Algoritmos, Azar y Autómatas
Resolución Ejercicios 9 a 13 (aleatoriedad)

Manuel Panichelli

November 21, 2021

Resultados previos

Ejercicio 9

Ejercicio 10

Dar un algoritmo que permite computar Ω con un oráculo para el problema de la detención.

Recuerdo la definición de Ω, la suma de las potencias de 2 de las longitudes de todos los programas que terminan. Si U es una máquina de Turing universal,

$$\Omega = \sum_{U(p)\downarrow} 2^{-|p|}$$

Para computar Ω voy a dar una función que computa los primeros n dígitos, $\Omega[1 \dots n]$, y luego voy incrementando n e imprimiendo los resultados.

Sea g una enumeración de todos los programas que terminan (se podría obtener computablemente mediante un método como *dovetailing*), defino una aproximación de Ω hasta el m-ésimo programa,

$$\alpha_m = \sum_{j=1}^{m} 2^{-|g(j)|}$$

Lo que me gustaría saber es para qué m α_m tiene los primeros n dígitos *definitivos*, es decir $\Omega[1 \dots n] = \alpha_m[1 \dots n]$. Para ello, debo verificar que

$$\alpha_m[1 \dots n] \stackrel{?}{=} \alpha'_m[1 \dots n] \ \forall m' > m$$

Es decir, no importa que sigamos considerando más programas, los primeros n dígitos no van a cambiar. Como no es algo finito, no lo podemos computar con un algoritmo, pero acá es donde nos salva el oráculo de Halt. La siguiente función logra lo buscado

function Q(m, n)
    $m' \leftarrow m + 1$
    while true **do**
        if $\alpha_m[1 \dots n] \neq \alpha'_m[1 \dots n]$ **then**
            break
        $m' \leftarrow m' + 1$

- Si `OraculoHalt(Q(m, n)) = true` (es decir, Q termina) es porque existía m tal que cambiaban los primeros n dígitos, y por lo tanto no eran definitivos.
- Si `OraculoHalt(Q(m, n)) = false` (es decir, Q *no* termina), entonces no existe m tal que cambien los primeros n dígitos, y por lo tanto son definitivos.

El algoritmo final es el siguiente, donde la función sin argumentos Print Ω imprime Ω segmento inicial por segmento inicial.

function PRINT Ω
    for $n = 1, 2 \dots$ **do**
        print Ω(n)

```
function Ω(n)
    for m = 1, 2 ... do
        if ¬ OraculoHalt(Q(m, n)) then
            return α_m[1 ... n]
function Q(m, n)
    m' ← m + 1
    while true do
        if α_m[1 ... n] ≠ α'_m[1 ... n] then
            break
        m' ← m' + 1
```

Ejercicio 11

Ejercicio 12

Ejercicio 13