
3. Definiciones Recursivas

Recursión

- Dado un conjunto inductivo, sabemos exactamente cómo se construyen sus elementos.
- Esta información sirve para:
 - Probar propiedades de sus elementos (inducción)✓
 - Definir funciones sobre sus elementos (*recursión*)

Qué es una función?

- Una función es una relación que:
 - Asocia **efectivamente** un único elemento del codominio a cada elemento del dominio.
- Una función es un mecanismo de cómputo tal que:
 - Cada vez que se le da un valor del dominio como entrada devuelve **efectivamente** el mismo valor del codominio.
- “Efectivamente” significa:
 - Para cualquier elemento del dominio hay una imagen.
 - El cómputo termina para cualquier elemento del dominio.
 - Ningún elemento del dominio tienen más de una imagen.

Recursión

- Ejemplo: definir una función f que calcula la cantidad de símbolos de una palabra de $\{a,b\}^*$
- Para asegurarse de cubrir todo el dominio se puede explotar conocimiento de cómo se generan los objetos de $\{a,b\}^*$:
 - $f(\epsilon)=0$
 - $f(aw)=1+f(w)$
 - $f(bw)=1+f(w)$

Es fácil definir No Funciones.

- Dado Z definido inductivamente como:

- $0 \in Z$
- $n + 1 \in Z$
- $n - 1 \in Z$

Conclusión:

Conviene seguir un esquema de definición que garantice que se define una función.

- $f: Z \rightarrow N$

- $f(0)=0$
- $f(n+1)=f(n)+1$
- $f(n-1)=f(n)+1$

Tiene más de un valor

- Cuánto vale $f(1)$?

- $f(0)=0$
- $f(n+1)=f(n+2)$
- $f(n-1)=f(n-2)$

Calcular $f(1)$ necesita $f(2), f(3), \dots$
Nunca se termina de calcular.

Esquema de Recursión Primitiva para N

- N definido inductivamente por:

- $0 \in N$
- Si $n \in N$ entonces $S(n) \in N$

- Esquema de Recursión Primitiva para N
(informal)

- Sea B un conjunto cualquiera.
- Entonces un conjunto de ecuaciones como sigue basta para definir una única función $F: N \rightarrow B$
 - $F(0) = \dots$
 - $F(S(n)) = ..F(n)..n..$

Aplicación del Esquema de Recursión

- Esquema de recursión primitiva para \mathbb{N} :
 - Método que se aplica para definir funciones sobre objetos de \mathbb{N}
 - Usa el conocimiento de cómo se generan los objetos de \mathbb{N}
 - Justificación: Recordar que al dar una definición inductiva decimos que la única forma de construir los objetos del conjunto es mediante las cláusulas dadas
- Aplicaciones:
 - Definir $+$; x ; $\sum_{1 \leq k \leq n} k$; k^n , para $k, n \in \mathbb{N}$; *fact* : $\mathbb{N} \rightarrow \mathbb{N}$

Esquema de Recursión Primitiva para un Conjunto Inductivo

- Sea A un conjunto definido inductivamente
- Para definir una función $f : A \rightarrow B$, alcanza con definir a f mediante ecuaciones que determinen:
 - El valor de f para los objetos de A obtenidos de aplicar cláusulas base
 - El valor de f para los objetos de A obtenidos de aplicar cláusulas inductivas, utilizando el valor de f en el (los) objeto(s) anterior(es) y también el (los) objeto(s) anterior(es) (*llamadas recursivas*)

Esquema de Recursión Primitiva para Σ^*

- Sea B un conjunto arbitrario.
- Entonces un conjunto de ecuaciones como sigue basta para definir una única función $F: \Sigma^* \rightarrow B$
 - i. $F(\epsilon) = \dots$
 - ii. $F(xw) = ..F(w)..w..x..$
- **Aplicaciones:**
 - Definir longitud: $\Sigma^* \rightarrow \mathbb{N}$; inversa: $\Sigma^* \rightarrow \Sigma^*$;
palindromo: $\Sigma^* \rightarrow \text{Bool}$; espejo: $\Sigma^* \rightarrow \Sigma^*$

Esquema de Recursión Primitiva para L1

- Sea B un conjunto arbitrario.
- Entonces un conjunto de ecuaciones como sigue basta para definir una única función $F: L1 \rightarrow B$
 - i. $F(a) = \dots$
 - ii. $F(bwb) = ..F(w)..w..$

Esquema de Recursión Primitiva para N -Formalización-

- **H) Sea B un conjunto y sean**
 - $f_0 \in B$
 - $f_s: N \times B \rightarrow B$
- **T) Entonces existe una única función**
F: N \rightarrow B tal que:
 - i. $F(0) = f_0$
 - ii. $F(S(n)) = f_s(n, F(n))$

Esquema de Recursión Primitiva -Formalización-

Ejemplo:

$$\begin{array}{l|l} \text{fact}(0) = 1 & f_0 = 1 \ (\in B) \\ \text{fact}(S(n)) = S(n) \times \text{fact}(n) & f_s(n,r) = S(n) \times r \ (N \times B \rightarrow B) \end{array}$$

Esquema de Recursión Primitiva -Formalización-

Esquema de Recursión Primitiva para Σ^*

Sea B un conjunto y sean

- $f_\epsilon \in B$
- $f_1: \Sigma \times \Sigma^* \times B \rightarrow B$

Entonces existe una única función $F: \Sigma^* \rightarrow B$ tal que:

- i. $F(\epsilon) = f_\epsilon$
- ii. $F(xw) = f_1(x, w, F(w))$

Ejemplo:

$\text{espejo}(\epsilon) = \epsilon$ $\text{espejo}(xw) = x \text{ espejo}(w)$	$f_\epsilon = 1 \ (\epsilon \in \Sigma^*)$ $f_1(x, w, r) = xr \ (\Sigma \times \Sigma^* \times \Sigma^* \rightarrow \Sigma^*)$
---	---

Otros Esquemas de Recursión

- Dado A definido inductivamente. Para definir $f: A \rightarrow B$ se debe:
 - definir f para los objetos base de A
 - definir f en los objetos obtenidos de aplicar cláusulas inductivas usando el valor de f en el objeto **inmediatamente anterior**
 (recursión primitiva)
- Se puede cambiar el segundo punto y poner:
 - definir f en los objetos obtenidos de aplicar cláusulas inductivas usando el valor de f objetos **anteriores**

Ejemplo de Recursión no Primitiva en \mathbf{N}

$$\text{fib} : \mathbf{N} \rightarrow \mathbf{N}$$

$$\text{fib}(0) = 1$$

$$\text{fib}(1) = 1$$

$$\text{fib}(n+2) = \text{fib}(n) + \text{fib}(n+1)$$

- Para que estas ecuaciones *definan una función* hay que probar que:
 - son *exhaustivas*
 - no hay *superposición* de casos
 - la definición *termina*

Ejemplo de Recursión no Primitiva en \mathbf{N}

$$\text{fib} : \mathbf{N} \rightarrow \mathbf{N}$$

$$\text{fib}(0) = 1$$

$$\text{fib}(1) = 1$$

$$\text{fib}(n+2) = \text{fib}(n) + \text{fib}(n+1)$$

Exhaustividad: ver que todo natural es 0, 1 o de la forma $n+2$

Superposición: ver que $0 \neq 1 \neq n+2$

Terminación: 1) ver que para todo $n \in \mathbf{N}$:

$$n < n+2 \quad \text{y}$$
$$n+1 < n+2$$

2) en \mathbf{N} toda cadena decreciente según la relación $<$ tiene mínimo (el orden $<$ de \mathbf{N} es *bien fundado*).

Ejemplo de Recursión no Primitiva en \mathbf{N}

$$\mathit{div} : \mathbf{N} \times \mathbf{N}^+ \rightarrow \mathbf{N}$$

$$\mathit{div}(n, m) = 0 \quad \text{si } n < m$$

$$\mathit{div}(n, m) = \mathit{div}(n-m, m) + 1 \quad \text{si } m \leq n$$

- Exhaustividad: ver que para todos $n, m \in \mathbf{N}$ se cumple que $n < m$ o $m \leq n$.
- Superposición: ver que para todos $n, m \in \mathbf{N}$ no puede pasar que $n < m$ y $m \leq n$.
- Terminación: ver que para todos $n, m \in \mathbf{N}$, si $0 < m \leq n$ entonces $n-m < n$ y ver que $<$ es bien fundado en \mathbf{N}

Definición inductiva de $\mathbf{N} \times \mathbf{N}$

i. $(0,0) \in \mathbf{N} \times \mathbf{N}$

ii. Si $(n,0) \in \mathbf{N} \times \mathbf{N}$ entonces $(n+1,0) \in \mathbf{N} \times \mathbf{N}$

iii. Si $(n, m) \in \mathbf{N} \times \mathbf{N}$ entonces $(n, m+1) \in \mathbf{N} \times \mathbf{N}$

• Principio de inducción primitiva para $\mathbf{N} \times \mathbf{N}$

• Sea P una propiedad que cumple:

– $P(0,0)$

– Si $P(n,0)$ entonces $P(n+1,0)$

– Si $P(n, m)$ entonces $P(n, m+1)$

• Entonces, $P(n, m)$ para todo $(n, m) \in \mathbf{N} \times \mathbf{N}$

Esquema de Recursión Primitiva para $\mathbf{N \times N}$

- Sea \mathbf{B} un conjunto y H_0, H_1 y H_2 tales que:
 - $H_0 \in \mathbf{B}$
 - $H_1 : \mathbf{N} \times \mathbf{B} \rightarrow \mathbf{B}$
 - $H_2 : \mathbf{N} \times \mathbf{N} \times \mathbf{B} \rightarrow \mathbf{B}$
- Entonces existe una única función $f : \mathbf{N} \times \mathbf{N} \rightarrow \mathbf{B}$ tal que:
 - $f(0,0) = H_0$
 - $f(n+1,0) = H_1(n, f(n,0))$
 - $f(n,m+1) = H_2(n, m, f(n,m))$

Ejemplo: *prod* : $\mathbf{N} \times \mathbf{N} \rightarrow \mathbf{N}$ (el producto de dos números Naturales)

Ejemplos de Recursión no Primitiva en $\mathbf{N \times N}$

$$\mathit{mcd} : \mathbf{N^+ \times N^+} \rightarrow \mathbf{N^+}$$

$$\mathit{mcd}(n, m) = n \quad \text{si } n = m$$

$$\mathit{mcd}(n, m) = \mathit{mcd}(n, m-n) \quad \text{si } n < m$$

$$\mathit{mcd}(n, m) = \mathit{mcd}(n-m, m) \quad \text{si } m < n$$

Exhaustividad: ???

Superposición: ???

Terminación: ??

EJERCICIO!!!

Más ejemplos de Recursión en $\mathbb{N} \times \mathbb{N}$

$suma : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$

$suma(n, 0) = n$

$suma(n, m+1) = suma(n, m) + 1$

¿Corrección?

$resta : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$

$resta(0, n) = 0$

$resta(n+1, 0) = n + 1$

$resta(n+1, m+1) = ???$

¿Corrección?

Definiciones Inductivas Libres

- Definimos \mathbf{Z} inductivamente con las siguientes cláusulas:
 - $0 \in \mathbf{Z}$
 - si $x \in \mathbf{Z}$ entonces $x+1 \in \mathbf{Z}$
 - si $x \in \mathbf{Z}$ entonces $x-1 \in \mathbf{Z}$
- ¿Cómo justificar que $3 \in \mathbf{Z}$?
 - $0 \rightarrow^{\text{ii}} 1 \rightarrow^{\text{ii}} 2 \rightarrow^{\text{ii}} 3$
 - $0 \rightarrow^{\text{iii}} -1 \rightarrow^{\text{iii}} -2 \rightarrow^{\text{ii}} -1 \rightarrow^{\text{ii}} 0 \rightarrow^{\text{ii}} 1 \rightarrow^{\text{ii}} 2 \rightarrow^{\text{ii}} 3$
 - $0 \rightarrow^{\text{ii}} 1 \rightarrow^{\text{ii}} 2 \rightarrow^{\text{ii}} 3 \rightarrow^{\text{ii}} 4 \rightarrow^{\text{ii}} 5 \rightarrow^{\text{iii}} 4 \rightarrow^{\text{iii}} 3$

Definiciones Inductivas Libres (cont.)

Definición: [definición inductiva libre]

Una definición inductiva de un conjunto es *libre* cuando cada elemento del conjunto admite una sola secuencia de formación.

Las definiciones que no son libres traen problemas para definir funciones usando esquemas de recursión

Ejemplo: $f : \mathbf{Z} \rightarrow \mathbf{Z}$ definida por

$$f(0) = 1$$

$$f(n+1) = 2 + f(n)$$

$$f(n-1) = 3 + f(n)$$

¿Cuánto vale $f(3)$?

Definiciones no Libres: moraleja

- Las definiciones inductivas no libres son problemáticas para definir funciones usando el esquema de recursión primitiva.
 - Igual se pueden usar, pero hay que verificar que se está definiendo **una función** (o sea, un **único resultado** para cada elemento del conjunto)
- No hay problemas con el principio de inducción (salvo que probamos más de una vez la misma cosa)

Definiciones Recursivas: moraleja

- Sea A un conjunto definido inductivamente
 - Si la definición de A es ***libre*** entonces se puede aplicar el *esquema de recursión primitiva* sin problemas
 - Si se quieren ***usar otros esquemas*** entonces hay que probar ***exhaustividad, no superposición y terminación***
 - Si la definición de A ***no es libre***, no alcanza el esquema de recursión, dado que **hay** superposición. Hay que probar ***unicidad de la definición*** (los casos repetidos dan el mismo resultado)