1 2 3 4
A A A A
10
HOJA N° 1

1)

Veamos primero que A no es computable por el absurdo. Para eso supongamos que sí lo es. Es decir, que

$$A(\langle x,y\rangle)=\begin{cases}1 & \text{si } \Phi_x^{(1)}(z)\uparrow \ \forall z\in\mathbb{N} \text{ y } STP^{(1)}(0,y,y)=1\\ 0 & \text{si no}\end{cases}$$

es computable.

Defino la siguiente función:

$$g(u,v)=\begin{cases}1 & \text{si } A(\langle u,d\rangle)=1\\ \uparrow & \text{si no}\end{cases}$$

donde d es un número de programa tal que $STP^{(1)}(0,d,d)=1$. ⊛

Esta función es parcial computable ya que se puede escribir un programa que compute $A(\langle u,d\rangle)$ (lo cual es posible porque A es computable por hipótesis) y devuelva 1 si $A(\langle u,d\rangle)=1$ o se quede ciclando infinitamente si $A(\langle u,d\rangle)=0$.

Entonces, por el teorema de la recursión existe un número de programa e tal que $\Phi_e(v)=g(e,v)=\begin{cases}1 & \text{si } A(\langle e,d\rangle)=1\\ \uparrow & \text{si no}\end{cases}=\begin{cases}1 & \text{si } \Phi_e^{(1)}(z)\uparrow\ \forall z\in\mathbb{N}\\ \uparrow & \text{si no}\end{cases}$

Pero luego tengo que:

- $\Phi_e(z)\uparrow\ \forall z\in\mathbb{N} \Rightarrow \Phi_e(v)=1 \Rightarrow \exists z\in\mathbb{N}$ tq $\Phi_e(z)\downarrow \Rightarrow \neg(\Phi_e(z)\uparrow\ \forall z\in\mathbb{N})$
- $\neg(\Phi_e(z)\uparrow\ \forall z\in\mathbb{N}) \Rightarrow \Phi_e(v)\uparrow \Rightarrow \Phi_e(z)\uparrow\ \forall z\in\mathbb{N}$

Esto es absurdo. Por lo tanto A no puede ser computable.

Veamos ahora que A es co-c.e. Es decir, que

$\bar{A}=\{\langle x,y\rangle \mid \exists z\in\mathbb{N}$ tq $\Phi_x^{(1)}(z)\downarrow$ ó $STP^{(1)}(0,y,y)=1\}$ es c.e. Para ello notamos primero que $\bar{A}=P\cup Q$ donde $P=\{\langle x,y\rangle \mid \exists z\in\mathbb{N}$ tq $\Phi_x^{(1)}(z)\downarrow\}$ y $Q=\{\langle x,y\rangle \mid STP^{(1)}(0,y,y)=1\}$.

P es c.e. ya que es el dominio de la función

NOTA

$p(\langle x,y\rangle) = \min_t STP^{(1)}(l(t), x, r(t))$, que es parcial computable por ser una minimización no acotada de funciones p.r. P es el dominio de $p(\langle x,y\rangle)$ ya que si $\exists\, z \in \mathbb{N}$ t.q. $\Phi_x(z)\downarrow$ entonces $STP^{(1)}(l(t), x, r(t)) = 1$ para algún $t \in \mathbb{N}$; si $\Phi_x(z)\uparrow\ \forall\, z \in \mathbb{N}$ entonces $STP^{(1)}(z, x, t) = 0\ \forall z, t \in \mathbb{N}$ y $p(\langle x,y\rangle)$ será indefinido.

Por el otro lado, Q es computable porque su función característica es $Q(\langle x,y\rangle) = STP^{(1)}(0, y, y)$, que es p.r. Luego Q es c.e.

Por lo tanto, como P y Q son c.e., y $\bar{A} = P \cup Q$, $\bar{A}$ también es c.e. (i.e. A es co-c.e.).

A no puede ser c.e. ya que si lo fuera, como $\bar{A}$ es c.e., sería computable. Pero ya vimos que esto no es cierto. Por lo tanto, A es solamente co-c.e.

(*) El programa con una única instrucción $X_1 \leftarrow X_1 + 1$ tiene número 3 y termina en 1 paso. Luego $STP^{(1)}(0, d, d) = 1$ para $d = 3$.

A

2

②

Veamos que $f(x)$ es p.r. dando el esquema de recursión primitiva.

$$f(0) = n(0) = 0$$

$$f(x) = g(f(x-1), x) \quad \text{donde} \quad g(u,x) = (u+x) \cdot STP^{(1)}(x, x, u+x)$$

$n(x)$ es p.r. porque es una función inicial y $g(u,x)$ es p.r. porque es composición de funciones p.r. (suma, multiplicación y STP).

Por lo tanto, f es p.r., lo cual implica también que es computable.

③

Sea $h(x) = g(x,1) = \begin{cases} 1 & \text{si } \phi_x^{(1)}(x)\downarrow,\ \phi_x^{(2)}(x,1)\downarrow \text{ y } \phi_x^{(1)}(x) < \phi_x^{(2)}(x,1) \\ 0 & \text{si no} \end{cases}$

Veamos que $h(x)$ no es computable. Si probamos esto, demostramos que $g(x,y)$ no puede ser computable ya que si lo fuera se podría computar $h(x)$ con $g(x,y)$.

Probamos que $h(x)$ no es computable por el absurdo. Supongamos que sí lo es y definamos

$$f(u,x,y) = \begin{cases} 1 \dot{-} y & \text{si } h(u) = 1 \\ y & \text{si no} \end{cases}$$

f es computable ya que basta que un programa compute $h(u)$ (el cual es computable por hipótesis) y ponga en la variable de salida $1 \dot{-} y$ ($\cdot \dot{-} \cdot$ es computable porque es p.r.) si $h(u)=1$ o y si $h(u)=0$.

Por el teorema de la recursión existe $e \in \mathbb{N}$ tq $\phi_e(x,y) = f(e,x,y)$. Luego

$$\phi_e(x,y) = \begin{cases} 1 \dot{-} y & \text{si } h(e)=1 \\ y & \text{si no} \end{cases} = \begin{cases} 1 \dot{-} y & \text{si } \phi_e^{(1)}(e)\downarrow, \phi_e^{(2)}(e,1)\downarrow \text{ y } \phi_e^{(1)}(e) < \phi_e^{(2)}(e,1) \\ y & \text{si no} \end{cases}$$

Pero entonces tengo que:

- $h(e) = 1 \Rightarrow \phi_e(x,y) = 1 \dot{-} y \Rightarrow \phi_e(e) = 1 \dot{-} 0 = 1 > 0 = 1 \dot{-} 1 = \phi_e(e,1) \Rightarrow h(e) = 0$
- $h(e) = 0 \Rightarrow \phi_e(x,y) = y \Rightarrow \phi_e(e)\downarrow,\ \phi_e(e,1)\downarrow \text{ y } \phi_e(e) = 0 < 1 = \phi_e(e,1) \Rightarrow h(e) = 1$

Esto es absurdo. Por lo tanto $h(x)$ no es computable y, por lo dicho al principio, $g(x,y)$ tampoco.

A

4)

a. Verdadero

Primero veamos que A es infinito. Como f es parcial computable existe $e \in \mathbb{N}$ tq $\Phi_e^{(1)} = f$. Al programa P que codifica e le podemos agregar cualquier instrucción que no modifique Y o que si la modifica que recupere su valor original (el que resulta de ejecutar P), y que no indefina el programa. Por ejemplo.

- $X_i \leftarrow X_i + 1$ para todo $i > 1$
- $Y \leftarrow Y + 1$
  $Y \leftarrow Y - 1$

Este nuevo programa también computará a f. Como podemos hacer esto tantas veces como queramos, hay infinitos programas que computan f. Luego A es infinito.

Defino ahora el siguiente conjunto, dado el programa P de antes:

$B = \{x \mid x$ es el número de un programa que resulta de agregarle a P una cantidad $K > 1$ de instrucciones $X_1 \leftarrow X_1 + 1$ al final$\}$.

Este conjunto es computable ya que su función característica es

$$B(x) = (\forall i)_{\leq |e+1|} \left((e+1)[i] = (x+1)[i] \right) \wedge (\forall j)_{\leq |x+1|} \left(j \leq |e+1| \vee (x+1)[j] = K \right)$$

donde K es el número de la instrucción $X_1 \leftarrow X_1 + 1$. $B(x)$ es composición de funciones p.r., por lo cual es p.r.

Por lo dicho antes, todo $x \in \mathbb{N}$ es el número de un programa que computa f y B es infinito. Luego B es un subconjunto infinito de A que es computable.

NOTA

b. Falso

Supongamos que la afirmación es verdadera y veamos que se produce una contradicción.

Sea la función f de la afirmación $f(x) = HALT(x,x)$. Por hipótesis existen funciones p.r. (y por lo tanto computables) $g_0, g_1, \ldots$ tales que $g_i(x) = HALT(x,x) \quad \forall\, x \leq i$, y existe una función computable h tal que $\Phi^{(1)}_{h(i)} \equiv g_i \quad \forall\, i \in \mathbb{N}$.

Defino la función $T(x,i) = \min_t (STP^{(1)}(x, h(i), t) = 1)$. Esta función es computable ya que se define por composición y minimización propia. Esto último quiere decir que es una minimización no acotada pero para la cual el argumento minimizado existe. Esto es cierto ya que $\Phi_{h(i)} \equiv g_i$ es computable, por lo que terminará en alguna cantidad finita de pasos y $STP^{(1)}(x, h(i), t)$ será 1.

Pero entonces podemos definir f como

$f(x) = r(SNAP^{(1)}(x, h(x), T(x,x)))[1]$.

Es decir, el valor de la variable Y del programa con número $h(x)$ en el paso $T(x,x)$. Esta igualdad vale porque el programa con número $h(x)$ es el que computa g_x y termina en a lo sumo $T(x,x)$ pasos con entrada x (por definición de $T(x,i)$) y $g_x(y) = HALT(y,y) \quad \forall\, y \leq x$ (en particular $g_x(x) = HALT(x,x) = f(x)$).

Pero entonces $f(x)$ es computable, pues es composición de funciones computables. Por lo tanto $HALT(x,x)$ es computable. Esto es absurdo. ✓

¡Está bien! Aunque era más sencillo plantear simplemente

$$f(x) = \Phi^{(1)}_{h(x)}(x)$$