Tema 2
La Jerarquía de Funciones

Alberto García González

Modelos de Computación - Universidad de Cádiz

Contenidos

Composición Funcional

Aproximación inicial

Dadas las funciones $f(x)$ y $g(x)$, su composición es

$$(g \circ f)(x) = g(f(x)).$$

Definición

Sean $f : \mathbb{N}^k \to \mathbb{N}$ y $g : \mathbb{N}^n \to \mathbb{N}$ funciones de k y n variables. La composición de $f, g_1, g_2, ..., g_k$ se define como sigue:

$$h(x_1, x_2, ..., x_n) = f(g_1(x_1, x_2, ..., x_n), ..., g_k(x_1, x_2, ..., x_n)).$$

Composición Funcional

Teorema (de Composición)

Si $f, g_1, g_2, \ldots, g_k$ *son totalmente computables,* $h(x_1, x_2, \ldots, x_n)$ *es totalmente computable.*

Corolario

Si alguna de las funciones $f, g_1, g_2, \ldots, g_k$ *es parcialmente computable, entonces* h *también lo es.*

Composición Funcional

Ejemplo

Sean

- $f(x_1, x_2, x_3) = (x_1 + x_2) \cdot x_3$,
- $g_1(x_1, x_2) = x_1 - x_2$,
- $g_2(x_1, x_2) = x_1 \cdot x_2$,
- $g_3(x_1, x_2) = x_1 + x_2$,

entonces

$$h(x_1, x_2) = ((x_1 - x_2) + x_1 \cdot x_2) \cdot (x_1 + x_2).$$

Recursión Primitiva

Aproximación

$$fac : \mathbb{N} \to \mathbb{N},$$
$$fac(0) = 1,$$
$$fac(n+1) = (n+1) \cdot fac(n),$$

donde:

- $fac(0) = 1$ es la forma de salir de la recursión.
- $fac(n+1) = (n+1) \cdot fac(n)$ es el caso general.
- $\cdot$ (el producto) es una función auxiliar.

Recursión Primitiva

Definición

Sea $g : \mathbb{N}^2 \to \mathbb{N}$ una función total y $k \in \mathbb{N}$, la función $h : \mathbb{N} \to \mathbb{N}$ está definida por recursión primitiva a partir de g si:

$$h(0) = k,$$
$$h(t+1) = g(t, h(t)).$$

Teorema (de recursión)

Si h se define a partir de g por recursión primitiva y g es computable, entonces h también lo es.

Recursión Primitiva

Definición

Sean $f : \mathbb{N}^n \to \mathbb{N}$ y $g : \mathbb{N}^{n+2} \to \mathbb{N}$ funciones totales, se define $h : \mathbb{N}^{n+1} \to \mathbb{N}$ por recursión primitiva generalizada:

$$h(x_1, x_2, ..., x_n, 0) = f(x_1, x_2, ..., x_n),$$
$$h(x_1, x_2, ..., x_n, t+1) = g(t, h(x_1, x_2, ..., x_n, t), x_1, x_2, ..., x_n).$$

Clases Cerradas por Recursión Primitiva (Clases PRC)

Definición (Funciones Iniciales)

Las funciones iniciales son:

1. La función nula:
$$n : \mathbb{N} \to \mathbb{N} \text{ dada por } n(x) = 0, \forall x \in \mathbb{N}.$$
2. La función sucesor:
$$s : \mathbb{N} \to \mathbb{N} \text{ dada por } s(x) = x + 1, \forall x \in \mathbb{N}.$$
3. Las funciones de proyección:
$$u_i^n : \mathbb{N} \to \mathbb{N}, \text{ dada por } u_i^n(x_1, x_2, ..., x_n) = x_i, \text{ con } n \in \mathbb{N} \text{ y } 1 \leqslant i \leqslant n.$$

Definición (Clase IMC)

$$IMC = \{f : f \text{ es inicial }\}.$$

Clases Cerradas por Recursión Primitiva (Clases PRC)

Definición (Clase PRC)

Una clase de funciones $\mathbb{C}$ es una clase *PRC* si:

1. Las funciones de $\mathbb{C}$ son totales.
2. $IMC \subseteq \mathbb{C}$.
3. $\mathbb{C}$ es cerrada bajo composición y recursión primitiva simple y generalizada.

Puede comprobarse que...

Clase de Funciones	Es *PRC*
INIC	No
COMP	Si
P_COMP	No
REC	Si

Clases Cerradas por Recursión Primitiva (Clases PRC)

Teorema

Las funciones computables (COMP) son una clase PRC.

Clases Cerradas por Recursión Primitiva (Clases PRC)

Definición (Función Recursiva Primitiva)

Una función $f : \mathbb{N}^k \to \mathbb{N}$ es recursiva primitiva si se puede obtener a partir de funciones iniciales, combinadas mediante un número finito de operaciones de composición y/o recursión primitiva.

Definición (REC)

$$REC = \{f : f \text{ es recursiva primitiva }\}.$$

Corolario

$$REC \subseteq COMP.$$

Proposición

$$COMP \nsubseteq REC.$$

Clases Cerradas por Recursión Primitiva (Clases PRC)

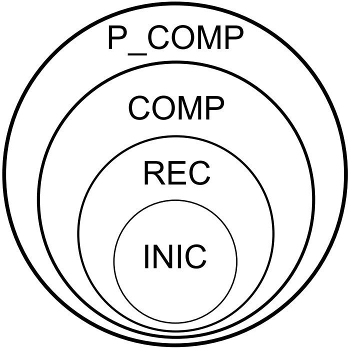

Figura: Esquema de clases de funciones.

Clases Cerradas por Recursión Primitiva (Clases PRC)

Teorema (de Caracterización)

$$REC = \bigcap_{\mathbb{C} \ \mathit{es} \ PRC} \mathbb{C}.$$

Teorema (de Jerarquía)

$$INIC \subseteq REC \subseteq COMP \subseteq P\_COMP.$$

Algunas Funciones Recursivas Primitivas

Función Suma

$$f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}, \qquad f(x,y) = x + y,$$

$$s(u_2^3(t, f(x,t), x)).$$

Pasos:

1. Reescribir la función de forma recursiva:

$$f(x,0) = x,$$

$$f(x, y+1) = f(x,y) + 1.$$

2. Expresar la función reescrita en el primer paso mediante funciones iniciales y composición y recursión primitiva:

$$f(x,0) = u_1^2(x_1, x_2) = u_1^1(x),$$

$$f(x, y+1) = s(u_2^3(y, f(x,y), x)).$$

Algunas Funciones Recursivas Primitivas

Función Producto

$$I : \mathbb{N} \times \mathbb{N} \to \mathbb{N}, \qquad I(x, y) = x \cdot y.$$

Pasos:

1. Reescribir la función de forma recursiva:

$$I(x, 0) = 0,$$

$$I(x, y + 1) = I(x, y) + x.$$

2. Expresar la función reescrita en el primer paso mediante funciones iniciales y composición y recursión primitiva:

$$I(x, 0) = n(x),$$

$$I(x, y + 1) = g(y, I(x, y), x),$$

siendo

$$g(x_1, x_2, x_3) = u_2^3(x_1, x_2, x_3) + u_3^3(x_1, x_2, x_3).$$

Función Factorial

$$h : \mathbb{N} \to \mathbb{N},$$

$$h(0) = 1, \qquad h(n+1) = (n+1) \cdot h(n).$$

Pasos:

1. Reescribir la función de forma recursiva. (Es innecesario dado que la función ya está escrita de forma recursiva.)
2. Expresar la función reescrita en el primer paso mediante funciones iniciales y composición y recursión primitiva:

$$h(0) = s(n(x)),$$

$$h(n+1) = g(n, h(n)),$$

siendo

$$g(x_1, x_2) = s(u_1^2(x_1, x_2)) \cdot u_2^2(x_1, x_2).$$

Algunas Funciones Recursivas Primitivas

Función Potencia

$$m : \mathbb{N} \times \mathbb{N} \to \mathbb{N}, \qquad m(x, y) = x^y.$$

Pasos:

1. Reescribir la función de forma recursiva:

$$m(x, 0) = 1,$$

$$m(x, y + 1) = m(x, y) \cdot x.$$

2. Expresar la función reescrita en el primer paso mediante funciones iniciales y composición y recursión primitiva:

$$m(x, 0) = s(n(x)),$$

$$m(x, y + 1) = g(y, m(x, y), x),$$

siendo

$$g(x_1, x_2, x_3) = u_2^3(x_1, x_2, x_3) \cdot u_3^3(x_1, x_2, x_3).$$

Algunas Funciones Recursivas Primitivas

Función Predecesor

$$p : \mathbb{N} \to \mathbb{N}, \qquad p(x) = \begin{cases} x - 1 & \text{si } x > 0 \\ 0 & \text{en otro caso.} \end{cases}$$

Pasos:

1. Reescribir la función de forma recursiva:

$$p(0) = 0, \qquad p(t+1) = t.$$

2. Expresar la función reescrita en el primer paso mediante funciones iniciales y composición y recursión primitiva:

$$p(0) = n(x),$$

$$p(t+1) = g(t, p(t)),$$

siendo

$$g(x_1, x_2) = u_1^2(x_1, x_2).$$

Algunas Funciones Recursivas Primitivas

Función Resta Restringida

$$p : \mathbb{N} \times \mathbb{N} \to \mathbb{N}, \qquad x \dot{-} y = \begin{cases} x - y & \text{si } x \geqslant y \\ 0 & \text{si } x < y. \end{cases}$$

Pasos:

1. Reescribir la función de forma recursiva:

$$x \dot{-} 0 = x, \qquad x \dot{-} (t+1) = p(x \dot{-} t).$$

2. Expresar la función reescrita en el primer paso mediante funciones iniciales y composición y recursión primitiva:

$$x \dot{-} 0 = u_1^1(x),$$

$$x \dot{-} (t+1) = g(t, x \dot{-} t),$$

siendo

$$g(x_1, x_2) = p(u_2^2(x_1, x_2)).$$

Algunas Funciones Recursivas Primitivas

Función Valor Absoluto

$$abs : \mathbb{N} \times \mathbb{N} \to \mathbb{N},$$

$$|x - y| = (x \dot{-} y) + (y \dot{-} x).$$

Función Negación

$$\alpha : \mathbb{N} \to \mathbb{N},$$

$$\alpha(x) = \begin{cases} 1 & \text{si } x = 0 \\ 0 & \text{en otro caso,} \end{cases}$$

$$\alpha(x) = 1 \dot{-} x.$$

Predicados Recursivos Primitivos

Predicado de Igualdad

$$d : \mathbb{N} \times \mathbb{N} \to \mathbb{N},$$

$$d(x, y) = \begin{cases} 1 & \text{si } x = y \\ 0 & \text{en otro caso} \end{cases} = \alpha(|x - y|).$$

Predicados Recursivos Primitivos

Proposición

Sean P y Q predicados tomados en REC. Entonces $P \wedge Q$, $P \vee Q$ y $\neg P$ también pertenecen a REC.

Predicado Menor o Igual

$$des : \mathbb{N} \times \mathbb{N} \to \mathbb{N},$$

$$des(x, y) = \begin{cases} 1 & \text{si } x \leqslant y \\ 0 & \text{en otro caso} \end{cases} = \alpha(x \dot{-} y).$$

Lema

El predicado $x < y \in REC$.

Predicados Recursivos Primitivos

Teorema (de Definición por Casos)

Sea $\mathbb{C}$ una clase PRC y sean $f(x_1, x_2, \ldots, x_n)$ y $g(x_1, x_2, \ldots, x_n)$ funciones de $\mathbb{C}$, junto con $P(x_1, x_2, \ldots, x_n)$ un predicado de $\mathbb{C}$. Entonces la función siguiente también pertenece a $\mathbb{C}$:

$$h(x_1, x_2, \ldots, x_n) = \begin{cases} f(x_1, x_2, \ldots, x_n) & \textit{si } P(x_1, x_2, \ldots, x_n) \\ g(x_1, x_2, \ldots, x_n) & \textit{en otro caso.} \end{cases}$$

Operaciones Iteradas y Cuantificadores Acotados

Teorema

Sea $\mathbb{C}$ una clase PRC y $f(t, x_1, x_2, ..., x_n) \in \mathbb{C}$. Entonces las funciones siguientes también pertenecen a $\mathbb{C}$:

$$g(y, x_1, x_2, ..., x_n) = \sum_{t=0}^{y} f(t, x_1, x_2, ..., x_n),$$

$$h(y, x_1, x_2, ..., x_n) = \prod_{t=0}^{y} f(t, x_1, x_2, ..., x_n).$$

Operaciones Iteradas y Cuantificadores Acotados

Definición (Cuantificación Acotada)

Sea $\mathbb{C}$ una clase *PRC* y sea $P(t, x_1, x_2, ..., x_n)$ un predicado de $\mathbb{C}$. Se definen:

- Cuantificación acotada universal:

$$(\forall t)_{\leqslant y} P(t, x_1, x_2, ..., x_n).$$

- Cuantificación acotada existencial:

$$(\exists t)_{\leqslant y} P(t, x_1, x_2, ..., x_n).$$

Proposición

Si $P(t, x_1, x_2, ..., x_n) \in \mathbb{C}$*, entonces los predicados* $(\forall t)_{\leqslant y} P(t, x_1, x_2, ..., x_n)$ *y* $(\exists t)_{\leqslant y} P(t, x_1, x_2, ..., x_n)$ *también pertenecen a* $\mathbb{C}$*.*

Operaciones Iteradas y Cuantificadores Acotados

Ejemplo

$$y|x \Leftrightarrow \text{"}x \text{ tiene a } y \text{ por divisor."}$$

El predicado $y|x \in REC$ y se puede construir de la siguiente forma:

$$y|x \Leftrightarrow (\exists t)_{\leqslant y}(x = y \cdot t).$$

Ejemplo

$$Primo(x) \Leftrightarrow \neg((\exists t)_{\leqslant x-1}(t \neq 1 \wedge t|x)).$$

Minimización

Definición

Sea $\mathbb{C}$ una clase *PRC* y sea $P(t, x_1, x_2, \ldots, x_n) \in \mathbb{C}$. Sea ahora

$$g(y, x_1, x_2, \ldots, x_n) = \sum_{u=0}^{y} \prod_{t=0}^{u} \alpha(P(t, x_1, x_2, \ldots, x_n)).$$

Lema

La función $g(y, x_1, x_2, \ldots, x_n)$ está en $\mathbb{C}$.

Minimización

Análisis de $g(y, x_1, x_2, ..., x_n)$

- Sea $t_0 \in [0, y]$ tal que $P(t, x_1, x_2, ..., x_n) = 0$ para $t < t_0$ y tal que $P(t_0, x_1, x_2, ..., x_n) = 1$. Entonces

$$\prod_{t=0}^{u} \alpha(P(t, x_1, x_2, ..., x_n)) = \begin{cases} 1 & \text{si } u < t_0 \\ 0 & \text{si } u \geqslant t_0. \end{cases}$$

- Por lo tanto,

$$g(y, x_1, x_2, ..., x_n) = \sum_{u<t_0} 1 = t_0,$$

es decir, $g(y, x_1, x_2, ..., x_n)$ es el mínimo valor de t para el que $P(t, x_1, x_2, ..., x_n)$ es verdadero.

Minimización

Definición (Función de Mínimo Acotado)

Sea $\mathbb{C}$ una clase *PRC* y $P(t, x_1, x_2, ..., x_n) \in \mathbb{C}$ un predicado de $\mathbb{C}$. Se define:

$$\min_{t \leqslant y} P(t, x_1, x_2, ..., x_n) = \begin{cases} g(t, x_1, x_2, ..., x_n) & \text{si } (\exists t)_{\leqslant y} P(t, x_1, x_2, ..., x_n) \\ 0 & \text{en otro caso.} \end{cases}$$

Lema

Si $\mathbb{C}$ *es una clase PRC y* $P(t, x_1, x_2, ..., x_n) \in \mathbb{C}$, *entonces*

$$\min_{t \leqslant y} P(t, x_1, x_2, ..., x_n) \in \mathbb{C}.$$

Minimización

Ejemplo

$\lfloor x/y \rfloor$: la parte entera de dividir a x por y.

$$\lfloor x/y \rfloor \Leftrightarrow \min_{t \leqslant x}[(t+1) \cdot y > x].$$

Ejemplo

$R(x,y)$: el resto de dividir a x por y.

$$R(x,y) \Leftrightarrow x \dot{-} (y \cdot \lfloor x/y \rfloor).$$

Ejemplo

p_n: el n-ésimo primo de la lista ordenada de primos.

$$p_0 = 0, \qquad p_{n+1} = \min_{t \leqslant p_n!+1}[Primo(t) \wedge t > p_n].$$

Minimización

Definición (Minimización No Acotada)

Dado un predicado $P(x_1, x_2, ..., x_n, y) \in \mathbb{C}$, siendo $\mathbb{C}$ una clase *PRC*, se define

$$\min_y P(x_1, x_2, ..., x_n, y).$$

Proposición

La función $\min_y P(x_1, x_2, ..., x_n, y)$ *es parcialmente computable.*

Codificación Numérica

Definición (Función de Emparejamiento)

$$\langle\ \rangle : \mathbb{N} \times \mathbb{N} \to \mathbb{N},$$

$$\langle x, y \rangle : 2^x(2y + 1) \dot{-} 1.$$

Lema

$\langle x, y \rangle$ *es una biyección.*

Ejemplo

$$\langle 0, 0 \rangle = 2^0(2 \cdot 0 + 1) \dot{-} 1 = 1 \cdot 1 \dot{-} 1 = 0.$$

$$\langle 1, 1 \rangle = 2^1(2 \cdot 1 + 1) \dot{-} 1 = 2 \cdot 3 \dot{-} 1 = 5.$$

Codificación Numérica

Definición (Decodificación)

Sea $z \in \mathbb{N}$, para obtener el par $(x, y) \in \mathbb{N} \times \mathbb{N}$ tal que $\langle x, y \rangle = z$:

1. x es el mayor número tal que $2^x | (z+1)$.
2. y es solución a la ecuación $2y + 1 = (z+1)/2^x$.

Definición (Funciones de Decodificación)

Dado $z \in \mathbb{N}$, se definen las funciones de decodificación:

$$l(z) = \min_{x \leqslant z} [(\exists y)_{\leqslant z} \langle x, y \rangle = z],$$

$$r(z) = \min_{y \leqslant z} [(\exists x)_{\leqslant z} \langle x, y \rangle = z].$$

Codificación Numérica

Proposición

Las funciones de emparejamiento $\langle x, y\rangle$ y decodificación $l(z)$ y $r(z)$ verifican:

1. $l(\langle x, y\rangle) = x, r(\langle x, y\rangle) = y.$
2. $\langle l(z), r(z)\rangle = z.$
3. $l(z), r(z) \leqslant z.$
4. $l(z), r(z), \langle x, y\rangle \in REC.$

Codificación Numérica

Definición (Números de Gödel)

Sea $(a_1, a_2, ..., a_n)$ una sucesión finita de números naturales. Se define su número de Gödel como sigue:

$$[a_1, a_2, ..., a_n] = \prod_{i=1}^{n} (p_i)^{a_i}.$$

Ejemplo (Codificación)

$$[2, 2, 6, 3] = 2^2 \cdot 3^2 \cdot 5^6 \cdot 7^3.$$

Ejemplo (Decodificación)

$$162 = 2 \cdot 3^4 = [1, 4].$$

Codificación Numérica

Teorema

Si $[a_1, a_2, ..., a_n] = [b_1, b_2, ..., b_n]$, *entonces* $a_i = b_i$ *para* $i = 1, 2, ..., n$.

Definición (Función Proyectora)

Dado un $x \in \mathbb{N}$, $(x)_i = a_i$.

Definición (Función de Longitud)

Dado un $x \in \mathbb{N}$, $Lt(x) = n$ tal que $\exists(a_1, a_2, ..., a_n)$ que verifica que $[a_1, a_2, ..., a_n] = x$.

Codificación Numérica

Lema

Las funciones proyectoras $(x)_i, Lt(x) \in REC$.

Teorema (de la Secuencia de Números)

1. $([a_1, a_2, \ldots, a_n])_i = \begin{cases} a_i & \text{si } 1 \leqslant i \leqslant n \\ 0 & \text{en otro caso.} \end{cases}$
2. $[(x)_1, (x)_2, \ldots, (x)_n] = x$.

Teoremas de las Formas Normales

Primer Teorema de la Forma Normal

Sea $f : \mathbb{N}^k \to \mathbb{N}$ *una función parcialmente computable, entonces existe un predicado* $R(x_1, x_2, ..., x_k, z) \in REC$ *tal que*

$$f(x_1, x_2, ..., x_k) = l(\min_z R(x_1, x_2, ..., x_k, z)).$$

Segundo Teorema de la Forma Normal

Cuando la función $\min_z$ *está siempre definida, se tiene a la función minimización propia (menos restrictiva que la minimización acotada). En esas condiciones, toda función computable se obtiene como una combinación de funciones iniciales, junto con un conjunto de operaciones de composición, recursión primitiva y minimización propia.*