Otros formalismos

Autómatas y lenguajes formales

26 de noviembre de 2023

1. Los otros modelos

Ahora que estamos a la distancia[1] y nos parece tan lejana la vida en la facultad (en algunas cosas para bien, en otras no) iniciaré esta nota con una anécdota. Tuve en la facultad un profesor de geometría y álgebra lineal que aún está ahí en el departamento de matemáticas. En su época de estudiante de ciencias la costumbre era estudiar las dos carreras de física y matemáticas (ya ven que en la facultad casi no se da eso), él por ser parte del *hype* entró a ambas carreras, originalmente estudiaba física.

Como era de esperar la demanda de tiempo y trabajo era demasiada, a medio semestre ya estaba flaco, ojeroso y sin ilusiones, no soportó tanto y decidió abandonar una carrera. Lo raro fue que no decidió abandonar la carrera extra, que era matemáticas, prefirió renunciar a física. Sus razones eran las siguientes: la ventaja de la matemática sobre la física es que puedes tomar varios caminos, el que te agrade, el que se te ocurra, y llegar al resultado, más de uno de esos caminos es correcto. En cambio en la física es todo más rígido, hay un sólo camino (por lo regular) y no hay mucha posibilidad para cambiar la ruta.

Esto se debe mucho a la experiencia que tuvo este profesor en ese entonces estudiante, seguro hay áreas de la física que dan libertad. Pero en lo que concuerdo completamente es que en las matemáticas hay un poco más de libertad para enfrentar un problema, más de una persona puede llegar al resultado correcto por caminos distintos y ser válido. Lo mismo sucede en teoría de la computación (que como han podido ver en este y otros cursos tiene un origen en común con las matemáticas), no sólo hay un único modelo computacional.

Ahora, sí yo quisiera hacer otro modelo ¿cómo podría saber que es correcto? '¿Cómo saber que en efecto estos modelos distintos hacen lo mismo? La idea de base es la *computabilidad efectiva*, el rasgo en común que tienen estos modelos. La tesis de Church-Turing (que es más una afirmación) dice:

Tesis de Church-Turing 1 *Toda función es calculable efectivamente sí y sólo sí es calculable por una máquina de Turing.*

Aquí van a decir que ya cayó más pronto un hablador que un cojo, hay una posición preferente a la máquina de Turing ¿no que muy incluyente? Es cierto, la máquina de Turing tiene una posición preferente, de acuerdo al texto por su simplicidad y claridad, pero todos los modelos equivalentes pueden representarse

[1]Estas notas fueron hechas como apoyo a las clases en línea de la facultad, por eso la mención a las clases a distancia. Dejo la mención para tener presente esa época en que se empezaron a redactar.

por una máquina de Turing, entonces, como lo hemos hecho en el curso, la máquina de Turing es la que tomamos como paradigma, pero los formalismos que veremos son equivalentes. Si para ustedes no fue tan claro y les queda más claro por estos modelos, entonces podrán decir que se toma la máquina de Turing por pura formalidad. Lo que veremos en estas opciones es que en algunos casos son más parecidos a un lenguaje de programación de los que ahora usamos.

2. Funciones recursivas μ

Una buena pregunta sería, ¿cuál es el mínimo de funciones necesarias para definir a todas las funciones computables? ¿Cuáles son las funciones de ese conjunto mínimo? De acuerdo a Gödel esas funciones numérico-teóricas $\mathbb{N}^k \to \mathbb{N}$ son:

1. *Sucesor.* La función $\mathbf{s} : \mathbb{N} \to \mathbb{N}$ dadas por $\mathbf{s}(x) = x + 1$ es computable.

2. *Cero.* La función $\mathbf{z} : \mathbb{N}^0 \to \mathbb{N}$ dada por $F() = 0$ es computable.

3. *Proyecciones.* Las funciones $\pi_k^n : \mathbb{N}^n \to \mathbb{N}$ dadas por $\pi_k^n(x_1, ..., x_n) = x_k,\ 1 \leq k \leq n$, son computables.

4. *Composición.* Si $f : \mathbb{N}^k \to \mathbb{N}$ y $g_1, ...g_k : \mathbb{N}^n \to \mathbb{N}$ son computables, entonces también lo es la función $f \circ (g_1, ..., g_k) : \mathbb{N}^n \to \mathbb{N}$ que en la entrada $\overline{x} = x_1, ..., x_n$, da

$$f(g_1(\overline{x}), ..., g_k(\overline{x}))$$

5. *Recursión primitiva.* Si $h_i : \mathbb{N}^n \to \mathbb{N}$ y $g_i : \mathbb{N}^{n+k+1} \to \mathbb{N}$ son computables, $1 \leq i \leq k$, entonces también lo son las funciones $f_i : \mathbb{N}^{n+1} \to \mathbb{N},\ 1 \leq i \leq k$, definidas por inducción mutua de la siguiente manera:

$$\begin{aligned} f_i(0, \overline{x}) &\stackrel{def}{=} h_i(x) \\ f_i(x+1, \overline{x}) &\stackrel{def}{=} g_i(x, \overline{x}, f_1(x, \overline{x}), ..., f_k(x, \overline{x})), \end{aligned}$$

   donde $\overline{x} = x_1, ..., x_n$.

6. *Minimización no acotada.* Si $g : \mathbb{N}^{n+1} \to \mathbb{N}$ es computable, entonces también lo es la función $f : \mathbb{N}^n \to$ que con la entrada $\overline{x} = x_1, ..., x_n$ de al menos y tal que $g(z, \overline{x})$ esté definida para todas las $z \leq y$ y $g(y, \overline{x}) = 0$ si tal y, y está indefinida de otra manera. Esto se denota como:

$$f(\overline{x}) = \mu y.(g(y, \overline{x}) = 0)$$

Algunas de estas funciones suenan demasiado enredadas, pero vamos viendo por pasos. Las primeras cuatro parecen bastante directas, el *sucesor* nos da la posibilidad de construir cualquier número a partir de una base, e incluso nos es útil para definir operaciones más complicadas, como la suma de dos o más números, la multiplicación, el factorial (no se preocupen, lo vamos viendo). La operación lo que hace es sumar un 1 al número que se le dé como entrada a la función. Noten que es una función unaria, sólo necesita de un número natural de entrada, tal como si fuera una cinta de entrada muy sencilla.