Funciones $\mathcal{S}$-computables
Lógica y Computabilidad

Christian G. Cossio-Mercado

Departamento de Computación,
Facultad de Ciencias Exactas y Naturales,
Universidad de Buenos Aires

18 de septiembre de 2020

Temario

Temario

Ejercicio 1: Números que cumplen un predicado

Sea $p(x)$ un predicado computable. Demostrar que la siguiente función es parcial computable:

$$EX_p(r) = \begin{cases} 1 & \text{si hay al menos } r \text{ números } n \text{ tales que } p(n) = 1 \\ \uparrow & \text{si no} \end{cases}$$

Ejercicio 2: Multiplicación de dos números

Exhibir un pseudo-programa P en el lenguaje $\mathcal{S}$ que compute la función $* : \mathbb{N}^2 \to \mathbb{N}$ definida por $*(x, y) = x \cdot y$.

- ¿Qué es necesario hacer para obtener un programa en $\mathcal{S}$ a partir de P?
- ¿Qué hay que tener en cuenta al hacerlo?

Ejercicio 2: Multiplicación de dos números

Sea Q el programa en $\mathcal{S}$ obtenido a partir de P. Caracterizar las siguientes funciones.

- $\Psi_Q^{(1)} : \mathbb{N} \rightarrow \mathbb{N}$
- $\Psi_Q^{(2)} : \mathbb{N}^2 \rightarrow \mathbb{N}$
- $\Psi_Q^{(3)} : \mathbb{N}^3 \rightarrow \mathbb{N}$

Temario

Repaso de las teórica 5

Ejercicio 3: Cantidad de pasadas por una instrucción

Sea P un programa que contiene exactamente una vez la instrucción siguiente (etiquetada o no):´

$$\begin{gathered} \dots \\ Y \leftarrow Y + 1 \\ \dots \end{gathered}$$

Demostrar que para todo $x \in \mathbb{N}$, si $\Psi_P^{(1)}(x) \downarrow$, entonces al ejecutar P con x como única entrada se pasa por dicha instrucción por lo menos $\Psi_P^{(1)}(x)$ veces

- ¿Cuál es la intuición?
- ¿Cómo lo formalizaríamos?

Ejercicio 3: Cantidad de pasadas por una instrucción

Dado cualquier $x \in \mathbb{N}$ tal que $\Psi_P^{(1)}(x) \downarrow$,
donde $d_0, \ \ldots, \ d_n$ es el cómputo del programa P a partir de la entrada $x \in \mathbb{N}$
siendo $d_0 = (1, \sigma_1)$ la descripción inicial correspondiente
Llamando:

- m al índice de la instrucción de P que incrementa la variable Y.
- k_i (para $i = 0, \ \ldots, \ n$) a la cantidad de veces que se pasó por la m-ésima instrucción de P tras i pasos de su ejecución
- $d_i[Y]$ al valor de Y indicado por la descripción d_i.

Queremos ver que para todo $i = 0, \ \ldots, \ n$, se cumple $k_i \geqslant d_i[Y]$

Ejercicio 3: Cantidad de pasadas por una instrucción

Queremos ver que para todo $i = 0, \ldots, n$, se cumple $k_i \geqslant d_i[Y]$

- **Caso base**
  Por definición, sabemos que en el estado inicial, $d_0[Y] = 0$ y $k_0 = 0$ (no se ejecutaron instrucciones). Luego, $k_0 \geqslant d_0[Y]$
- **Paso Inductivo**
  Suponemos $k_i \geqslant d_i[Y]$, para $i \in \{0, \ldots, n-1\}$. Queremos ver que $k_{i+1} \geqslant d_{i+1}[Y]$

Ejercicio 3: Cantidad de pasadas por una instrucción

Paso Inductivo

Suponemos $k_i \geqslant d_i[Y]$, para $i \in \{0, \ldots, n-1\}$. Queremos ver que $k_{i+1} \geqslant d_{i+1}[Y]$

Sabemos que $d_i = (j_i, \sigma_i)$, donde j_i es el índice de la próxima instrucción a ejecutar.

Caso 1 Si $j_i = m$, entonces

$$k_{i+1} = k_i + 1 \geqslant d_i[Y] + 1 = d_{i+1}[Y],$$

ya que tanto el valor de Y como la cantidad de veces que se ejecutó la m-ésima instrucción se incrementan en 1

Caso 2 Si $j_i \neq m$, la instrucción puede no involucrar a la variable Y, que lo haga y no la modifique (condicional) o la decremente en 1

En los tres casos, se da que $d_i[Y] \geqslant d_{i+1}[Y]$

Como además $k_{i+1} = k_i$, tenemos que

$$k_{i+1} = k_i \geqslant d_i[Y] \geqslant d_{i+1}[Y]$$

Temario

Repaso de las teórica 5

Ejercicio 4: Programa con saltos al final

Decimos que un programa P *tiene un salto al final* si posee alguna instrucción de la forma:

$$\text{IF } V \neq 0 \text{ GOTO } L$$

donde ninguna instrucción de P está etiquetada con L.
Demostrar que el siguiente predicado es primitivo recursivo.

$$r_1(x) = \begin{cases} 1 & \text{si el programa cuyo número es } x \text{ tiene un salto al final} \\ 0 & \text{si no} \end{cases}$$

Ejercicio 4: Programa con saltos al final

Posible solución

Si definimos el siguiente predicado

$$\tilde{r}_1(x, y) = \begin{cases} 1 & \text{si la instrucción codificada por } y \text{ es un salto al final en el prog. } x \\ 0 & \text{si no} \end{cases}$$

Entonces, si $\tilde{r}_1$ fuera p.r., entonces $r_1(x)$ sería parcial computable:

$$r_1(x) = (\exists t)_{\leqslant |x+1|}\Big(t > 0 \wedge \tilde{r}_1\big(x, (x+1)[t]\big)\Big)$$

Ejercicio 4: Programa con saltos al final

Posible solución

Así, si definimos el predicado $\tilde{r}_1$ como:

$$\tilde{r}_1(x,\ y) = l(r(y)) \geqslant 3 \wedge \neg(\exists t)_{\leqslant |x+1|}\Big(t > 0 \wedge l\big((x+1)[t]\big) = l(r(y)) \dot{-} 2\Big)$$

Al estar definido por composición de funciones p.r. (como vimos en clases anteriores), $\tilde{r}_1$ es p.r. Luego, r_1 es p.r.

Ejercicio 5: Programa que respeta sus entradas

Decimos que un programa P *respeta sus entradas* si para ningún $i \in \mathbb{N}$ tiene instrucciones de los tipos:

$$X_i \leftarrow X_i + 1 \qquad \text{o} \qquad X_i \leftarrow X_i \dot{-} 1$$

Demostrar que el siguiente predicado es primitivo recursivo:

$$r_2(x) = \begin{cases} 1 & \text{si el programa cuyo número es } x \text{ respeta sus entradas} \\ 0 & \text{si no} \end{cases}$$

Queda de tarea

Se pueden dar cuenta de que se resuelve muy parecido al anterior

Temario

Ejercicio 6: Programa termina antes que otro

Demostrar que la siguiente función es $\mathcal{S}$-parcial computable.

$$f_1(x,\ y,\ z) = \begin{cases} 1 & \text{si la ejecución de } \Phi_x^{(1)}(z) \text{ termina estrictamente en menos pasos que la ejecución de } \Phi_y^{(1)}(z) \\ \uparrow & \text{si no} \end{cases}$$

Ejercicio 6: Programa termina antes que otro

Posible solución I

$$
\begin{array}{ll}
[A] & Z_2 \leftarrow \mathrm{STP}^{(1)}(X_3,\ X_1,\ Z_1) \\
 & \text{IF } Z_2 \neq 0 \text{ GOTO } B \\
 & Z_1 \leftarrow Z_1 + 1 \\
 & \text{GOTO } A \\
[B] & Z_2 \leftarrow \mathrm{STP}^{(1)}(X_3,\ X_2,\ Z_1) \\
 & \text{IF } Z_2 \neq 0 \text{ GOTO } B \\
 & Y \leftarrow 1
\end{array}
$$

Ejercicio 6: Programa termina antes que otro

Posible solución II

Podemos usar lo siguiente:
Si un predicado $p : \mathbb{N}^n \to \{0, 1\}$ es $\mathcal{S}$-computable, entonces es $\mathcal{S}$-parcial computable el *existencial no acotado* sobre este, es decir,

$$(\exists t)\ p(x_1,\ \ldots,\ x_n,\ t) = \begin{cases} 1 & \text{si existe } t \text{ tal que } p(x_1,\ \ldots,\ x_n, t) = 1 \\ \uparrow & \text{si no} \end{cases}$$

Para obtener el predicado pedido podemos...

a) Hacer un programa que compute la función anterior desde cero

b) Aprovechar la minimización no acotada que vimos anteriormente

Ejercicio 6: Programa termina antes que otro

Posible solución II

a) Hacer un programa que compute la función anterior desde cero

$$
\begin{array}{ll}
[A] & Z_2 \leftarrow p(X_1,\ \dots,\ X_n,\ Z_1) \\
 & \text{IF } Z_2 \neq 0 \text{ GOTO } B \\
 & Z_1 \leftarrow Z_1 + 1 \\
 & \text{GOTO } A \\
[B] & Y \leftarrow 1
\end{array}
$$

b) Aprovechar la minimización no acotada que vimos anteriormente

$$
\begin{array}{l}
Z_1 \leftarrow \min\limits_{t}\ p(x_1,\ \dots,\ x_n,\ t) \\
Y \leftarrow 1
\end{array}
$$

Ejercicio 7: Punto fijo

Dada una función $f : \mathbb{N} \to \mathbb{N}$, decimos que $x \in \mathbb{N}$ es un **punto fijo** de f si $f(x) = x$. Demostrar que la siguiente función es $\mathcal{S}$-parcial computable:

$$f_2(x) = \begin{cases} 1 & \text{si } \Phi_x^{(1)} \text{ tiene algún punto fijo} \\ \uparrow & \text{si no} \end{cases}$$

Ejercicio 8: Salida distinta con misma entrada

Demostrar que, para todo $n \in \mathbb{N}$, $n \geqslant 1$, la siguiente función es $\mathcal{S}$-parcial computable.

$$f(x,y) = \begin{cases} 1 & \text{si existe } \bar{z} \in \mathbb{N}^n \text{ tq } \Phi_x^{(n)}(\bar{z}) \downarrow, \Phi_y^{(n)}(\bar{z}) \downarrow \text{ y } \Phi_x^{(n)}(\bar{z}) \neq \Phi_y^{(n)}(\bar{z}) \\ \uparrow & \text{si no} \end{cases}$$