir enumerando los teoremas de $\mathcal{L}$ uno detrás de otro. Este hecho no lo vamos a demostrar en este trabajo. El lector interesado en la demostración puede consultar por ejemplo Church (1956) o Piza.

No obstante, el hecho que $T_{\mathcal{L}}$ sea recursivamente enumerable no nos brinda un método eficaz para decidir si una fórmula bien formada arbitraria W es un teorema en $\mathcal{L}$ o no lo es. La dificultad surge al buscar W dentro de la lista de teoremas recursivamente enumerados de $\mathcal{L}$, pues en caso que W no aparezca en la lista de los primeros teoremas generados, no tendremos manera de establecer si W es o no un teorema de $\mathcal{L}$. Extenderemos la definición de problema de decisión para $\mathcal{L}$ de la siguiente manera:

Definición 5. *Por "el problema de decisión para la lógica $\mathcal{L}$" entenderemos el problema de determinar, para una palabra cualquiera dada, si es o no un teorema de $\mathcal{L}$. El problema de decisión para una lógica $\mathcal{L}$ es recursivamente resoluble si $T_{\mathcal{L}}$ es recursivo. De otra forma, diremos que el problema de decisión de $\mathcal{L}$ es recursivamente irresoluble.*

En la década de los años 40 Emil Post introdujo los llamados *sistemas combinatorios* o *sistemas semi-Thue* y demostró que, dado cualquier conjunto recursivamente enumerable A, es posible definir una lógica $\mathcal{L}$ cuyo "conjunto generado" $S_{\mathcal{L}}$ coincida con A. Como consecuencia de lo anterior, podemos enunciar el siguiente resultado.

Teorema 6. *Existe una lógica $\mathcal{L}$ cuyo problema de decisión es recursivamente irresoluble.*

Demostración: Los detalles de la construcción de esta lógica los veremos en el apéndice. Aquí solamente se comenta que la lógica $\mathcal{L}$ obtenida corresponderá a un sistema semi-Thue asociado con el conjunto K de Gödel,

$$K := \{\ x : \varphi_x^{(1)}(x) \downarrow\},$$

el cual sabemos que es recursivamente enumerable pero no es recursivo. ■

4. Lógicas de primer orden

El alfabeto de una *lógica de primer orden* $\mathcal{F}$ contiene a los símbolos

$$- \supset [\ ]\ ,\ (\ )$$

También en el alfabeto habrá una infinidad numerable de símbolos llamados *variables individuales*. Estas las escribiremos como

$$x,\ y,\ z,\ x_1,\ y_1,\ z_1,\ x_2,\ y_2,\ z_2,\ \ldots$$

Los símbolos restantes son las *constantes individuales*, los *símbolos de predicados* y los *símbolos de funciones*. Cada símbolo de predicado o de función está asociado con un entero mayor o igual a 1, llamado el *grado* del símbolo en cuestión, que corresponde al número de argumentos del mismo. Por *símbolo individual* entenderemos o bien una variable individual o bien una variable constante.

Por lo general nuestro desarrollo será elaborado en términos de un alfabeto fijo y numerable

$$a_0,\ a_1,\ a_2,\ \ldots$$

Supondremos además que los conjuntos correspondientes a las variables individuales, las constantes individuales, los símbolos de predicados y los símbolos de funciones son todos recursivos. Supondremos además que existe una función recursiva $Q(n)$ tal que, si a_n es un símbolo de predicado o de función, entonces $Q(n)$ es el grado de a_n.

Decimos que una palabra X es un *término* de una lógica de primer orden $\mathcal{F}$ si existe una secuencia de palabras $X_1, \ldots, X_n$ tal que X_n es X y, para cada $i \in \{1, \ldots, n\}$ se cumple una de las siguientes condiciones:

1. X_i es un símbolo individual, o bien

2. X_i es $a(\ X_{j1}, X_{j2}, \ldots, X_{jm}\ )$, donde $j_1, j_2, \ldots, j_m < i$ y a es un símbolo de función de grado m.

Decimos que una palabra W es una *fórmula bien formada* (que abreviaremos por f.b.f.) de una lógica de primer orden $\mathcal{F}$, si existe una secuencia de palabras $W_1, \ldots, W_n$ tal que W_n es W y, para cada $i \leq n$ se cumple una de las siguientes condiciones:

1. W_i es $p(X_1, \ldots, X_m)$, donde $X_1, \ldots, X_m$ son términos y p es un símbolo de predicado de grado m, o bien

2. W_i es $[W_j \supset W_k]$, donde $j, k < i$, o bien

3. W_i es $-W_j$, donde $j < i$, o bien

4. W_i es $(v)W_j$, donde $j < i$ y v es una variable individual.

En el caso de un símbolo de función o de predicado de grado 2, con frecuencia escribimos (XaY) en vez de $a(X,Y)$. Una lógica de primer orden tiene dos *reglas de inferencia*, a saber:

- *Modus ponens:* Esta regla de inferencia está gobernada por el predicado de 3 variables $\mathcal{R}(Y, X_1, X_2)$, el cual es verdadero si y solo si X_2 es $[X_1 \supset Y]$.[2]

- *Generalización:* Esta regla de inferencia está gobernada por el predicado $\Xi(Y,X)$, el cual es verdadero si y solo si Y es $(v)X$, donde v es una variable individual.[3]

No es difícil verificar que estas reglas de inferencia son predicados recursivos de palabras.

Decimos que una ocurrencia de una variable individual v en una f.b.f. W es una *ocurrencia acotada* si se trata de una parte bien formada de W de la forma $(v)A$. Una ocurrencia de v en una f.b.f. W que no es acotada es llamada *ocurrencia libre*. Una f.b.f. en la cual ninguna variable individual tenga alguna ocurrencia libre se dice que es *cerrada*.

Escribimos f.b.f.c. para las f.b.f. cerradas. Si W es una f.b.f., v es una variable individual y X es un término, entonces definimos el predicado $S(W,v,X)$ como sigue: si ninguna ocurrencia libre de v en W se encuentra en una parte bien formada de W de la forma $(w)A$, donde w ocurre en X, entonces $S(W,v,X)$ es el resultado de reemplazar v por X en todas las ocurrencias de v en W; de otra manera, $S(W,v,X)$ es W.

Los axiomas de una lógica de primer orden $\mathcal{F}$ incluyen a todas las palabras obtenidas al reemplazar v por una variable individual, X por un término, y A, B y C por fórmulas bien formadas en los siguientes cinco esquemas:

1. $[A \supset [B \supset A]]$.

2. $[[A \supset [B \supset C]] \supset [[A \supset B] \supset [A \supset C]]]$.

3. $[[-B \supset -A] \supset [A \supset B]]$.

4. $[(v)A \supset S(A,v,X)]$.

5. $[(v)[A \supset B] \supset [A \supset (v)B]]$, cuando v no tiene ocurrencias libres en A.

Analizamos a continuación nuestro primer resultado elemental, válido en cualquier lógica de primer orden.

Teorema 7. *Sea $\mathcal{F}$ una lógica de primer orden y sea A una fórmula bien formada de $\mathcal{F}$. Entonces tendremos:* $\vdash_{\mathcal{F}} [A \supset A]$.

Demostración: Al reemplazar B por $[A \supset A]$ y C por A en el esquema (2), obtenemos

$\vdash_{\mathcal{F}} [[A \supset [[A \supset A] \supset A]] \supset [[A \supset [A \supset A]] \supset [A \supset A]]]$.

Al aplicar el esquema (1) dos veces aparecen los siguientes dos teoremas:

$\vdash_{\mathcal{F}} [A \supset [[A \supset A] \supset A]]$, $\vdash_{\mathcal{F}} [A \supset [A \supset A]]$.

La tesis del teorema se obtiene entonces de aplicar la regla del *modus ponens* dos veces. ■

5. Lógicas especializadas de primer orden

Adicionalmente a los 5 axiomas básicos, una lógica de primer orden $\mathcal{F}$ contiene un número finito (posiblemente ninguno) de axiomas adicionales, todos los cuales son fórmulas bien formadas cerradas; estos son llamados los *axiomas especializados* de la lógica.

Definición 8. *Sea $\mathcal{F}$ una lógica de primer orden y sean $A_1, A_2, \ldots, A_n$ fórmulas bien formadas cerradas de $\mathcal{F}$. Entonces, $\mathcal{F}(A_1, \ldots, A_n)$ es la lógica de primer orden obtenida a partir de $\mathcal{F}$ al agregarle como axiomas especializados aquellas fórmulas $A_1, \ldots, A_n$ que no sean axiomas de $\mathcal{F}$.*

A continuación se establece la relación fundamental entre los teoremas de una lógica

especializada y sus correspondientes versiones en la lógica que le dio origen.

Teorema 9. *Sea $\mathcal{F}$ una lógica de primer orden y sea A una fórmula bien formada cerrada de $\mathcal{F}$.* Luego tendremos: $\vdash_{\mathcal{F}(A)} W$ *si y solo si* $\vdash_{\mathcal{F}} [A \supset W]$.

Demostración: Supongamos primero que $\vdash_{\mathcal{F}}[A \supset W]$. Entonces, tendremos $\vdash_{\mathcal{F}(A)} [A \supset W]$. Luego, aplicando la regla del *modus ponens* obtenemos $\vdash_{\mathcal{F}(A)} W$.

Recíprocamente, supongamos ahora que $\vdash_{\mathcal{F}(A)} W$. Sea $W_1, \ldots, W_n$ una prueba de W en $\mathcal{F}$ (A), de manera que W_n es W. Mostraremos que, para cada índice $i \in \{1,\ldots, n\}$, tendremos $\vdash_{\mathcal{F}(A)} [A \supset W_i]$. Para ello supongamos como hipótesis inductiva que este resultado se sabe cierto para todo $j < i$. Distinguiremos varios casos:

Caso I: Cuando W_i es A. Entonces, $[A \supset W_i]$ es $[A \supset A]$, que en virtud del teorema 7 se trata de un teorema de $\mathcal{F}$.

Caso II: Cuando W_i es un axioma de $\mathcal{F}$. Entonces, $\vdash_{\mathcal{F}} W_i$. Empleando el esquema (1) obtenemos $\vdash_{\mathcal{F}} [W_i \supset [A \supset W_i]]$. Por la regla del *modus ponens*, tendremos $\vdash_{\mathcal{F}} [A \supset W_i]$.

Caso III: Cuando W_j es $[W_k \supset W_i]$, para j, $k < i$. Luego, de la hipótesis inductiva obtenemos los dos teoremas: *i)* $\vdash_{\mathcal{F}} [A \supset W_k]$; ii) $\vdash_{\mathcal{F}} [A \supset [W_k \supset W_i]]$. Del esquema (2), además tendremos

$\vdash_{\mathcal{F}} [[A \supset [W_k \supset W_i]] \supset [[A \supset W_k] \supset [A \supset W_i]]]$.

Entonces, aplicando la regla del *modus ponens* dos veces, obtenemos finalmente $\vdash_{\mathcal{F}} [A \supset W_i]$.

Caso IV: Cuando W_i es $(v)W_j$, para $j < i$, donde v es una variable individual. Aplicando la regla de generalización, tendremos $\vdash_{\mathcal{F}} (v)[A \supset W_j]$. Como A es una f.b.f.c., el esquema (5) es aplicable, obteniéndose

$$\vdash_{\mathcal{F}} [(v)[A \supset W_j] \supset [A \supset (v)W_j]].$$

Aplicando la regla del *modus ponens*, obtenemos finalmente $\vdash_{\mathcal{F}} [A \supset (v)W_j]]$, esto es, $\vdash_{\mathcal{F}} [A \supset W_i]$. Esto completa la demostración. ■

La generalización al caso de lógicas formales de primer orden con n axiomas especializados es elemental y se presenta en el siguiente corolario.

Corolario 10. *Sea $\mathcal{F}$ una lógica de primer orden y sean $A_1, A_2, \ldots, A_n$ fórmulas bien formadas cerradas de $\mathcal{F}$. Luego,* $\vdash_{\mathcal{F}(A1,\ldots,An)} W$ *si y solo si* $\vdash_{\mathcal{F}} [A_1 \supset [A_2 \supset \ldots [A_n \supset W] \ldots]]$.

Demostración: Se aplica el teorema precedente n veces. ■

6. Traslabilidad entre sistemas formales

Uno de los mecanismos más útiles para poder comparar sistemas lógicos formales es el concepto de la *traslabilidad*, el cual se introduce a continuación.

Definición 11. *Sean $\mathcal{F}$ y $\mathcal{F}'$ dos lógicas. Decimos que $\mathcal{F}$ es trasladable hacia $\mathcal{F}'$ si existe una función de palabras f, recursiva y 1-1 tal que, para toda palabra X*:

$$\vdash_{\mathcal{F}} X \Leftrightarrow \vdash_{\mathcal{F}'} f(X).$$

La importancia de este concepto se pone de manifiesto en los siguientes resultados.

Teorema 12. *Sea $\mathcal{F}$ una lógica trasladable hacia $\mathcal{F}'$ y suponga que el problema de decisión para $\mathcal{F}'$ es recursivamente resoluble. Entonces, el problema de decisión para $\mathcal{F}$ también es recursivamente resoluble.*

Demostración: De las hipótesis, tenemos que $T_{\mathcal{F}'}$ es recursivo. Sea f la función recursiva y 1-1 de palabras tal que

$$\vdash_{\mathcal{F}} X \Leftrightarrow \vdash_{\mathcal{F}'} f(X).$$

Entonces,

$$T_{\mathcal{F}} = \{ x : f^*(x) \in T_{\mathcal{F}'} \}$$

Llamando por c y c' las funciones características de T_F y $T_{F'}$ respectivamente, tendremos entonces que $c = c'o f^*$. Como por hipótesis tanto c' como f^* son recursivas, lo anterior pone en evidencia que c es recursiva. Luego el conjunto T_F es recursivo y por tanto el problema de decisión de F es recursivamente resoluble. ■

Teorema 13. *Supóngase que la lógica $\mathcal{F}$ es trasladable hacia la lógica $\mathcal{F}$ ' y además $\mathcal{F}$ ' es trasladable hacia la lógica $\mathcal{F}$ ". Entonces, $\mathcal{F}$ es trasladable hacia $\mathcal{F}$ ".*

Demostración: Existen funciones recursivas y 1-1 de palabras f y g tales que

$$\vdash_{\mathcal{F}} X \Leftrightarrow \vdash_{\mathcal{F}'} f(X),$$

$$\vdash_{\mathcal{F}'} X \Leftrightarrow \vdash_{\mathcal{F}''} g(X),$$

Sea $h = g$ o f. Luego h es recursiva y 1-1. Además, para cada palabra X se cumple:

$$\vdash_{\mathcal{F}} X \Leftrightarrow \vdash_{\mathcal{F}'} f(X),$$

$$\Leftrightarrow \vdash_{\mathcal{F}''} h(X). \blacksquare$$

Teorema 14. *Sea $\mathcal{F}$ una lógica de primer orden y sean $A_1, \ldots, A_n$ fórmulas bien formadas cerradas de $\mathcal{F}$. Luego, $\mathcal{F}(A_1, \ldots, A_n)$ es trasladable hacia $\mathcal{F}$.*

Demostración: En virtud del corolario 10, solamente es necesario demostrar que la función de palabras f dada por

$$f(W) = [A_1 \supset [A_2 \supset \ldots \supset [\, A_n \supset W] \ldots]]$$

es recursiva, asunto cuya demostración es rutinaria. ■

Definición 15. *Decimos que una lógica de primer orden $\mathcal{F}$ es no-especializada si $\mathcal{F}$ no tiene axiomas especializados.*

Teorema 16. *Toda lógica de primer orden es trasladable hacia una lógica de primer orden no-especializada.*

Demostración: Sea $\mathcal{F}$ una lógica de primer orden y sea $\mathcal{F}'$ la lógica de primer orden no-especializada obtenida de $\mathcal{F}$ al suprimirle los axiomas especializados. Más aún, supongamos que los axiomas especializados de $\mathcal{F}$ son $A_1, \ldots, A_n$. Entonces, $\mathcal{F} = \mathcal{F}'(A_1, \ldots, A_n)$. Luego, del teorema 14, tendremos que $\mathcal{F}$ es trasladable hacia $\mathcal{F}'$. ■

7. La "Engere Prädikatenkalkül" $\mathcal{F}_0$

Introducimos ahora la lógica no-especializada de primer orden $\mathcal{F}_0$, denominada *lógica de primer orden canónica*. $\mathcal{F}_0$ tiene una infinidad numerable de constantes individuales y, para cada entero $m \geq 1$, una infinidad numerable de símbolos de funciones y predicados de grado m. Concretamente, los siete símbolos

$$- \qquad \supset \qquad [\,] \qquad , \qquad (\,)$$

serán identificados mediante $a_0, a_1, a_2, a_3, a_4, a_5, a_6$, respectivamente. Además, para $n > 6$, a_n denotará una *variable individual* si $\sigma_2(n) = 0$ y $\sigma_1(n)$ es par[4], mientras que a_n denotará una *constante individual* si $\sigma_2(n) = 0$ y $\sigma_1(n)$ es impar. Luego, las variables individuales

$$x, y, z, x_1, y_1, z_1, x_2, y_2, z_2, \ldots$$

son identificadas por $a_{\pi(4,0)}$, $a_{\pi(6,0)}$, $a_{\pi(8,0)}$, …, etc. Finalmente, para $n > 6$, con $\sigma_2(n) > 0$, tendremos que cuando $\sigma_2(n)$ es par entonces a_n denotará el *símbolo de predicado* de grado $\frac{1}{2}\,\sigma_2(n)$, mientras que cuando $\sigma_2(n)$ es impar, entonces a_n denotará el *símbolo de función* de grado $\frac{1}{2}(\sigma_2(n)+1)$.

$\mathcal{F}_0$ esencialmente es la lógica denominada "la *Engere Prädikatenkalkül*" por Hilbert y Ackermann, mientras que Church la denominaba "el *cálculo funcional puro de primer orden*. La importancia de $\mathcal{F}_0$ proviene del siguiente hecho.

Teorema 17. *Toda lógica de primer orden no-especializada es trasladable hacia $\mathcal{F}_0$.*

Demostración: Sea $\mathcal{F}$ una lógica de primer orden no-especializada. Definimos la función recursiva φ como sigue. Para cada n, buscamos el papel que juega el símbolo a_n dentro de $\mathcal{F}$:

- Si a_n es uno de los siete símbolos "–", "$\supset$", "[", "]", ",", "(", ")" de $\mathcal{F}$, entonces $\varphi(n)$ tendrá el valor 0, 1, 2, 3, 4, 5 o 6, respectivamente.

- Si $\varphi(x)$ está definida para $x < n$ y a_n es una *variable individual* en $\mathcal{F}$, o una *constante individual* en $\mathcal{F}$, o un *símbolo de predicado* de grado m en $\mathcal{F}$, o un *símbolo de función* de grado m en $\mathcal{F}$, entonces definimos $\varphi(n)$ como el menor entero p tal que $\varphi(x) \neq p$, para $x < n$ y a_p es una *variable individual* en $\mathcal{F}_0$, o una *constante individual* en $\mathcal{F}_0$, o un símbolo de predicado de grado m en $\mathcal{F}_0$, o un *símbolo de función* de grado m en $\mathcal{F}_0$, respectivamente.

Ahora, definimos la función recursiva de palabras $f(X)$ mediante

$$f(a_{r1}a_{r2}\cdots a_{rk}) = a_{\varphi(r1)}a_{\varphi(r2)}\cdots a_{\varphi(rk)}.$$

Es claro que f es 1-1 y además satisface la condición de traslabilidad:

$$\vdash_{\mathcal{F}} X \Leftrightarrow \vdash_{\mathcal{F}_0} f(X). \blacksquare$$

Teorema 18. *Si el problema de decisión para $\mathcal{F}_0$ es recursivamente resoluble, entonces también lo es el de cualquier lógica de primer orden.*

Demostración: Inmediata, de los teoremas 12, 13, 16 y 17. ■

El teorema precedente era ya bien conocido (necesariamente a nivel informal) por Hilbert y su Escuela Formalista aún antes del desarrollo formal de la teoría de las funciones recursivas. Sobre la base de este teorema, Hilbert declaró que el problema de decisión para $\mathcal{F}_0$ (el "Entscheidungsproblem") era el problema central de la lógica matemática. Subsecuentemente se realizó mucha investigación dirigida a buscar una solución positiva al "Entscheidungsproblem". En parte, estos esfuerzos tomaron la forma de buscar reducciones del problema de decisión para $\mathcal{F}_0$ al correspondiente problema de decisión para clases especiales de fórmulas bien formadas, como por ejemplo, la clase de las fórmulas bien formadas del tipo[5]

$$(\exists x_1)\,(\exists x_2)\,(\exists x_3)\,(y_1)\,(y_2)\,\ldots\,(y_n)\,A,$$

donde A es un término libre de cuantificadores. Otros trabajos consistieron directamente en demostrar que, para clases de fórmulas bien formadas cada vez mayores, el problema de decisión podía ser resuelto. En particular, se llegó a demostrar que el problema de decisión puede ser resuelto para la clase de todas las fórmulas bien formadas del tipo

$$(\exists x_1)\,(\exists x_2)\,(y_1)\,(y_2)\,\ldots\,(y_n)\,A,$$

donde A es un término libre de cuantificadores. Estos resultados preliminares alimentaron por un tiempo la conjetura errónea que el "Entscheidungsproblem" era recursivamente resoluble. Parecía además que el menor avance en cualquier dirección podría resolver el problema.

No obstante, en 1936 Church y Turing demostraron al mismo tiempo (aunque de manera independiente y con métodos muy diferentes el uno del otro) que el problema de decisión para $\mathcal{F}_0$ era recursivamente irresoluble. El artículo de Church se titulaba "*A Note on the Entscheidungsproblem*" y empleó su instrumental propio de *lambda cálculo*. Por su parte, el artículo de Turing se titulaba "*On Computable Numbers, with an Application to the Entscheidungsproblem*". Fue aquí donde Turing introdujo por vez primera el concepto de las famosas máquinas que llevan su nombre. Por aquel entonces, Turing era apenas un muchacho que acababa de terminar sus estudios de matemática a nivel de pregrado. Church quedó tan bien impresionado con la metodología empleada por Turing, que lo invitó a realizar estudios doctorales en Princeton, bajo su tutela, cosa que Turing aceptó gustoso.

8. Solución al "Entscheidungsproblem"

Teorema 19. *Todo sistema semi-Thue* σ *es trasladable hacia una lógica de primer orden* $\mathcal{F}_\sigma$.

Demostración: Aplazaremos para el final de esta sección la demostración de este resultado, con el fin de no perder el hilo de los siguientes acontecimientos. ■

Corolario 20. *Existe una lógica de primer orden cuyo problema de decisión es recursivamente irresoluble.*

Demostración: Supóngase lo contrario. Luego, toda lógica de primer orden posee problema de decisión recursivamente resoluble. En particular las lógicas de primer orden del tipo $\mathcal{F}_\sigma$. engendradas a partir de sistemas semi-Thue tendrán problemas de decisión recursivamente resolubles. Al aplicar el teorema 12 junto con el teorema 19 precedente, obtenemos entonces que todo sistema semi-Thue σ tiene problema de decisión recursivamente resoluble. Pero esto es precisamente lo contrario de lo que establece el teorema 29 del apéndice. Llegamos entonces a un absurdo. ■

Teorema 21. (Church, Turing, 1936) *El problema de decisión para* $\mathcal{F}_0$ *es recursivamente irresoluble.*

Demostración: En efecto, si el problema de decisión para $\mathcal{F}_0$ fuera recursivamente resoluble, entonces, por el teorema 18, también sería recursivamente resoluble el problema de decisión de cualquier lógica de primer orden. Pero precisamente el corolario precedente establece que existe una lógica de primer orden cuyo problema de decisión es recursivamente irresoluble. Luego, se concluye el aserto del teorema. ■

En lo que resta de esta sección nos dedicaremos a demostrar formalmente el teorema 19.

Demostración del teorema 19: Sea $\sigma = (P, A_0)$ un sistema semi-Thue sobre el alfabeto finito Σ, con reglas de producción $P = \{ \alpha_i \rightarrow \beta_i : 1 \leq i \leq m \}$ y axioma A_0. Vamos a construir la lógica de primer orden $\mathcal{F}_\sigma$ asociada a σ de la siguiente manera:

- *Constantes individuales de* $\mathcal{F}_\sigma$: las letras del alfabeto Σ.

- *Símbolo de función de* $\mathcal{F}_\sigma$: * , de grado 2 (el cual imitará en $\mathcal{F}_\sigma$ la operación de la concatenación en Σ).

- *Símbolos de predicados de* $\mathcal{F}_\sigma$: *T*, de grado 1 y "=", de grado 2.

A cada palabra *W* sobre el alfabeto Σ le asociamos el *término W'* de $\mathcal{F}_\sigma$ en forma recursiva de acuerdo al siguiente patrón:

$$a' = a, \forall\, a \in \Sigma,$$

$$(Wa)' = (W' * a), \forall\, a \in \Sigma.$$

Definimos la función de palabras *f* mediante $f(W) = T(W')$. Claramente *f* es recursiva. Los axiomas especializados de $\mathcal{F}_\sigma$ serán escogidos de tal manera que $\vdash_\sigma W$ sí y solo si $\vdash_{\mathcal{F}\sigma} f(W)$.

- *Axiomas especializados de* $\mathcal{F}_\sigma$:

1. $(x_1)\ (x_1 \equiv x_1)$.

2. $(x_1)(x_2)\ [(x_1 \equiv x_2) \supset (x_2 \equiv x_1)]$.

3. $(x_1)(x_2)(x_3)\ [(x_1 \equiv x_2) \supset [(x_2 \equiv x_3) \supset (x_1 \equiv x_3)]]$.

4. $(x_1)(x_2)\ [(x_1 \equiv x_2) \supset [T(x_1) \equiv T(x_2)\ ]]$.

5. $(x_1)(x_2)(x_3)\ [(x_1 \equiv x_2) \supset ((x_1 * x_3) \equiv (x_2 * x_3))]$.

6. $(x_1)(x_2)(x_3)\ [(x_1 \equiv x_2) \supset ((x_3 * x_1) \equiv (x_3 * x_2))]$.

7. $(x_1)(x_2)(x_3)\ ((x_1 * (x_2 * x_3)) \equiv ((x_1 * x_2) * x_3))$.

8. $T(A_0')$.

9. $(x_1)(x_2)\ [T(((x_1 * \alpha_i')\ * x_2)) \supset T(((x_1 * \beta_i')\ * x_2))]$, para cada $i \in \{1, \ldots, m\}$.

Vamos a demostrar que $\vdash_\sigma W$ si y solo si $\vdash_{\mathcal{F}\sigma} T(W')$. Un término de $\mathcal{F}_\sigma$ se dice ser una *constante* si no contiene variables individuales. Si *X* es una constante, escribimos $\{X\}$ para denotar la palabra obtenida de *X* al remover todos los paréntesis y las estrellas *. Por ejemplo, $\{W'\} = W$. Decimos que dos constantes *X*, *Y* son *asociadas* si $\{X\} = \{Y\}$. Luego, tendremos los siguientes lemas.

Lema 22. *Si X y Y son constantes, entonces* si $\vdash_{\mathcal{F}\sigma} (X = Y)$ *sí y solo si X y Y son asociadas.*

Demostración: Si *X* y *Y* son asociadas, entonces $\{X\} = \{Y\}$, de manera que $\{X\}$ y $\{Y\}$ tienen el mismo largo. Si este largo es 1 o 2, el lema es obvio. Si el largo es 3, luego $\{X\} = \{Y\} = acb$, con $a, b, c \in \Sigma$. Entonces, las únicas posibilidades para *X*, *Y* son $(a * (b * c))$, $((a * b) * c)$. Ahora, aplicando el axioma especializado (7), tendremos

$$\vdash_{\mathcal{F}\sigma} (x_1)(x_2)(x_3)\ ((x_1 * (x_2 * x_3)) \equiv ((x_1 * x_2) * x_3)).$$

Por otra parte, en virtud del esquema de axiomas (4) para lógicas de primer orden, tendremos además los siguientes tres teoremas de $\mathcal{F}_\sigma$:

$$\vdash_{\mathcal{F}\sigma} [(x_1)(x_2)(x_3)\ ((x_1 * (x_2 * x_3)) \equiv ((x_1 * x_2) * x_3)) \supset (x_2)(x_3)\ ((a * (x_2 * x_3)) \equiv ((a * x_2) * x_3))],$$

$$\vdash_{\mathcal{F}\sigma} [(x_2)(x_3)\ ((a * (x_2 * x_3)) \equiv ((a * x_2) * x_3)) \supset (x_3)\ ((a * (b * x_3)) \equiv ((a * b) * x_3))],$$

$$\vdash_{\mathcal{F}\sigma} [(x_3)\ ((a * (b * x_3)) \equiv ((a * b) * x_3)) \supset ((a * (b * c)) \equiv ((a * b) * c))].$$

Aplicando la regla del *modus ponens* tres veces, obtenemos

$$\vdash_{\mathcal{F}\sigma} ((a * (b * c)) = ((a * b) * c)),$$

esto es,

$$\vdash_{\mathcal{F}\sigma} (X = Y).$$

El resultado para largos mayores que 3 se demuestra usando inducción matemática, de manera similar. Finalmente, el recíproco se obtiene de la observación que los axiomas especializados (1) y (7) son los únicos que producen abiertamente equivalencias "=", las

cuales ciertamente no pueden ser producidas por constantes no-asociadas. Por otra parte, los axiomas del (2) al (6) producen equivalencias entre partes asociadas a partir de otras equivalencias entre partes asociadas. ■

Lema 23. *Si* $\vdash_{\sigma} W$ *entonces* $\vdash_{\mathcal{F}\sigma} T(W')$.

Demostración: Sea $W_1, W_2, \ldots, W_n$ una prueba en Σ, donde W_n es W. Mostraremos que, para cada $i \in \{1, \ldots, n\}$, tendremos $\vdash_{\mathcal{F}\sigma} T(W_i')$. Esto es claramente cierto para $i = 1$, pues W_1 es A_0. Supongamos que es cierto para $i = k$, con $k < n$. Luego, para algún índice j, tendremos $W_k = P\ \alpha_j\ Q$, $W_{k+1} = P\ \beta_j\ Q$. Del esquema de axiomas (4) de las lógicas de primer orden, tendremos

$$\vdash_{\mathcal{F}\sigma} [(x_1)(x_2)\ [T\ (((x_1 * \alpha_j') * x_2)) \supset T\ (((x_1 * \beta_j') * x_2))] \supset [T\ (((P' * \alpha_j') * Q')) \supset T\ (((P' * \beta_j') * Q'))]].$$

Esto, junto con el axioma (9) y la regla del *modus ponens*, nos brinda

$$\vdash_{\mathcal{F}\sigma} [T\ (((P' * \alpha_j') * Q')) \supset T\ (((P' * \beta_j') * Q'))].$$

Por otra parte, del lema 22, tenemos

$$\vdash_{\mathcal{F}\sigma} ((P\ \alpha_j\ Q)' \equiv ((P' * \alpha_j') * Q')).$$

Utilizando el axioma especializado (4), obtenemos entonces

$$\vdash_{\mathcal{F}\sigma} [T\ ((P\ \alpha_j\ Q)') \supset T\ (((P' * \alpha_j') * Q'))].$$

Ahora, aplicando nuestra hipótesis de inducción y la regla del *modus ponens* dos veces, obtenemos

$$\vdash_{\mathcal{F}\sigma} T\ (((P' * \beta_j') * Q')).$$

Finalmente, aplicando el lema 22, el axioma especializado (4), el esquema de axiomas (4) de las lógicas de primer orden y la regla de *modus ponens*, obtenemos

$$\vdash_{\mathcal{F}\sigma} T\ ((P\ \beta_j\ Q)').$$

esto es, $\vdash_{\mathcal{F}\sigma} T\ (W'_{k+1})$ ■

Lema 24. *Si X es una constante y* $\vdash_{\sigma} \{X\}$, *entonces* $\vdash_{\mathcal{F}\sigma} T\ (X)$.

Demostración: Si $X = \{X\}'$, el resultado es consecuencia inmediata del lema 23. En caso contrario, X es un asociado de $Y = \{X\}'$. Luego, de los lemas 22, 23 y la hipótesis, tendremos los dos teoremas

$$\vdash_{\mathcal{F}\sigma} (X = Y)\ ,\ \vdash_{\mathcal{F}\sigma} T(Y).$$

Aplicando el axioma especializado (2), obtenemos

$$\vdash_{\mathcal{F}\sigma} (\ Y \equiv X).$$

Aplicando el axioma especializado (4), obtenemos

$$\vdash_{\mathcal{F}\sigma} [T\ (Y) \supset T\ (X)].$$

Finalmente, al aplicar la regla del *modus ponens* a esta última expresión, obtenemos el resultado. ■

Lema 25. *Si X es una constante y* $\vdash_{\mathcal{F}\sigma} T(X)$, *entonces* $\vdash_{\sigma} \{X\}$.

Demostración: El único axioma de $\mathcal{F}_{\sigma}$ de la forma $T(X)$, donde X es una constante, tiene la propiedad deseada. Por otra parte, para derivar $T(X)$ en $\mathcal{F}_{\sigma}$, con X constante, deberán ser empleados los axiomas especializados (4) o (9). Pero, en virtud del lema 22, el axioma (4) puede producir a $T(X)$ como teorema solamente si ya conocemos que $\vdash_{\mathcal{F}\sigma} T(Y)$, donde Y y X son asociados. Por otra parte, el axioma (9) puede producir a $T(X)$ como teorema solamente si $\{X\}$ es una consecuencia de $\{Y\}$ al aplicar una de las producciones de σ y si $\vdash_{\mathcal{F}\sigma} T(Y)$. ■

Lema 26. $\vdash_{\sigma} W$ *si y solo si* $\vdash_{\mathcal{F}\sigma} T(W')$.

Demostración: Es clara, de los lemas 22 y 23. Esto completa además la prueba del teorema 19. ■

Apéndice: Sistemas semi-Thue

El término *semi-Thue* fue introducido por Emil Post a principios de la década de los años 40, cuando introdujo ciertos sistemas combinatorios que los bautizó con este nombre en honor al matemático noruego Thue, quien introdujo algunas de las ideas preliminares al principio del siglo XX.

Definición 27. *Un sistema formal semi-Thue es un par ordenado* (T, Γ), donde:

a. T es un conjunto finito de pares ordenados (α, β), *con* α y β *hileras de símbolos de un alfabeto finito* Σ, *llamado el alfabeto del sistema. Los pares ordenados* $(\alpha, \beta) \in T$ *son llamados las producciones del sistema y*

alternativamente se acostumbra escribirlos en la forma "$\alpha \rightarrow \beta$".

b. Γ *es una hilera no nula y fija sobre* Σ. *A* Γ *se le llama el axioma del sistema.*

La interpretación de las producciones es como sigue: sean δ y γ hileras de Σ (posiblemente nulas). Cuando $(\alpha, \beta) \in T$, entonces escribimos

$$\delta\ \alpha\ \gamma \Rightarrow_T \delta\ \beta\ \gamma$$

y decimos que la hilera $\delta\ \beta\ \gamma$ es *directamente derivable* de la hilera $\delta\ \alpha\ \gamma$. Sean w_1 y w_2 dos hileras de Σ. Decimos que w_2 es *derivable* de w_1 y escribimos

$$w_1 \Rightarrow_T^* w_2$$

si $w_1 \Rightarrow_T w_2$, o bien existe una secuencia de hileras $z_1, z_2, \ldots, z_n$ sobre Σ tales que

$$w_1 \Rightarrow_T z_1 \Rightarrow_T z_2 \Rightarrow_T \ldots \Rightarrow_T z_n \Rightarrow_T w_2.$$

Por ejemplo, consideremos el sistema semi-Thue que resulta de tomar $\Sigma = \{a, b\}$ y T compuesto por las siguientes dos producciones: $ab \rightarrow bbb$, $bb \rightarrow \lambda$, donde λ denota la hilera nula (esto es, $T = \{ (ab, bbb), (bb, \lambda) \}$. Entonces, el lector puede comprobar fácilmente los siguientes hechos:

1. $abbab \Rightarrow_T^* bbbbbbb$,

2. $babb \Rightarrow_T^* b$,

3. Es falso que $abbabbb \Rightarrow_T^* bbbb$.

Definición 28. *Definimos el lenguaje generado por el sistema semi-Thue* (T, Γ) *el cual denotaremos por* $\mathcal{L}(T, \Gamma)$, *mediante*

$$\mathcal{L}(T, \Gamma) = \{ w \in \Sigma^* : \Gamma \Rightarrow_T^* w \}.$$

El lector puede observar que cada sistema semi-Thue (T, Γ) sobre un alfabeto Σ define a la vez un sistema lógico formal $\mathcal{L} = \mathcal{L}(T, \Gamma)$, en el cual Γ es el único axioma de $\mathcal{L}$ y las reglas de inferencia son precisamente las producciones de T. Bajo esta asociación, para cualquier hilera no nula w de Σ, son equivalentes las frases

$$w \in \mathcal{L}(T, \Gamma) \Leftrightarrow \vdash_{\mathcal{L}} w.$$

Consideremos el siguiente problema de decisión: dado un sistema semi-Thue (T, Γ) sobre Σ y dada w una hilera no nula de Σ, ¿será cierto que $w \in \mathcal{L}(T, \Gamma)$? Vamos a demostrar que este problema es recursivamente indecidible.

Teorema 29. *Existe un sistema semi-Thue* (T, Γ) *tal que el problema de decisión ¿* $w \in \mathcal{L}(T, \Gamma)$? es *recursivamente irresoluble*.

Demostración: Sea g una función cualquiera recursiva y sea M una máquina de Turing que evalúa a g en la forma estándar. Supongamos que los estados de M son $q_0, q_1, \ldots, q_r$ y que M se detiene (cuando lo hace) solamente en el estado q_r. Sean $s_0 = 0, s_1 = 1, s_2, \ldots, s_k$ los símbolos de M. Sea $m \in N$. Definimos el sistema semi-Thue (T_M, Γ_m) de la siguiente manera:[6]

a. Alfabeto: $\Sigma_M = \{ q0, \ldots, qr, s0, \ldots, sk, \# \}$.

b. Axioma Γ_m : $\#q00\,\bar{m}\#$, donde $\bar{m} = 11\ldots1$ (m+1 veces).[7]

c. Producciones de T_M :

- Por cada quíntuplo[8] de M de la forma (q, s, s', R, q'), tendremos las siguientes dos producciones:

$$(qs, s'q') \in T_M\ ,\ (q\#, s'q'\#) \in T_M.$$

- Por cada quíntuplo de M de la forma (q, s, s', L, q') y cada símbolo u, tendremos la producción:

$$(uqs, q'us') \in T_M.$$

- Por cada quíntuplo de M de la forma $(q, 0, x' \neq 0, L, q')$ y cada símbolo u, tendremos la producción:

$$(uq\#, q'us'\#) \in T_M.$$

Con esta construcción el lector puede fácilmente comprobar que

$\#qr0\bar{u}\# \in \mathcal{L}(T_M, \Gamma_m) \Leftrightarrow g(m) \downarrow$ y $g(m) = u$.

Tomemos ahora la función g dada por

$$g(m) = \begin{cases} 0, & \text{si } m \in K, \\ \uparrow & \text{si } m \notin K, \end{cases}$$

donde K es el conjunto de Gödel y consideremos la máquina M correspondiente, que evalúa a g con

los requisitos señalados. Obtenemos un sistema semi-Thue (T_M, Γ_m) tal que

$$\#qr01\# \in \mathcal{L}\,(T_M, \Gamma_m) \Leftrightarrow m \in K.$$

Tomemos ahora el inverso del sistema (T_M, Γ_m), que consiste en el sistema semi-Thue (T, Γ) dado por:

1. **Alfabeto:** el mismo.

2. **Axioma:** $\#qr01\#$.

3. **Producciones:** $(\alpha, \beta) \in T_M \Leftrightarrow (\beta, \alpha) \in T$.

Obtenemos finalmente:

$$\#q_0 0\overline{m}\# \in \mathcal{L}\,(T, \Gamma) \Leftrightarrow m \in K.$$

Por lo tanto, el problema de decisión para el sistema semi-Thue (T, Γ) es recursivamente indecidible. ■

De paso hemos construido los cimientos que sustentan el resultado que establece la existencia de sistemas lógicos $\mathcal{L}$ cuyo problema de decisión es recursivamente irresoluble (teorema 6).

Notas

1. Aunque de manera intuitiva, ya que la teoría de la recursión aún no estaba suficientemente desarrollada para aquel entonces, lo que obstaculizaba la formalización del concepto de "problema de decisión".
2. Intuitivamente, la regla del *modus ponens* establece que si tenemos el teorema $\vdash_{\mathcal{F}} [X_1 \supset Y]$ y además tenemos el teorema $\vdash_{\mathcal{F}} X_1$, entonces también tendremos el teorema $\vdash_{\mathcal{F}} Y$.
3. Intuitivamente la regla de *generalización* establece que si tenemos el teorema $\vdash_{\mathcal{F}} X$ y v es una variable individual, entonces también tendremos el teorema $\vdash_{\mathcal{F}} (v)X$.
4. Aquí π: NXN $\rightarrow$ N es la función codificadora de Cantor dada por $\pi(x,y) = x + (x+y)\,(x+y+1) / 2$, mientras que σ_1 y σ_2 son las funciones decodificadoras de Cantor, caracterizadas por la identidad $n = \pi(\sigma_1(n),\sigma_2(n))$, para todo n $\in$ N. Tanto π como σ_1 y σ_2 son funciones recursivas.
5. Aquí, por $(\exists x_i)\ W$ entenderemos $-(x_i)\ -W$.
6. El sistema semi-Thue (T_M, Γ_m) que estamos construyendo se encargará de imitar "en papel y lápiz" el comportamiento de la máquina de Turing M. El símbolo adicional "#" denotará el principio o final de la cinta de M, que en este enfoque es potencialmente infinita en ambas direcciones.
7. La interpretación de este axioma es simplemente poner a la máquina de Turing M a calcular, empezando en el estado q_0 y frente al entero m.
8. El esquema empleado aquí para denotar los quíntuplos de las máquinas de Turing es el siguiente: los quíntuplos son de la forma (q, s, s', D, q'), donde q es el estado actual de M, s es el símbolo leído de la cinta, s' es el símbolo que escribe M en la cinta en reemplazo de s, D es la dirección de la cinta hacia la cual se mueve un cuadro la cabeza de M ("R" de "right" o "L" de "left"), y finalmente q' es el nuevo estado en que entra M luego de realizar esta acción.

Bibliografía

Church, Alonso. *Introduction to Mathematical Logic*, Vol. 1. Princeton, N.J.: Princeton University Press, 1956.

_____. "A note on the Entscheidungsproblem." *The Journal of Symbolic Logic*, vol. 1, pp. 40–41, 1936.

Davis, Martin. *Computability & Unsolvability.* New York: Dover Publications, 1982.

Davis, Martin & Elaine Weyuker. *Computability, Complexity, and Languages.* Boston: Academic Press, 1983.

Dekker, J. C. E. *Equivalencia recursiva*. San José: Editorial CAEM, 1976.

Hennie, Fred. *Introduction to Computability.* Reading, Mass.: Addison-Wesley, 1977.

Hopcroft, John & Jeffrey Ullman. *Introduction to automata theory, languages, and computation*. Addison-Wesley, Reading Mass, 1979.

Kfoury, Moll, Arbib. *A Programming Approach to Computability.* New York: Springer-Verlag, 1982.

Piza, Eduardo. *Aritmética recursiva y algunas de sus aplicaciones*. San José: Editorial CIMPA, 2001.

Rogers, Hartley. *Theory of Recursive Function and Effective Computability.* New York: McGraw-Hill, 1967.

Soare, Robert. *Recursive Enumerate Sets and Degrees: a Study of Computable Functions and Computable Generated Sets.* Berlin: Springer-Verlag, 1980.

Tourlakis, George. *Computability.* Virginia: Reston Publishing, 1984.

Turing, Alan. "On computable numbers, with an application to the Entscheidungsproblem". *Proceedings of the London Mathematical Society*. Serie 2, vol. 42, pp. 230-265, 1936.