Capítulo IV:
FUNCIONES RECURSIVAS

IV.2: FUNCIONES PRIMITIVAS RECURSIVAS

Mario de J. Pérez Jiménez
Grupo de investigación en Computación Natural
Dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla

Lógica Matemática

Curso 2010-11

Procedimientos de definición de funciones

Composición de funciones

Sean las funciones $g : \mathbb{N}^k - \longrightarrow \mathbb{N}$, y $h_1, .., h_k : \mathbb{N}^p - \longrightarrow \mathbb{N}$.

Diremos que $f : \mathbb{N}^p - \longrightarrow \mathbb{N}$ es la composición de g y $h_1, \ldots, h_k$, si para cada $\vec{x} = (x_1, \ldots, x_p) \in \mathbb{N}^p$ se verifica: $f(\vec{x}) = g(h_1(\vec{x}), \ldots, h_k(\vec{x}))$.

Notaremos $f = \mathcal{C}(g; h_1, \ldots, h_k)$.

(a) La composición de funciones totales es una función total.

(b) La composición de funciones GOTO–computables es una función GOTO–computable. Es decir: GCOMP es cerrado bajo composición:

- El siguiente programa GOTO calcula $f = \mathcal{C}(g; h_1, ..., h_k)$:

$$
\begin{array}{rl}
Z_1 & \longleftarrow h_1(X_1, ..., X_p) \\
 & \vdots \\
Z_k & \longleftarrow h_k(X_1, ..., X_p) \\
Y & \longleftarrow g(Z_1, ..., Z_k)
\end{array}
$$

Recursión Primitiva (I):

Sean $g : \mathbb{N}^k \longrightarrow \mathbb{N}$, $h : \mathbb{N}^{k+2} \longrightarrow \mathbb{N}$.

Diremos que $f : \mathbb{N}^{k+1} \longrightarrow \mathbb{N}$ está definida por recursión primitiva a partir de g y h si para cada $\vec{x} \in \mathbb{N}^k$ y cada $y \in \mathbb{N}$ se verifica:

$$\begin{cases} f(\vec{x}, 0) = g(\vec{x}) \\ f(\vec{x}, y+1) = h(\vec{x}, y, f(\vec{x}, y)) \end{cases}$$

Notaremos $f = \mathcal{R}(g, h)$.

- Esta definición se extiende al caso $k = 0$:

$$\begin{cases} f(0) = c \\ f(x+1) = h(x, f(x)) \end{cases}$$

  donde $c \in \mathbb{N}$. En este caso escribimos: $f = \mathcal{R}(c, h)$.

Recursión Primitiva (II):

(a) Toda función definida por recursión primitiva a partir de funciones totales es una función total.

(b) Toda función definida por recursión primitiva a partir de funciones GOTO–computables es GOTO–computable. Es decir, GCOMP es cerrado bajo recursión primitiva.

- Sean $g : \mathbb{N}^k \dashrightarrow \mathbb{N}$, $h : \mathbb{N}^{k+2} \dashrightarrow \mathbb{N}$ funciones GOTO–computables. El siguiente programa calcula $f = \mathcal{R}(g; h)$:

$$
\begin{array}{ll}
 & Y \longleftarrow g(X_1, ..., X_k) \\
[A] & IF\ X_{k+1} = 0\ GOTO\ E \\
 & Y \longleftarrow h(X_1, ..., X_k, Z, Y) \\
 & Z \longleftarrow Z + 1 \\
 & X_{k+1} \longleftarrow X_{k+1} - 1 \\
 & GOTO\ A
\end{array}
$$

Funciones básicas

Llamaremos *funciones básicas* a las siguientes funciones:

- *Siguiente*: $\mathcal{S}(x) = x + 1$, para cada $x \in \mathbb{N}$.
- *Idénticamente nula de aridad 1*: $\mathcal{O}(x) = 0$, para cada $x \in \mathbb{N}$.
- *Proyecciones*: $\prod_j^k(x_1, \ldots, x_k) = x_j$, para cada $j, k \in \mathbb{N}$ $(1 \leq j \leq k)$ y cada $(x_1, \ldots, x_k) \in \mathbb{N}$.

Todas las funciones básicas son GOTO–computables:

- Un programa que calcula $\mathcal{S}$ es $\begin{cases} X \longleftarrow X + 1 \\ Y \longleftarrow X \end{cases}$
- Un programa que calcula $\mathcal{O}$ es el Programa vacío, $P_\emptyset$.
- Un programa que calcula la proyección j–ésima de aridad k, $\prod_j^{(k)}$, es $Y \longleftarrow X_j$.

La clase de las Funciones Primitivas Recursivas

Es la menor clase **PR** de funciones que verifican las condiciones siguientes:

- Toda función básica pertenece a **PR**.
- **PR** es cerrada bajo los procedimientos de composición y de recursión primitiva.
- Si $\mathcal{C}$ es una clase de funciones que:
  - contiene a todas las funciones básicas; y
  - es cerrada bajo los procedimientos de composición y de recursión primitiva,

  entonces **PR** $\subseteq \mathcal{C}$.

Obviamente, toda función primitiva recursiva es total.

Inducción sobre la clase **PR**

Cómo probar que toda función **PR** satisface una propiedad θ:

- Se considera la clase $\mathcal{D}$ de todas las funciones que verifican θ.
- Se prueba que $\mathcal{D}$ contiene a todas las funciones básicas.
- Se prueba que $\mathcal{D}$ es cerrada bajo composición y bajo recursión primitiva.

Entonces se concluye que **PR** $\subseteq \mathcal{D}$.

- Toda función **PR** es `GOTO`–computable.
- Existen funciones `GOTO`–computables que no son **PR** (incluso funciones `GOTO`–computables y totales: función de Ackermann).

Funciones primitivas recursivas relevantes (I)

Las siguientes funciones son primitivas recursivas:

- La función identidad en $\mathbb{N}$, $I_{\mathbb{N}} : \mathbb{N} \longrightarrow \mathbb{N}$, definida por $I_{\mathbb{N}}(x) = x$, para cada $x \in \mathbb{N}$.
- Las funciones constantes de aridad k, $C_a^k : \mathbb{N}^k \longrightarrow \mathbb{N}$, definida por $C_a^k(x_1, \ldots, x_k) = a$, para cada $a, k \in \mathbb{N}$ y $k \geq 1$.
- La función predecesor, $pr : \mathbb{N} \longrightarrow \mathbb{N}$, definida por

$$pr(x) = \begin{cases} 0 & si\ x = 0 \\ x - 1 & si\ x > 0 \end{cases}$$

- La función diferencia reducida, $\dot{-} : \mathbb{N}^2 \longrightarrow \mathbb{N}$, definida por

$$x \dot{-} y = \begin{cases} 0 & si\ x \leq y \\ x - y & e.c.o.c. \end{cases}$$

- La función signo: $sg : \mathbb{N} \longrightarrow \mathbb{N}$, definida por

$$sg(x) = \begin{cases} 0 & si\ x = 0 \\ 1 & e.c.o.c. \end{cases}$$

Funciones primitivas recursivas relevantes (II)

- La función signo inverso, $\bar{sg}\colon \mathbb{N} \longrightarrow \mathbb{N}$, definida por

$$\bar{sg}\ (x) = 1 \stackrel{\bullet}{-} sg(x)$$

- La función suma, $+\ : \mathbb{N}^2 \longrightarrow \mathbb{N}$, definida por

$$+(x, y) = x + y$$

- La función producto, $\cdot\ : \mathbb{N}^2 \longrightarrow \mathbb{N}$, definida por

$$\cdot(x, y) = x \cdot y$$

- La función mínimo, $min : \mathbb{N}^2 \longrightarrow \mathbb{N}$, definida por

$$min(x, y) = \begin{cases} x & si\ x \leq y \\ y & e.c.o.c. \end{cases}$$

- La función máximo, $max : \mathbb{N}^2 \longrightarrow \mathbb{N}$, definida por

$$max(x, y) = \begin{cases} x & si\ x \geq y \\ y & e.c.o.c. \end{cases}$$

Funciones primitivas recursivas relevantes (III)

- La función distancia, $|\ \ | : \mathbb{N}^2 \longrightarrow \mathbb{N}$, definida por

$$|x - y| = \begin{cases} x - y & si\ x \geq y \\ y - x & e.c.o.c. \end{cases}$$

- La función resto, $rm\ : \mathbb{N}^2 \longrightarrow \mathbb{N}$, definida por $rm(x, y) = z \Longrightarrow x = q \cdot y + z$, en donde $0 \leq z < y$.
- La función cociente, $qt\ : \mathbb{N}^2 \longrightarrow \mathbb{N}$, definida por $qt(x, y) = z \Longrightarrow x = z \cdot y + rm(x, y)$.
- La función exponencial, $exp\ : \mathbb{N}^2 \longrightarrow \mathbb{N}$, definida por

$$exp(x, y) = \begin{cases} 1 & si\ x = 0 \wedge y = 0 \\ x^y & e.c.o.c. \end{cases}$$

- La función factorial, $fact\ : \mathbb{N} \longrightarrow \mathbb{N}$, definida por

$$fact(x) = \begin{cases} 1 & si\ x = 0 \\ \prod\limits_{1 \leq j \leq x} j & e.c.o.c. \end{cases}$$

- **Proposición**: Sean $k \geq 2, n \geq 1$. Si $f_1, f_2, ..., f_k \in \mathbf{PR}^{(n)}$ entonces $f_1 + f_2 + ... + f_k \in \mathbf{PR}^{(n)}$ y $f_1 \cdot f_2 \cdot ... \cdot f_k \in \mathbf{PR}^{(n)}$

Predicados y conjuntos primitivos recursivos

Definición: La función característica de $B \subseteq A^k$ es el predicado C_B sobre A^n:

$$C_B(\vec{x}) = \begin{cases} 1 & si\ \vec{x} \in B \\ 0 & si\ \vec{x} \notin B \end{cases}$$

- La función característica de $B \subseteq A^n$, identifica el conjunto B con un predicado (precisamente, C_B).
- Si θ es un predicado n–ario sobre A y $B = S_\theta$ entonces $C_B = \theta$.

Definición: Un predicado sobre $\mathbb{N}$ es primitivo recursivosi la función que lo define es primitiva recursiva.

Definición: Un conjunto $B \subseteq \mathbb{N}^k$ es primitivo recursivo si la función C_B es primitivo recursivo. Notaremos $B \in \Delta^0_*$.

Ejemplos:

- Los conjuntos $\emptyset$ y $\mathbb{N}^k$, para cada $k \in \mathbb{N}$, son primitivos recursivos.
- Los predicados: $\theta_1(x, y) \equiv x = y$; $\theta_2(x, y) \equiv x \leq y$; $\theta_3(x, y) \equiv x < y$ son primitivos recursivos.
- Si $f : \mathbb{N}^k \longrightarrow \mathbb{N}$ es una función primitiva recursiva, entonces los siguientes predicados $(k+1)$-arios son primitivos recursivos:

$$\theta(\vec{x}, y) \equiv f(\vec{x}) = y; \quad \theta'(\vec{x}, y) \equiv f(\vec{x}) \leq y; \quad \theta''(\vec{x}, y) \equiv f(\vec{x}) < y$$

Teorma del grafo: Sea $f : \mathbb{N}^{k+1} \to \mathbb{N}$ una función **PR**. El grafo de f, $G(f) = \{(\vec{x}, y) \in \mathbb{N}^{k+1} : \ f(\vec{x}) = y\}$, es un conjunto primitivo recursivo.

Proposición. Sean θ, θ' predicados **PR** sobre $\mathbb{N}$, de aridad k. Entonces son **PR** los predicados: $\neg\theta$, $\theta \wedge \theta'$, $\theta \vee \theta'$, $\theta \Rightarrow \theta'$ y $\theta \Leftrightarrow \theta'$.

Basta tener presente que:

- $\neg\theta(\vec{x}) = \overline{sg}(\theta(\vec{x}))$
- $(\theta \wedge \theta')(\vec{x}) = \theta(\vec{x}) \cdot \theta'(\vec{x})$
- $(\theta \vee \theta')(\vec{x}) = sg(\theta(\vec{x}) + \theta'(\vec{x}))$
- $\theta \Rightarrow \theta' \equiv \neg\theta \vee \theta'$
- $\theta \Leftrightarrow \theta' \equiv \theta \Rightarrow \theta' \wedge \theta' \Rightarrow \theta$

Corolario. Si $A, B \subseteq \mathbb{N}^k \in \Delta^0_*$, entonces $\mathbb{N}^k - A$; $A \cap B$; $A \cup B \in \Delta^0_*$.

Basta tener presente que:

- $C_{\mathbb{N}^k - A} = \neg C_A$.
- $C_{A \cap B} = C_A \wedge C_B$.
- $C_{A \cup B} = C_A \vee C_B$.

Teorma de definición por casos para funciones PR: Sean $k \geq 2$ y $f_1, \ldots, f_k : \mathbb{N}^p \longrightarrow \mathbb{N}$ funciones **PR**. Sea $\{A_1, \ldots, A_k\} \in \Delta^0_*$ una partición de $\mathbb{N}^p$. Entonces, la función $g : \mathbb{N}^p \longrightarrow \mathbb{N}$ definida por

$$g(\vec{x}) = \begin{cases} f_1(\vec{x}) & si \quad \vec{x} \in A_1 \\ \vdots & \vdots \quad \vdots \\ f_k(\vec{x}) & si \quad \vec{x} \in A_k \end{cases}$$

es **PR**.

- Basta tener presente que $g = f_1 \cdot C_{A_1} + \ldots + f_k \cdot C_{A_k}$ es GOTO-computable.

Nota: La proposición anterior se puede expresar con predicados $\theta_1, \ldots, \theta_k$ que sean **PR**, exhaustivos y excluyentes (es decir, tales que, para todo $\vec{x} \in \mathbb{N}^p$: $\theta_1(\vec{x}) + \ldots + \theta_k(\vec{x}) = 1$)

Suma y producto acotados

Definición: Si $f : \mathbb{N}^{k+1} \longrightarrow \mathbb{N}$ es total, se definen:

- La suma acotada de f es: $\sum_f(\vec{x}, y) = \sum_{z \leq y} f(\vec{x}, z)$
- El producto acotado de f es: $\prod_f(\vec{x}, y) = \prod_{z \leq y} f(\vec{x}, z)$.

Observaciones:

- En lugar de la última variable, se podría utilizar cualquiera otra como cota.
- Generalizamos para $n = 0$: $\sum_f(y) = \sum_{z \leq y} f(z)$, $\prod_f(y) = \prod_{z \leq y} f(z)$.

Proposición: Si $f \in \mathbf{PR}^{(k+1)}$ entonces $\sum_f, \prod_f \in \mathbf{PR}^{(k+1)}$

Demostración para la suma acotada:

a) $\sum_f(\vec{x}, 0) = g(\vec{x})$

b) $\sum_f(\vec{x}, y+1) = h(\vec{x}, y, \sum_f(\vec{x}, y))$

Donde: $g(\vec{x}) = f(\vec{x}, 0)$ y $h(\vec{x}, y, z) = z + f(\vec{x}, y+1)$
Por tanto, $\sum_f = \mathcal{R}(g, h) \in \mathbf{PR}$ pues g y h son **PR**.

Cuantificación acotada de predicados

Proposición: Sea θ un predicado (k+1)-ario y **PR**. Los predicados siguientes son **PR**:

$$\theta_1(\vec{x}, y) \equiv \forall z \leq y\, \theta(\vec{x}, z) \quad \theta_2(\vec{x}, y) \equiv \exists z \leq y\, \theta(\vec{x}, z)$$

- Basta tener pesente que
  $\theta_1(\vec{x}, y) = \prod_{z \leq y} \theta(\vec{x}, z); \quad \theta_2(\vec{x}, y) = sg(\sum_{z \leq y} \theta(\vec{x}, z)).$
  O bien que:

$$\theta_1(\vec{x}, y) = sg(y + 1 \dot{-} \mu t \leq y(\theta(\vec{x}, t)))$$

$$\theta_2(\vec{x}, y) = \overline{sg}(y + 1 \dot{-} \mu t \leq y(\neg\theta(\vec{x}, t)))$$

Ejemplos.

- El predicado de divisibilidad, $x|y$, es GOTO–computable:

$$\theta(x, y) \equiv \exists z \leq y(y = z \cdot x)$$

- El predicado *ser primo* es GOTO–computable:

$$primo(x) \equiv (x > 1) \wedge \forall t \leq x((t|x) \rightarrow (t = 1 \vee t = x))$$

Minimización acotada de predicados

Definición. Sea $\theta(\vec{x}, y)$ un predicado $(k+1)$–ario. Definimos la función total $\theta^*_\mu : \mathbb{N}^{k+1} \longrightarrow \mathbb{N}$, así:

$$\theta^*_\mu(\vec{x}, y) = \begin{cases} min\{z \leq y : \theta(\vec{x}, z)\} & \text{si existe tal mínimo} \\ y+1 & \text{e.c.o.c.} \end{cases}$$

- Usualmente escribiremos $\mu z \leq y\ (\theta(\vec{x}, z))$ y diremos que la función θ^*_μ se obtiene del predicado θ por minimización acotada.

Proposición. Si $\theta \in \mathbf{PR}^{(k+1)}$ entonces $\theta^*_\mu \in \mathbf{PR}^{(k+1)}$.

- Basta tener presente que:

$$\theta^*_\mu(\vec{x}, y) = \sum_{t \leq y} (\prod_{z \leq t} \neg\theta(\vec{x}, z))$$

Minimización acotada de funciones

Definición. Sea $f(\vec{x}, y)$ una función $(k+1)$–aria y total. Definimos la función de aridad $k+1$, f^*_μ, así:

$$f^*_\mu(\vec{x}, y) = \mu z \leq y \, (f(\vec{x}, z) = 0)$$

- Diremos que la función f^*_μ se obtiene de la función f por minimización acotada.

Proposición: Si $f \in \mathbf{PR}^{(k+1)}$, entonces $f^*_\mu \in \mathbf{PR}^{(k+1)}$.

Ejemplos.

- Es **PR** la función cociente definida por

$$qt(x, y) = \begin{cases} 0 & \text{si } y = 0 \\ \mu t \leq x \, (x < (t+1) \cdot y) & \text{e.o.c.} \end{cases}$$

- Es **PR** la función resto definida por

$$rm(x, y) = \begin{cases} x \stackrel{\bullet}{-} (y \cdot qt(x, y)) & si \ y \neq 0 \\ 0 & si \ y = 0 \end{cases}$$

La sucesión de números primos

Consideremos la siguiente función $p : \mathbb{N} \to \mathbb{N}$ definida así:

$$p(n) = \begin{cases} 0 & \text{si} \quad n = 0 \\ \text{el } n\text{–ésimo primo} & \text{si} \quad n \geq 1 \end{cases}$$

La función p proporciona la sucesión de números primos (para $n \geq 1$).
Notaremos $p(n) = p_n$.

Proposición. La función p es primitiva recursiva.

Para demostrar este resultado hay que usar el siguiente lema de Euclides:

- **Lema**. $\forall n\,(p_{n+1} \leq 1 + p_n!)$.

Entonces basta describir la función p como sigue:

$$\begin{cases} p(0) & = & 0 \\ p(n+1) & = & \mu y \leq (1 + p(n)!)\ (primo(y) \wedge y > p(n)) \end{cases}$$