

# MODELOS DE COMPUTACIÓN

## CUESTIONARIO 0

Elena María Gómez Ríos

6 de octubre de 2016

### 1. El Entscheidungsproblem.

El Entscheidungsproblem (problema de decisión) fue un reto en lógica simbólica de encontrar el algoritmo general que decidiera si una fórmula de cálculo de primer orden es un teorema.

### 2. Church, Turing y el Entscheidungsproblem.

Alonzo Church y Alan Turing, de manera independiente, demostraron en 1936 que es imposible escribir el algoritmo de Entscheidungsproblem. Por lo tanto es imposible decir con un algoritmo si ciertas frases concretas de la aritmética son ciertas o falsas.

La noción de función efectivamente calculable que Church proponía era la de función  $\lambda$ -definible. Este concepto tiene su origen en un sistema lógico que en 1932-33 Church introdujo para fundamentar las matemáticas. Church empleó una notación formal que él denominó ‘cálculo lambda’ para transformar todas las fórmulas matemáticas a una forma estándar. La demostración de teoremas se convierte así en una transformación de una cadena de símbolos en otra, en cálculo lambda, según un conjunto de reglas formales. Este sistema resultó ser inconsistente, pero la capacidad para expresar-calcular funciones numéricas como términos del sistema llamó pronto la atención de él y sus colaboradores. Así, Church habla en 1934 de la noción de función ‘efectivamente calculable’ como función  $\lambda$ -definible. Además, bajo su tesis *Tesis de Church* y utilizando la noción de función  $\lambda$ -definible, dio ejemplos de problemas de decisión irresolubles, y demostró que el Entscheidungsproblem era uno de esos problemas.

Turing reflexionó sobre la frase de Newman “un proceso mecánico” en 1936 publicó su célebre trabajo “Números Computables: Una Aplicación al Entscheidungsproblem”. Turing argumentó el Entscheidungsproblem podía atacarse con la ayuda de una máquina, al menos con el concepto abstracto de máquina. El propósito de Turing era reducir los cálculos a sus rasgos esenciales más escuetos, describiendo de un modo sencillo algunos procedimientos básicos que son manifiestamente efectivos y a los que pueda reducirse cualquier procedimiento efectivo. Turing utilizó su concepto de máquina para demostrar que existen funciones que no son calculables por un método definido y en particular, que el Entscheidungsproblem era uno de esos problemas.

### 3. Vida y contribución de Turing a la computabilidad.

Alan Turing, pionero de las modernas Ciencias de la Computación y la inteligencia Artificial, nació en 1912 en Paddington, Reino Unido. Es considerado uno de los padres de la ciencia de la computación y precursor de la informática moderna.

Alan Turing introdujo el concepto de máquina de Turing en el trabajo *On computable numbers, with an application to the Entscheidungsproblem*, publicado por la Sociedad Matemática de Londres

en 1936, en el que se estudiaba la cuestión planteada por David Hilbert sobre si las matemáticas son decidibles, es decir, si hay un método definido que pueda aplicarse a cualquier sentencia matemática y que nos diga si esa sentencia es cierta o no. Turing ideó un modelo formal de computador, la máquina de Turing, y demostró que existían problemas que una máquina no podía resolver.

Con este aparato extremadamente sencillo es posible realizar cualquier cómputo que un computador digital sea capaz de realizar.

Mediante este modelo teórico y el análisis de la complejidad de los algoritmos, fue posible la categorización de problemas computacionales de acuerdo a su comportamiento, apareciendo así, el conjunto de problemas denominados P y NP, cuyas soluciones pueden encontrarse en tiempo polinómico por máquinas de Turing deterministas y no deterministas, respectivamente.

Precisamente, la tesis de Church-Turing formulada por Alan Turing y Alonzo Church, de forma independiente a mediados del siglo XX caracteriza la noción informal de computabilidad con la computación mediante una máquina de Turing.

La idea subyacente es el concepto de que una máquina de Turing puede verse como un autómata ejecutando un procedimiento efectivo definido formalmente, donde el espacio de memoria de trabajo es ilimitado, pero en un momento determinado sólo una parte finita es accesible.

Durante la segunda guerra mundial, trabajó en descifrar los códigos nazis, particularmente los de la máquina Enigma, y durante un tiempo fue el director de la sección Naval Enigma de Bletchley Park. Se ha estimado que su trabajo acortó la duración de esa guerra entre dos y cuatro años. Tras la guerra, diseñó uno de los primeros computadores electrónicos programables digitales en el Laboratorio Nacional de Física del Reino Unido y poco tiempo después construyó otra de las primeras máquinas en la Universidad de Mánchester.

En el campo de la inteligencia artificial, es conocido sobre todo por la concepción del test de Turing (1950), un criterio según el cual puede juzgarse la inteligencia de una máquina si sus respuestas en la prueba son indistinguibles de las de un ser humano.

## **4. Tesis de Church-Turing. Importancia.**

En teoría de la computabilidad, la tesis de Church-Turing formula hipotéticamente la equivalencia entre los conceptos de función computable y máquina de Turing, que expresado en lenguaje corriente vendría a ser “todo algoritmo es equivalente a una máquina de Turing”. No es un teorema matemático, es una afirmación formalmente indemostrable, una hipótesis que, no obstante, tiene una aceptación prácticamente universal.

Aunque se asume como cierta, la tesis de Church-Turing no puede ser probada ya que no se poseen de los medios necesarios, por eso es una tesis. Ello debido a que “procedimiento efectivo” y “algoritmo” no son conceptos dentro de ninguna teoría matemática y no son definibles fácilmente. La evidencia de su verdad es abundante pero no definitiva. Precisamente la tesis de Church establece que la definición de algoritmo o procedimiento efectivo es una máquina de Turing. La idea abstracta de una máquina que funciona como parámetro para decidir cuándo algo es un algoritmo o procedimiento efectivo es de gran valor, esto es una máquina de Turing.

## **5. Tesis de Church-Turing. Variantes y detractores.**

### **5.1. Variantes.**

#### **5.1.1. Principio Church-Turing-Deutsch**

En 1985, Deutsch afirma que la tesis de Church es demasiado general en comparación con algunos de los principios físicos conocidos. Deutsch propone referirse a las “funciones que de manera natural

sean consideradas computables como “funciones que puedan ser computadas por un sistema físico real”, ya que de esta forma lo que se quiere expresar es mucho más concreto. Todo esto le permitió introducir la concepción del diseño de una máquina de Turing.

Enunciado de la tesis Church-Turing-Deutsch: “Todos los sistemas físicos finitos comprensibles pueden ser simulados por una máquina de computación universal que opere en pasos finitos”.

#### **5.1.2. Tesis Física de Church-Turing (PCTT)**

Enunciado de la tesis física de Church-Turing (PCTT): “Every function that can be physically computed can be computed by a Turing machine” (es decir, cualquier función que pueda ser físicamente computable, puede ser computada por una máquina de Turing).

En 2002 este teorema fue refutado por Willem Fouché, puesto que descubrió que las máquinas de Turing no pueden obtener todos los valores posibles, unidimensionales, de los números racionales.

#### **5.1.3. Strong Church-Turing thesis (SCTT)**

En 1997, Bernstein y Vazirani expusieron la tesis Strong Church-Turing (SCTT), cuyo enunciado es: “Any ‘reasonable’ model of computation can be efficiently simulated on a probabilistic Turing machine”.

Los estudios actuales sugieren que este enunciado es falso, puesto que no es posible capturar todas las formas de computación tanto existentes como posibles mediante un algoritmo. Por ejemplo, los protocolos interactivos no son algoritmos.

Uno de las principales razones de que se llegara a la conclusión de que dicho enunciado era falso, fue que Peter Shor demostró que el problema de factorizar los números primos se puede resolver usando un ordenador cuántico, mientras que no es posible encontrar una solución al mismo problema usando una máquina probabilística de Turing.

#### **5.1.4. Tesis de Church-Turing Extendida (ECT)**

Esta tesis afirma que la máquina de Turing es tan eficiente como un computador. Es decir, si alguna función es computable por algún dispositivo hardware para una entrada de tamaño  $n$ , entonces dicha función es computable por una máquina de Turing en  $(T(n))^k$  para algún  $k$  fijo (dependiente del problema).

#### **5.1.5. La Tesis de Zuse-Fredkin**

Esta tesis fue enunciada formalmente en 1960 por Zuse y Fredkin. La idea principal de la misma es que “el universo es un autómata celular”. Si una persona considera que la tesis de Church es verdadera entonces no puede encontrar ninguna razón que le indica que la tesis de Zuse-Fredkin es falsa, y viceversa.

#### **5.1.6. La Tesis M**

La tesis M dice: Todo aquello que pueda ser calculado por una máquina (en caso de que se trabaje con datos finitos de acuerdo a un número finito de instrucciones) es computable por una máquina de Turing.

### **5.2. Detractores.**

Es claro que es más “fácil” probar la falsedad de la tesis que la verdad de la misma. Basta con exponer un método efectivo o algoritmo que no sea computable en el sentido de Turing.

Aunque exponer un algoritmo que no sea Turing-computable no es tan fácil, pero, es más “fácil” que probar la verdad de la tesis, ya que parece imposible negar que sea verdadera a pesar de que eso es una posibilidad lógica.

Existe una tesis relativizada de Church-Turing que depende de los grados de Turing definidos por máquinas de Turing con oráculos. Los oráculos son medios formales que suponen que se le facilita cierta información a la máquina de Turing para resolver algún problema, esto define una jerarquía que se ha estudiado tanto en la teoría de la Computabilidad como en la Teoría de la Complejidad.