

Modelos de Computación y de Complejidad

Grado en Ingeniería Informática. Tecnologías Informáticas (Curso 2021–2022)

ETS Ingeniería Informática - Universidad de Sevilla

FUNCIONES GOTO-COMPUTABLES

Ejercicio 1.– Diseñar programas GOTO sin usar macros que calculen los siguientes predicados binarios sobre $\mathbb{N}$: $\theta_1(a, b) = 1 \implies a = b$; $\theta_2(a, b) = 1 \implies a \leq b$; $\theta_3(a, b) = 1 \implies a < b$.

Verificar formalmente los programas diseñados.

Ejercicio 2.– Diseñar programas GOTO sin usar macros que calculen los siguientes predicados sobre $\mathbb{N}$: $\theta_1(a) = 1 \iff a$ es un número par; $\theta_2(a, b) = 1 \implies a + b$ es un número par.

Verificar formalmente los programas diseñados.

Ejercicio 3.– Diseñar programas GOTO que, sin usar macros (salvo GOTO L), calculen las funciones siguientes:

$$f_1(x) = 2x ; \quad f_2(x) = \begin{cases} 1 & \text{si } x \text{ es par} \\ \uparrow & \text{si } x \text{ es impar} \end{cases} ; \quad f_3(x) = \begin{cases} 2 & \text{si } x = 0 \\ \uparrow & \text{si } x = 1 \\ x + 1 & \text{si } x \geq 2 \end{cases}$$

Comprobar, para algunos valores concretos, que los programas diseñados funcionan correctamente.

Ejercicio 4.– Diseñar un programa GOTO que calcule la función total $f : \mathbb{N} \rightarrow \mathbb{N}$ caracterizada por las condiciones siguientes:

$$\begin{cases} f(0) & = 1 \\ f(x+1) & = 3 \cdot f(x) + 1, \text{ para cada } x \in \mathbb{N} \end{cases}$$

Ejercicio 5.– Probar que si una función es GOTO-computable, entonces existen **infinitos programas** GOTO que calculan dicha función.

Ejercicio 6.– Diseñar un programa GOTO tal que contenga alguna instrucción condicional y, además, toda computación del programa realice, exactamente, 4 pasos de transición.

Ejercicio 7.– Usando el teorema de inducción, demostrar que se verifica lo siguiente: para cada número natural k , cualquier un programa GOTO de longitud k que carece de instrucciones condicionales, **para** sobre **todo** número natural y siempre devuelve un número menor o igual que k .

Ejercicio 8.– Sea M el modelo de computación cuyos procedimientos mecánicos son los programas GOTO tales que **todas** las instrucciones son **del mismo tipo**. Estudiar si son computables en M : (a) la función vacía; (b) la función $f(x) = 0$, para cada $x \in \mathbb{N}$; y (c) la función identidad en $\mathbb{N}$ ($f(x) = x$, para cada $x \in \mathbb{N}$).

Ejercicio 9.– Sea M el modelo de computación cuyos procedimientos mecánicos son los programas GOTO (distintos del programa vacío) tales que **sólo** contienen instrucciones **condicionales**. Demostrar que la función vacía **no** es computable en dicho modelo.

Ejercicio 10.— Sea P un programa GOTO.

1. ¿Qué se puede deducir si en P **no existe ninguna variable de entrada** X_j ?
2. ¿Qué se puede deducir si en P **sólo existen variables de entrada**?
3. ¿Qué se puede deducir si en P **sólo existen variables auxiliares o de trabajo**?
4. ¿Qué se puede deducir si en P **no existe la variable de salida** Y ?
5. ¿Qué se puede deducir si en P **no existe ninguna instrucción del tipo incremento**?
6. ¿Qué se puede deducir si en P **no existe ninguna instrucción condicional**?

Ejercicio 11.— Probar que es GOTO-computable la función total $f : \mathbb{N} \rightarrow \mathbb{N}$ definida por:

$$f(0) = 1 \text{ y } f(n+1) = \text{número de dígitos de } n+1 \text{ en base } 10$$

Ejercicio 12.— Probar que las funciones binarias que calculan el *máximo común divisor* y el *mínimo común múltiplo* de dos números naturales son GOTO-computables.

Ejercicio 13.— Probar que los siguientes predicados binarios son GOTO-computables.

- $\theta_1(x, y) \equiv x$ e y tienen distinta paridad y, además, tienen el mismo *número de divisores* primos.
- $\theta_2(x, y) \equiv x$ e y son distintos de cero y tienen los *mismos divisores* primos.
- $\theta_3(x, y) \equiv$ el *número de divisores* de x es, *al menos*, y .

Ejercicio 14.— Probar que las siguientes funciones totales de $\mathbb{N}$ en $\mathbb{N}$ son GOTO-computables:

- (a) $\begin{cases} f(0) = 0 \\ f(x) = \text{la suma de los divisores de } x, & \text{si } x \neq 0 \end{cases}$
- (b) $g(x) =$ el número de primos menores o iguales que x .
- (c) $h(x) =$ el único número natural n tal que $n \leq \sqrt{2} \cdot x < n+1$.

Ejercicio 15.— Probar que es GOTO-computable la función total $f : \mathbb{N} \rightarrow \mathbb{N}$ definida como sigue: $f(0) = 0$ y $f(x) =$ el número de **divisores impares** de x , para cada $x > 0$.

Ejercicio 16.— Sea $f : \mathbb{N}^{k+1} \rightarrow \mathbb{N}$ una función total $(k+1)$ -aria, con $k \geq 0$. Se define la *suma acotada* de f , que notaremos Σ_f , así:

$$\Sigma_f(\vec{x}, y) = \sum_{z \leq y} f(\vec{x}, z)$$

Probar que si f es GOTO-computable, entonces la función suma acotada Σ_f también lo es.

Nota: El *producto acotado*, Π_f , de una función total $(k+1)$ -aria f : $\Pi_f(\vec{x}, y) = \prod_{z \leq y} f(\vec{x}, z)$, y se prueba que si f es GOTO-computable, entonces el producto acotado Π_f también lo es.

Ejercicio 17.— Justificar la veracidad o falsedad de las siguientes afirmaciones.

- (a) Un programa GOTO es una sucesión finita de instrucciones básicas GOTO, etiquetadas o no.
- (b) Las computaciones del programa GOTO que carece de instrucciones (el *programa vacío*) no realiza ningún paso de transición.
- (c) El programa GOTO que carece de instrucciones (el *programa vacío*) calcula la función vacía.
- (d) Un programa GOTO con sólo tres instrucciones calcula una función de aridad 3.
- (e) Un programa GOTO con sólo cinco instrucciones calcula una función de aridad 7.
- (f) Una configuración inicial de un programa GOTO **no** puede ser una configuración de parada de dicho programa.
- (g) La primera componente de cualquier configuración inicial de un programa GOTO **siempre** es igual a 1.
- (h) La función vacía se puede calcular con un programa GOTO que posea dos instrucciones.
- (i) La función idénticamente nula de aridad 1 ($f(x) = 0$, para cada $x \in \mathbb{N}$) se puede calcular con un programa GOTO que posea exactamente **tres** instrucciones.
- (j) La función vacía y la función idénticamente nula de aridad 1 se pueden calcular mediante programas GOTO que contengan, al menos, una instrucción del tipo **incremento** y, al menos, una instrucción del tipo **condicional**.
- (k) La función vacía y la función identidad pueden ser calculadas por programas GOTO que **no** usen la instrucción decremento.
- (l) La función vacía, la función idénticamente nula de aridad 1 y la función identidad en $\mathbb{N}$ se pueden calcular mediante programas GOTO que contengan únicamente instrucciones del tipo **incremento** y del tipo **condicional**.
- (m) La función vacía se puede calcular con un programa GOTO que posea exactamente **cuatro** instrucciones.
- (n) Usando programas GOTO que constan **sólo de instrucciones condicionales**, se pueden calcular la función idénticamente nula y la función vacía.
- (o) Usando programas GOTO tales que **todas** sus instrucciones sean del **mismo tipo**, se puede calcular la función idénticamente nula pero, en cambio, **no** se puede calcular función vacía.
- (p) La función **identidad** en $\mathbb{N}$, la función **idénticamente nula** de aridad 1 y la función **vacía**, se pueden calcular mediante programas GOTO tales que todas sus instrucciones tienen la **misma variable**.
- (q) La función vacía se puede calcular con un programa GOTO que posea una única instrucción.
- (r) Cualquier programa GOTO que **no** contenga la variable X , calcula la función total $C_0^{(1)}$ de $\mathbb{N}$ en $\mathbb{N}$ definida así: $C_0^{(1)}(x) = 0$, para cada $x \in \mathbb{N}$.
- (s) Toda función definida por composición a partir de funciones GOTO-computables es, también, una función GOTO-computable.

Ejercicio 18.— Sea GOTO_l el modelo de computación cuyos procedimientos mecánicos son los programas GOTO que **no** contienen ninguna instrucción condicional (tales programas GOTO se denominan **programas lineales**). Probar que:

1. Para cada $k \in \mathbb{N}$ se verifica:
 - La función constante de aridad 1 $C_k^{(1)}(x) = k$, para cada $x \in \mathbb{N}$, es GOTO_l -computable.
 - Si P es un programa GOTO_l de longitud k , entonces $\llbracket P \rrbracket^{(1)}(x) \leq k$.
 - Si P es un programa lineal que calcula la función constante C_k , entonces $|P| \geq k$.
2. La función $f(x) = x + 1$ **no** es GOTO_l -computable.
3. Las funciones GOTO_l -computables 1-arias son *exactamente* las funciones constantes.
4. El modelo de computación GOTO es estrictamente más potente (“calcula” más funciones) que el modelo de computación GOTO_l .

Ejercicio 19.— Sea GOTO_f el modelo de computación cuyos procedimientos mecánicos son los programas GOTO en los que toda instrucción condicional de dicho programa debe ser del tipo:

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

siendo L una etiqueta que **sólo puede aparecer etiquetada a la izquierda** en una instrucción del programa *posterior* a I . Probar que:

1. Toda función GOTO_f -computable es total.
2. Los modelos de computación GOTO_f y GOTO_l no son equivalentes; es decir, probar que existe una función computable en el modelo GOTO_f que **no** lo es en el modelo GOTO_l .
3. Toda función GOTO_f -computable de aridad 1 tiene rango finito.
4. El modelo de computación GOTO es estrictamente más potente (“calcula” más funciones) que el modelo de computación GOTO_f .

Ejercicio 20.— Sea GOTO_1 el modelo de computación cuyos procedimientos mecánicos son los programas GOTO en los que sólo aparece una variable.

1. Probar que la función $f(x) = 2 \cdot x$ no es GOTO_1 -computable.
2. Probar que el modelo de computación GOTO es estrictamente más potente (“calcula” más funciones) que el modelo de computación GOTO_1 .
3. Los modelos GOTO_l y GOTO_1 ¿son equivalentes? (es decir, ¿ambos modelos calculan exactamente las mismas funciones?). Razónese la respuesta.
4. Sea $\text{GOTO}_{1,f}$ el modelo de computación cuyos procedimientos mecánicos son los programas de GOTO_1 cuyas instrucciones condicionales deben cumplir la restricción de GOTO_f . Los modelos GOTO_l y $\text{GOTO}_{1,f}$ ¿son equivalentes? Razónese la respuesta.