Lenguajes Formales, Autómatas y Computabilidad

Práctica 8: Funciones computables

Versión del 31 de octubre de 2024

Ejercicio 1. Definir macros en el lenguaje $\mathcal{S}$++ para las siguientes pseudo-instrucciones:

a. $V = 0$	Asigna el valor 0 a la variable V
b. $V = V + k$	Suma k al valor de la variable V
c. $V = k$	Asigna el valor k (constante) a la variable V
d. $V_1 = V_2$	Asigna el valor de la variable V_2 a la variable V_1
e. $V_1 = V_1 + V_2$	Suma el valor de la variable V_2 al valor de la variable V_1
f. **if** $V \neq 0$ **then** $\{$ P $\}$	Si la variable V es distinta de 0, ejecuta el programa P

g. **if** $V \neq 0$ **then** $\{$
    P_1
$\}$ **else** $\{$
    P_2
$\}$
	Si la variable V es distinta de 0, ejecuta el programa P_1, y en caso contrario ejecuta el programa P_2

h. **loop**	Entra en un ciclo infinito
i. $V = \Psi_P^{(n)}(V_1, ..., V_n)$	Si el programa P termina al tomar como entrada los valores de $V_1, ..., V_n$, asigna el resultado de su ejecución a la variable V; en caso contrario, se cuelga

Ejercicio 2.

a. Dar un pseudo-programa en el lenguaje $\mathcal{S}$++ que compute la función de dos variables $f(x_1, x_2) = x_1 + x_2$. Se pueden usar las macros definidas en el ejercicio anterior (o definir nuevas).

b. Dar el programa que resulta de expandir todas las macros utilizadas en el inciso anterior. Prestar especial atención a la instanciación de variables frescas.

c. Sea P el programa obtenido en el ejercicio anterior. Determinar cuáles son las siguientes tres funciones computadas por P según la cantidad de argumentos que recibe:

$$\Psi_P^{(1)} : \mathbb{N} \to \mathbb{N}, \qquad \Psi_P^{(2)} : \mathbb{N}^2 \to \mathbb{N}, \qquad \Psi_P^{(3)} : \mathbb{N}^3 \to \mathbb{N}$$

Ejercicio 3.

a. Considerar la clase de funciones computables (por un programa de $\mathcal{S}$++):

$$\left\{ \Psi_P^{(n)} \;\middle|\; P \text{ es un programa de } \mathcal{S}\text{++}, n \geq 1 \right\}.$$

Mostrar que todas las funciones primitivas recursivas son computables.

b. Sin dar un programa en $\mathcal{S}$++, mostrar que la función $f(x, y) = xy$ es una función computable.

Ejercicio 4. Sea $r : \mathbb{N}^n \to \{0,1\}$ un predicado primitivo recursivo. Definir macros en el lenguaje $\mathcal{S}$++ para las siguientes pseudo-instrucciones:

a. **while** $r(x_1, ..., x_n)$ **do** $\{$
    P
$\}$

b. **if** $r(x_1, ..., x_n)$ **then** $\{$
    P_1
$\}$ **else** $\{$
    P_2
$\}$

Ejercicio 5. Demostrar en cada caso que el lenguaje $\mathcal{L}$ es computable. Es decir, usando la codificación de cadenas y las funciones definidas en el ejercicio 11 de la práctica 7, dar para cada lenguaje $\mathcal{L}$ un (pseudo-)programa P tal que

$$\Psi_P^{(1)}(x) = \begin{cases} 1 & \text{si la cadena codificada por } x \in \mathcal{L} \\ 0 & \text{si no.} \end{cases}$$

a. $\mathcal{L} = \{\omega \in \{a,b\}^* \mid |\omega|_a = |\omega|_b\}$.

b. $\mathcal{L} = \{a^n b^m \mid 1 \leq n \leq m\}$.

c. $\mathcal{L} = \{a^n b^n c^n \mid n \geq 1\}$.

d. $\mathcal{L} = \{\omega \in \{a,b\}^* \mid \omega = \omega^{\mathrm{r}}\}$.

e. $\mathcal{L} = \{\omega\omega \mid \omega \in \{a,b\}^*\}$.

f. $\mathcal{L} = \{a^n \mid n \text{ es primo}\}$.

Ejercicio 6. Dadas las funciones parciales $f, g : \mathbb{N} \to \mathbb{N}$ dadas por:

$$f(x) = \begin{cases} 1 & \text{si } x = 3 \\ \uparrow & \text{si no} \end{cases} \qquad g(x) = 2x$$

Demostrar que la siguiente función es parcialmente computable:

$$h(x) = \begin{cases} f(x) & \text{si } x \geq 5 \vee x = 3 \\ g(x) & \text{si no} \end{cases}$$

Ejercicio 7.

a. Sea $r : \mathbb{N}^{n+1} \to \{0,1\}$ un predicado computable. Mostrar que la siguiente función es parcialmente computable:

$$\mu_p(x_1, ..., x_n, y) = \begin{cases} \min\{t \mid y \leq t \wedge p(x_1, ..., x_n)\} & \text{si existe tal } t \\ \uparrow & \text{si no} \end{cases}$$

b. Usando el resultado anterior, demostrar que si una función $f : \mathbb{N} \to \mathbb{N}$ es biyectiva y computable, su inversa f^{-1} también es computable.

Ejercicio 8. Decimos que un programa de $\mathcal{S}$++ es *aburrido* si en todas sus instrucciones del tipo **while** $V \neq 0$ **do** $\{\ P\ \}$, la variable usada en la condición es una variable temporal (es decir, del tipo Z_i).

Demostrar que el siguiente predicado es primitivo recursivo:

$$r(x) = \begin{cases} 1 & \text{si el programa cuyo número es } x \text{ es aburrido} \\ 0 & \text{si no} \end{cases}$$

Ejercicio 9. Usando las funciones primitivas recursivas $\text{STEP}^{(n)}$ y $\text{SNAP}^{(n)} : \mathbb{N}^{n+2} \to \mathbb{N}$ vistas en clase, demostrar que las siguientes funciones son parcialmente computables:

a. $f_1(x,y) = \begin{cases} 1 & \text{si } y \in \text{Dom } \Phi_x^{(1)} \\ \uparrow & \text{si no} \end{cases}$

b. $f_2(x) = \begin{cases} 1 & \text{si Dom } \Phi_x^{(1)} \neq \varnothing \\ \uparrow & \text{si no} \end{cases}$

c. $f_3(x,y) = \begin{cases} 1 & \text{si } y \in \text{Im } \Phi_x^{(1)} \\ \uparrow & \text{si no} \end{cases}$

d. $f_4(x,y) = \begin{cases} 1 & \text{si Dom } \Phi_x^{(1)} \cap \text{Im } \Phi_x^{(1)} = \varnothing \\ \uparrow & \text{si no} \end{cases}$

Ejercicio 10. Sea $f : \mathbb{N} \to \mathbb{N}$ una función parcialmente computable en tiempo polinomial; i.e., existe un programa P con $\Psi_P^{(1)}(x) = f(x)$ tal que, para algún polinomio Q, P no requiere más de $Q(\lceil \log_2(x) \rceil)$ pasos para terminar.

a. Mostrar que f es primitiva recursiva.

b. ¿Sucede lo mismo si la cota es exponencial, doblemente exponencial, etc.?

c. ¿Qué podemos decir, en general, sobre la complejidad temporal de una función computable que no sea primitiva recursiva?