Lógica

Mauro Polenta Mora

CLASE 3 - 10/2/2025

Recursión

Observación

Dado un conjunto inductivo, sabemos exactamente como se construyen sus elementos. Esta información sirve para:

- Probar propiedades de los elementos del conjunto (inducción)
- Definir funciones sobre sus elementos (recursión)

Definición (función)

Tenemos varias formas de ver lo que es una función en este contexto:

1. Una función es una relación que asocia un único elemento del codominio a cada elemento del dominio.
2. Una función es un mecanismo de cómputo que para cada valor de entrada (valor del dominio) devuelve **efectivamente** un valor de salida (valor del codominio).

Observación: Donde efectivamente significa que la función termina de computar en un tiempo finito para cualquier elemento del dominio

Esquema de recursión primitiva para $\mathbb{N}$ (informal)

Sea $\mathbb{N} \subseteq \mathbb{R}$ definido inductivamente por:

1. $0 \in \mathbb{N}$
2. Si $n \in \mathbb{N}$, entonces $n + 1 \in \mathbb{N}$

Planteamos el esquema de recursión primitiva (ERP) para $\mathbb{N}$ de la siguiente forma:

Sea B un conjunto cualquiera. Entonces para definir una única función $F : \mathbb{N} \to B$ basta con:

1. $F(0) = ...$
2. $F(n + 1) = ... F(n) ... n ...$

Donde lo dado por los puntos suspensivos es lo que se debe completar para definir la función F.

Ejemplo 1 (ERP)

Definamos el factorial de un número usando el ERP para $\mathbb{N}$.

1. $F(0) = 1$
2. $F(n+1) = F(n) \cdot (n+1)$

Ejemplo 2 (ERP)

Sea $L_1 \subset \{a, b\}^*$ definido inductivamente por:

1. $a \in L_1$
2. Si $w \in L_1$, entonces $bwb \in L_1$

Planteemos el ERP para L_1.

1. $F(a) = ...$
2. $F(bwb) = ... F(w) ... w ...$

Ejemplo 3 (ERP)

Sea $\sum^* \subseteq \sum^*$ definido inductivamente por:

1. $\varepsilon \in \sum^*$
2. Si $w \in \sum^*$ y $x \in \sum$, entonces $xw \in \sum^*$

Planteemos el ERP para $\sum^*$.

1. $F(\varepsilon) = ...$
2. $F(xw) = ... F(w) ... x ... w ...$

Ejemplo 1 (funciones definidas usando ERP)

Definamos la función $F : \sum^* \to \mathbb{N}$ que cuenta la cantidad de letras a en una palabra.

1. $F(\varepsilon) = 0$
2. $F(xw) = F(w) + 1$

Ejemplo 2 (funciones definidas usando ERP)

Definamos la función $F : \sum^* \to \{0, 1\}$ que indica si una palabra es vacía

1. $F(\varepsilon) = 1$
2. $F(xw) = 0$

Ejemplo 3 (funciones definidas usando ERP)

Definamos la función $F : \sum^* \to \sum^*$ que nos devuelve el espejo de una palabra. Por ejemplo $F(ab) = abba$.

1. $F(\varepsilon) = \varepsilon$
2. $F(xw) = x \cdot F(w) \cdot x$

Observación: En este caso, no podemos afirmar que la función está bien definida, ya que definimos $\sum^*$ con la inserción por la izquierda, y en este caso la función también inserta a la derecha. A priori no podemos confirmar que un elemento tras la inserción por la derecha también forme parte de $\sum^*$.

Definición (ERP formalizado para $\mathbb{N}$)

(H) Sea B un conjunto y:

1. $f_0 \in B$
2. $f_s : \mathbb{N} \times B \to B$

(T) Entonces existe una función única $F : \mathbb{N} \to B$ tal que:

1. $F(0) = f_0$
2. $F(n+1) = f_s(n, F(n))$

Ejemplo (factorial)

Factorial: $f_0 = 1$ y $f_s(n, x) = x \cdot (n+1)$, entonces: - $F(0) = 1$ - $F(n+1) = f_s(n, F(n)) = F(n) \cdot (n+1)$

Definición (ERP para L_1)

(H) Sea B un conjunto y:

1. $f_a \in B$
2. $f_s : L_1 \times B \to B$

(T) Entonces existe una función única $F : L_1 \to B$ tal que:

1. $F(a) = f_a$
2. $F(bwb) = f_s(w, F(w))$

Definición (ERP para $\sum^*$)

(H) Sea B un conjunto y:

1. $f_\varepsilon \in B$
2. $f_s : \sum \times \sum^* \times B \to B$

(T) Entonces existe una función única $F : \sum^* \to B$ tal que:

1. $F(\varepsilon) = f_\varepsilon$
2. $F(xw) = f_s(x, w, F(w))$

Ejemplo (función espejo)

Espejo: $f_\varepsilon = \varepsilon$ y $f_s(x, w, y) = x \cdot y \cdot x$, entonces: - $F(\varepsilon) = \varepsilon$ - $F(xw) = f_s(x, w, F(w)) = x \cdot F(w) \cdot x$

Definición (definición inductiva libre)

Decimos que una definición es libre cuando cada elemento del conjunto, se forma de una única manera.

Dado $\mathbb{X}$ un conjunto inductivo definido por:

1. $3 \in \mathbb{X}$
2. Si $x \in \mathbb{X}$, entonces $x - 2 \in \mathbb{X}$
3. Si $x, y \in \mathbb{X}$, entonces $x + y \in \mathbb{X}$

No deberíamos usar definiciones inductivas no libres para definir funciones. Por ejemplo, si definimos $f : \mathbb{X} \to \mathbb{N}$ con las ecuaciones:

1. $f(3) = 0$
2. $f(n - 2) = 0$
3. $f(x + y) = 1 + f(x) + f(y)$

No podemos afirmar que la función está bien definida, ya que pueden haber múltiples formas de llegar a un mismo elemento de $\mathbb{N}$.

Resumen (ERP)

Para definir $f : A \to B$ se debe

- definir f para los objetos base de A, y
- definir f en los objetos obtenidos de aplicar cláusulas inductivas usando el valor de f en objetos inmediatamente anteriores

Resumen (ERG - Esquema de recursión general)

Para definir $f : A \to B$ se debe

- definir f para los objetos base de A, y
- definir f en los objetos obtenidos de aplicar cláusulas inductivas usando el valor de f obtenido para objetos estrictamente menores

Ejemplo (ERG)

Sea $FIBO : \mathbb{N} \to \mathbb{N}$ definida por:

1. $FIBO(0) = 0$
2. $FIBO(1) = 1$
3. $FIBO(n + 2) = FIBO(n + 1) + FIBO(n)$

Veamos las condiciones que cumple para ser una función:

- **Exhaustividad**: Todo natural es cero, uno, o de la forma $n + 2$; hay alguna regla que lo computa
- **No superposición**: Ser cero, uno, o de la forma $n+2$ son condiciones mutuamente incompatibles; es decir, cada cómputo está únicamente determinado.
- **Terminación**: Usando el orden habitual tenemos que $n < n + 2$ y $n + 1 < n + 2$

Resumen

Sea A un conjunto definido inductivamente.

- Si la definición es libre, se puede aplicar sinproblemas el esquema de recursión primitiva.
- Si la definición no es libre, hay superposición. Hay que probar que los casos repetidos dan el mismo resultado.
- Si se usa un esquema de recursión general hay que probar exhaustividad, no superposición y terminación