Resueltos Lógica y Computabilidad

Ignacio E. Losiggio

February 15, 2019

1 Práctica 2 — Funciones $\mathcal{S}$–computables

1.1 Ejercicio 1

1.1.1 Definir *macros* para las siguientes pseudo-instrucciones (con su interpretación natural) e indicar en cada caso qué etiquetas se asumen "frescas"

- $V_i \leftarrow k$

$$
\begin{array}{ll}
[R] & V_i \leftarrow V_i - 1 \\
 & IF\ V_i \neq 0\ GOTO\ R \\
 & V_i \leftarrow V_i + 1 \\
 & \quad \vdots \qquad k \text{ veces} \\
 & V_i \leftarrow V_i + 1
\end{array}
$$

  Se toma sólo la etiqueta R cómo fresca.

- $V_i \leftarrow V_j + k$

$$
\begin{array}{ll}
 & V_i \leftarrow k \\
 & Z_a \leftarrow Z_a + 1 \\
 & IF\ Z_a \neq 0\ GOTO\ C \\
[S] & V_j \leftarrow V_j - 1 \\
 & V_i \leftarrow V_i + 1 \\
 & Z_a \leftarrow Z_a + 1 \\
[C] & IF\ V_j \neq 0\ GOTO\ S \\
 & IF\ Z_a \neq 0\ GOTO\ F \\
[L] & V_j \leftarrow V_j + 1 \\
[F] & Z_a \leftarrow Z_a - 1 \\
 & IF\ Z_a \neq 0\ GOTO\ L
\end{array}
$$

Se toman las etiquetas S, C, L, F y la variable Z_a cómo frescas.

- $IF\ V_i = 0\ GOTO\ L$

$$
\begin{aligned}
& IF\ V_i \neq 0\ GOTO\ C \\
& Z_a \leftarrow Z_a + 1 \\
& IF\ Z_a \neq 0\ GOTO\ L \\
[C]\ & Z_a \leftarrow Z_a + 1
\end{aligned}
$$

Se toman la etiqueta C y la variable Z_a cómo frescas.

- $GOTO\ L$

$$
\begin{aligned}
& Z_a \leftarrow Z_a + 1 \\
& IF\ Z_a \neq 0\ GOTO\ L
\end{aligned}
$$

Se toma sólo la variable Z_a cómo fresca.

1.1.2 Definir dos pseudo-programas distintos en el lenguaje $\mathcal{S}$ (usando las macros convenientes del punto anterior) que computen la función de dos variables $f(x_1, x_y) = x_1 + x_2$. Par aalguno de los dos, expandir las macros utilizadas prestando atención a la instanciación de variables y etiquetas frescas.

$$
\begin{aligned}
& Y \leftarrow X_1 + 0 \\
& GOTO\ B \\
[A]\ & Y \leftarrow Y + 1 \\
& X_2 \leftarrow X_2 - 1 \\
[B]\ & IF\ X_2 \neq 0\ GOTO\ A
\end{aligned}
$$

$$
\begin{aligned}
[A]\ & IF\ X_1 = 0\ GOTO\ B \\
& Y \leftarrow Y + 1 \\
& X_1 \leftarrow X_1 - 1 \\
& GOTO\ A \\
[B]\ & IF\ X_2 = 0\ GOTO\ E \\
& Y \leftarrow Y + 1 \\
& X_2 \leftarrow X_2 - 1 \\
& GOTO\ B
\end{aligned}
$$

Vamos a expandir la segunda de las formulaciones (por ser la que tiene macros más simples).

$$
\begin{array}{ll}
[A] & IF\ X_1 \neq 0\ GOTO\ Y \\
 & Z_1 \leftarrow Z_1 + 1 \\
 & IF\ Z_1 \neq 0\ GOTO\ B \\
[Y] & Z_1 \leftarrow Z_1 + 1 \\
 & Y \leftarrow Y + 1 \\
 & X_1 \leftarrow X_1 - 1 \\
 & Z_2 \leftarrow Z_2 + 1 \\
 & IF\ Z_2 \neq 0\ GOTO\ A \\
[B] & IF\ X_2 \neq 0\ GOTO\ Z \\
 & Z_3 \leftarrow Z_3 + 1 \\
 & IF\ Z_3 \neq 0\ GOTO\ E \\
[Z] & Z_3 \leftarrow Z_3 + 1 \\
 & Y \leftarrow Y + 1 \\
 & X_2 \leftarrow X_2 - 1 \\
 & Z_4 \leftarrow Z_4 + 1 \\
 & IF\ Z_4 \neq 0\ GOTO\ B
\end{array}
$$

1.1.3 Sea P el programa en $\mathcal{S}$ que resulta de expandir todas las macros en aguno de los códigos del punto anterior. Determinar cuál es la función computada en cada caso:

- $\Psi_P^{(1)} : \mathbb{N} \to \mathbb{N}$

  $f(x) = x$, se puede ver fácil desde planteo del ejercicio anterior, los parámetros no inicializados son ceros por lo que la función pedida se instancia como $f(x_1, 0) = x + 0$ y se transforma nuestra suma en la función identidad.

- $\Psi_P^{(2)} : \mathbb{N}^2 \to \mathbb{N}$

  $f(x, y) = x + y$, la función pedida el ejercicio anterior.

- $\Psi_P^{(3)} : \mathbb{N}^3 \to \mathbb{N}$

  $f(x, y, z) = x + y$, dado que ignoramos el tercer parámetro en ambas formulaciones del programa.

1.2 Ejercicio 2

1.2.1 Sea $\mathcal{C}_\mathcal{S} = \{\Psi_P^{(n)} \mid P \text{ es un programa en } S, n \geq 1\}$ la clase de funciones $\mathcal{S}$-parciales computables. Mostrar que $\mathcal{C}_\mathcal{S}$ es una clase PRC

Para mostrar que es PRC necesito dar un programa para las iniciales y demostrar que $\mathcal{C}_\mathcal{S}$ está cerrado por composición y recursión primitiva.

Vamos primero por las iniciales:

$$\begin{array}{lll} n(x) = 0 & s(x) = x + 1 & u_i^n(x_1, \ldots, x_n) = x_i \\ Z_1 \leftarrow Z_1 + 1 & Y \leftarrow X_1 + 1 & Y \leftarrow X_i + 0 \end{array}$$

Y ahora veamos cómo resolver la composición, tomo $h : \mathbb{N}^k \to \mathbb{N}$ y $g_1, \ldots, g_k : \mathbb{N}^n \to \mathbb{N}$ que pertenezcan a $\mathcal{C}_\mathcal{S}$ (y por lo tanto tengan programas que podamos usar cómo macros):

$$f(x_1, \ldots, x_n) = h(g_1(x_1, \ldots, x_n), \ldots, g_k(x_1, \ldots, x_n))$$

$$\begin{array}{l} Z_1 \leftarrow g_1(X_1, \ldots, X_n) \\ \qquad \vdots \\ Z_k \leftarrow g_k(X_1, \ldots, X_n) \\ Y \leftarrow h(Z_1, \ldots, Z_k) \end{array}$$

Para que esto funcione tenemos que restringir a cada macro a dejar sus variables de entrada intactas al finalizar su ejecución. Una forma de mecanizarlo es que cada macro copie todas sus variables de entrada a variables temporales “frescas” (que no se hayan usado ni se vayan a usar) y que cada macro designe como variable de salida una variable temporal “fresca”. La única excepción a esto es el macro de asignación $V_i \leftarrow Expr$ que aunque el resultado de $Expr$ esté en una variable “fresca” debe modificar V_i para cumplir con su tarea.

Dicho todo esto, vamos por la recursión primitiva, la cuál es más simple de lo que parece, tomo $h : \mathbb{N}^n \to \mathbb{N}$ y $g : \mathbb{N}^{n+2} \to \mathbb{N}$ que pertenezcan a $\mathcal{C}_\mathcal{S}$ (¡Osea que tenemos programas!):

$$
\begin{aligned}
f(x_1, \ldots, x_n, 0) &= h(x_1, \ldots, x_n) \\
f(x_1, \ldots, x_n, t+1) &= g(f(x_1, \ldots, x_n, t), x_1, \ldots, x_n, t)
\end{aligned}
$$

$$
\begin{array}{ll}
 & Y \leftarrow h(X_1, \ldots, X_n) \\
[L] & IF\ X_{n+1}\ = 0\ GOTO\ E \\
 & Z_1 \leftarrow Z_1 + 1 \\
 & X_{n+1} \leftarrow X_{n+1} - 1 \\
 & Y \leftarrow g(Y, X_1, \ldots, X_n, Z_1) \\
 & GOTO\ L
\end{array}
$$