Examen de prueba de Historia de las Matemáticas

Se ha mostrado así que no es posible obtener una definición de calculabilidad efectiva más general que la arriba propuesta por ninguna de las dos vías que se ofrecen naturalmente, a saber, (1) definir una función como efectivamente calculable si hay un algoritmo para calcular sus valores y (2) definir una función *F* (de un entero positivo) como efectivamente calculable si, para cada entero positivo m hay un entero positivo *n* tal que *F*(*m*) = *n* es un teorema demostrable.

(Church 1936, "An unsolvable problem of elementary number theory", p. 358, citado en Torretti, *El paraíso de Cantor*, p. 375)

1. Explique por qué la definición de función efectivamente calculable (2) obedecería el programa de Hilbert (3 puntos)

La tesis de Church sostiene que una función es efectivamente calculable si es recursiva, lo cual significa que sus valores e pueden determinar mediante un número finito de operaciones o reglas procesales a partir de valores conocidos , que mantiene paralelismo con el programa de Hilbert que enuncia la necesidad de establecer un sistema finito y consistente de axiomas a partir del cual , mediante derivación en un número finito de inferencias lógicas, debe obtener un resultado sin contradicciones, esto es, se objetiviza el razonamiento matemático, en la búsqueda de la consistencia y en detrimento del "intuicionismo". Los requerimientos de Hilbert a su programa son la simplicidad, la independencia, la completitud y la consistencia. La tesis de Church se basa en un conjunto completo de teoremas enumerables y recursivos, con los que toda función calculable es recursiva o lambda-definible.

2. ¿Qué diferencias hay entre la noción de calculabilidad efectiva de Church y la noción de computabilidad de Turing? ¿Por qué son extensionalmente equivalentes? (3 puntos)

La idea de calculabilidad efectiva de Church se basa en que una función es efectivamente calculable si es posible construir un algoritmo para computar el valor de la función, pero la noción de algoritmo se establece considerando la recursividad, dado que el término logaritmo no puede definirse con claridad matemática. La computabilidad de Turing se vincula a la posibilidad de construir con una máquina de Turing el resultado de una función a partir de un argumento sobre el que opera dicha función; la máquina de Turing se concibe como un dispositivo que, mediante el uso de unas reglas establecidas, manipula símbolos siguiendo los algoritmos que se le predefinen.

La versión de Turing de la Tesis de Church, una función es computable si y sólo si es Turing-computable, prueba que calculabilidad efectiva y computabilidad de Turing son extensionalmente equivalentes, obteniendo por diferentes caminos conclusiones similares sobre la computabilidad o no computabilidad de diferentes algoritmos, llegando a demostrar de manera independiente la insolubilidad del problema de la decisión de Hilbert, utilizando ambos recorridos.

3. Explique por qué no se puede demostrar matemáticamente la tesis de Church (4 puntos)

La Tesis de Church requiere un primer análisis sobre su naturaleza, dado que no resulta fácil precisar si la tesis se trata de una conjetura o de una definición de la calculabilidad, que en cualquier caso requeriría un contraejemplo para desacreditar el enunciado o rectificar el matiz del enunciado. Entre tanto, la tesis de Church facilita la demostración de una serie de resultados notables y nos proporciona un camino para demostrar que una función no-recursiva no es calculable. En todo caso, se puede afirmar que la Tesis de Church es empírica, como lo avalan múltiples argumentos en esta línea, el primero de ellos, ya mencionado, la no existencia de contraejemplos. El segundo argumento en favor de esta línea es que el análisis conceptual sobre los procesos de cálculo efectivo concluye que solo las funciones recursivas pueden ser efectivamente calculables. Como tercer argumento debe apuntarse la invarianza, es decir, las conclusiones equivalentes obtenidas de las diversas formulaciones matemáticas llevadas a cabo y desarrolladas en la segunda mitad del siglo XX. Finalmente, debe apuntarse el éxito del desarrollo de la computación moderna como una manera de ratificar la tesis.

Examen de prueba de Historia de las Matemáticas

1) Explique por qué la definición de función efectivamente calculable (2) obedecería el programa de Hilbert (3 puntos)

(2) ¿Qué diferencias hay entre la noción de calculabilidad efectiva de Church y la noción de computabilidad de Turing? ¿Por qué son extensionalmente equivalentes? (3 puntos)

(3) Explique por qué no se puede demostrar matemáticamente la tesis de Church (4 puntos)

1) Hay varias maneras de responder a esta pregunta. Una bien sencilla es esta. La definición (2) nos dice que una función *F* (de un entero positivo) es efectivamente calculable si, para cada entero positivo m hay un entero positivo *n* tal que *F*(*m*) = *n* es un teorema demostrable. Tal como explica Torretti en la página anterior (375(, Church construye esta definición para un sistema de lógica que cumple con los desiderata del programa de Hilbert (construir un sistema formal basado en la lógica de primer orden lo bastante potente como para dar cuenta de la matemática ordinaria). Las reglas para efectuar operaciones han de ser efectivamente calculables (es decir, es decir hay un algoritmo para ejecutarlas sin preocuparse por el significado de los símbolos); el conjunto de las reglas si es infinito, ha de ser enumerable, de modo que no dé lugar a paradojas como las de los transfinitos cantorianos; y se puede establecer la correspondencia entre los teoremas demostrables en el sistema y las funciones que las representan, de modo que las verdades lógicas pueden ser demostradas con los recursos del sistema formal. Una forma alternativa de responder es explicar la definición de función efectivamente calculable en el cálculo lambda y mostrar cómo este cálculo cumple con los desiderata de Hilbert sobre razonamiento formal.

- Si hubieras explicado algo más la definición de función efectivamente calculable, habría sido perfecta.

2) La diferencia entre el concepto de calculabilidad de Church y el de computabilidad de Turing radica en que el concepto de calculabilidad pretende captar las operaciones de un calculista sin dar una definición completamente formal de las mismas, mientras que Turing las formaliza a partir del funcionamiento de su máquina. Esto, como vamos a ver, es importante para la respuesta siguiente. Una forma más técnica de verlo es a partir de la tesis de Church, que sostiene que una función definida sobre los números naturales es efectivamente calculable si y sólo si es recursiva. Pero es Turing quien muestra que toda función computable debe de ser recursiva. De ahí que se pueda probar la equivalencia extensional: toda función efectivamente calculable es computable etc, pero los matices de la definición de calculabilidad y computabilidad no son los mismos y por esos os pregunto.

- El problema con tu respuesta es que no entiendo esta frase: "pero la noción de algoritmo se establece considerando la recursividad, dado que el término logaritmo no puede definirse con claridad matemática" ¿Si el término algoritmo no puede definirse con claridad matemática, cómo se puede definir la recursividad? Caracterizas bien la posición de Turing, pero habrías debido explicar algo más la de Church.

3) Torretti da indicaciones bastante claras (pp.375-76), a las que aludía en mis comentarios de las PEC. ¿Se pueden reducir los conceptos del lenguaje natural (calculabilidad etc) de los que parte Church a una definición formal susceptible de análisis matemático? Quizá se vea más claro si pensamos en la definición de computabilidad de Turing: una función computable era aquella que pudiese procesar una Máquina de Turing. ¿Pero cómo sabemos si eso agota el concepto de computable? ¿Hacemos una encuesta para saber si a la gente le parecen "computables" todas las funciones que pueda computar una máquina de Turing? (Es decir, una prueba de la equivalencia entre el concepto informal de computabilidad y el concepto formal de Turing)
- Me gusta cómo planteas la respuesta, aunque deberías explicar algo mejor el tema de la invariancia (y, ya puestos, por qué el éxito de la computación moderna ratifica la tesis).