FP de John Backus

En su artículo *Can Programming Be Liberated from the von Neumann Style? A Functional Style and Its Algebra of Programs*, publicado en razón de su premiación con el *ACM Turing Award* de 1977, John Backus expuso su visión crítica hacia el diseño de los lenguajes convencionales de aquel entonces (vale mencionar que él mismo, años antes, había formado parte del equipo de diseñadores del FORTRAN) y mostró la potencia de la programación funcional a través de su sistema FP.

El sistema FP propuesto por Backus consiste de 5 partes:

1. Un conjunto O de *objetos*;
2. Una única operación, la *aplicación*;
3. Un conjunto F de *funciones f* que convierten objetos en otros objetos;
4. Un conjunto FF de *formas funcionales*, usadas para combinar funciones u objetos existentes, y formar con ellos nuevas funciones en F;
5. Un conjunto D de *definiciones* de funciones de F.

1. OBJETOS

Un *objeto* puede ser:

- un *átomo*, es decir, una cadena no nula formada por caracteres (letras, dígitos, etc.) excluyendo los utilizados por la notación del sistema FP;
- una *secuencia* $<x_1, ..., x_n>$ cuyos elementos x_i son objetos;
- *indefinido* (⊥)

El átomo Ø representa la secuencia vacía, y es el único objeto que es a la vez átomo y secuencia.

Los átomos *T* y *F* se utilizan para representar los valores verdadero y falso, respectivamente.

Si una secuencia contiene ⊥, está indefinida. Por ejemplo: *<5, A, 7, 4, ⊥, 3>* = ⊥

Ejemplos de *objetos*:

⊥	*1.5*	
Ø	*AB3*	
<AB, 1, 2.3>	*<A, <<B>, C>, D>*	
<A, ⊥>	<>	Obs: <> equivale a Ø

2. APLICACIÓN

Si *f* es una función y *x* es un objeto, entonces *f* : *x* es una *aplicación* y representa el objeto que resulta cuando se le aplica *f* a *x*.

f es el *operador* de la aplicación y *x* es el *operando*.

Ejemplos de *aplicaciones*:

+ : *<1, 2>* resulta *3*	tl : *<A, B, C>* resulta *<B, C>*
1 : *<A, B, C>* resulta *A*	2 : *<A, B, C>* resulta *B*

3. FUNCIONES

Todas las funciones f del conjunto F convierten objetos en otros objetos, y preservan el valor indefinido ($f : \perp = \perp$).

Las funciones *primitivas* son las funciones básicas provistas por el sistema FP.

3.1. Selectores

a) Selector desde la izquierda

Para cualquier entero positivo s

$s : x \equiv x = <x_1, ..., x_n> \wedge s \leq n \;\rightarrow x_s \;;\; \perp$

Por ejemplo:

$1 : <A, B, C>$ resulta A

$2 : <4, 3, 2>$ resulta 3

$4 : <A, B, C>$ resulta $\perp$

Obs: los símbolos 1, 2, etc. son nombres de funciones distintos de los átomos *1*, *2*, etc.

b) Selector desde la derecha

Para cualquier entero positivo s

$sr : x \equiv x = <x_1, ..., x_n> \wedge s \leq n \;\rightarrow x_{n-s+1} \;;\; \perp$

Por ejemplo:

$1r : <A, B, C>$ resulta C

$2r : <A, B, C>$ resulta B

$4r : <A, B, C>$ resulta $\perp$

c) Cola desde la izquierda

$tl : x \equiv x = <x_1> \rightarrow \emptyset;$

$\qquad x = <x_1, ..., x_n> \wedge n \geq 2 \rightarrow <x_2, ..., x_n> \;;\; \perp$

Por ejemplo:

$tl : <A, B, C>$ resulta $<B, C>$

d) Cola desde la derecha

$tlr : x \equiv x = <x_1> \rightarrow \emptyset;$

$\qquad x = <x_1, ..., x_n> \wedge n \geq 2 \rightarrow <x_1, ..., x_{n-1}>; \perp$

Por ejemplo:

$tlr : <A, B, C>$ resulta $<A, B>$

e) Identidad

$id : x \equiv x$

Por ejemplo:

$id : <A, B, C>$ resulta $<A, B, C>$

3.2. Predicados

a) atom

atom : $x \equiv x$ es átomo $\rightarrow T$;
$x \neq \perp \rightarrow F$; $\perp$

Por ejemplo:

atom : *5* resulta *T*

atom : $<A, B, C>$ resulta *F*

b) eq

eq : $x \equiv x = <y, z> \wedge y = z \rightarrow T$;
$x = <y, z> \wedge y \neq z \rightarrow F$; $\perp$

Por ejemplo:

eq : $<A, A>$ resulta *T*

eq : $<A, 7>$ resulta *F*

eq : $<A, B, C>$ resulta $\perp$

Aunque Backus no menciona las funciones > y <, la existencia de estas puede ser útil.

c) null

null : $x \equiv x = \emptyset \rightarrow T$;
$x \neq \perp \rightarrow F$; $\perp$

Por ejemplo:

null : $\emptyset$ resulta *T*

null : $<A, 7>$ resulta *F*

3.3. Funciones aritméticas

a) Suma

+ : $x \equiv x = <y, z> \wedge y, z$ son números $\rightarrow y + z$; $\perp$

Por ejemplo:

+ : $<2, 7>$ resulta *9*

+ : $<3, A, 7>$ resulta $\perp$

b) Resta

- : $x \equiv x = <y, z> \wedge y, z$ son números $\rightarrow y - z$; $\perp$

Por ejemplo:

- : $<9, 7>$ resulta *2*

c) Producto

× : $x \equiv x = <y, z> \wedge y, z$ son números $\rightarrow y \times z$; $\perp$

Por ejemplo:

× : $<2, 7>$ resulta *14*

d) Cociente

$\div : x \equiv x = <y, z> \wedge y, z$ son números $\rightarrow y \div z \,;\, \perp$ (además, $y \div 0 = \perp$)

Por ejemplo:

$\div : <10, 2>$ resulta 5

$\div : <10, 0>$ resulta $\perp$

3.4. Funciones lógicas

a) and

and **:** $x \equiv x = <T, T> \rightarrow T;$
$x = <F, T> \vee x = <T, F> \vee x = <F, F> \rightarrow F \,;\, \perp$

Por ejemplo:

and : $<T, F>$ resulta F

and : $<1, 0>$ resulta $\perp$

b) or

or **:** $x \equiv x = <F, F> \rightarrow F;$
$x = <T, T> \vee x = <T, F> \vee x = <F, T> \rightarrow T \,;\, \perp$

Por ejemplo:

or : $<T, F>$ resulta T

c) not

not **:** $x \equiv x = T \rightarrow F;$
$x = F \rightarrow T; \perp$

Por ejemplo:

not : F resulta T

3.5. Funciones para manipular secuencias

a) Longitud

length : $x \equiv x = <x_1, \ldots, x_n> \rightarrow$ n ;
$x = \emptyset \rightarrow 0; \perp$

Por ejemplo:

length : $<2, A, 7>$ resulta 3

length : $\emptyset$ resulta 0

b) Invertir

reverse : $x \equiv x = \emptyset \rightarrow \emptyset;$
$x = <x_1, \ldots, x_n> \rightarrow <x_n, \ldots, x_1>; \perp$

Por ejemplo:

reverse : $<2, A, 7>$ resulta $<7, A, 2>$

reverse : $\emptyset$ resulta $\emptyset$

c) Transponer

trans : $x \equiv x = \langle \emptyset, \ldots, \emptyset \rangle \rightarrow \emptyset;$
$x = \langle x_1, \ldots, x_n \rangle \rightarrow \langle y_1, \ldots, y_m \rangle; \perp$

Donde $x_i = \langle x_{i1}, \ldots, x_{im} \rangle$, $y_j = \langle y_{1j}, \ldots, y_{nj} \rangle$, $1 \leq i \leq n$, $1 \leq j \leq m$.

Por ejemplo:
trans: $\langle\langle 1, 2, 3 \rangle, \langle A, B, C \rangle\rangle$ resulta $\langle\langle 1, A \rangle, \langle 2, B \rangle, \langle 3, C \rangle\rangle$

d) Distribuir desde la izquierda

distl : $x \equiv x = \langle y, \emptyset \rangle \rightarrow \emptyset$;
$x = \langle y, \langle x_1, \ldots, x_n \rangle\rangle \rightarrow \langle\langle y, x_1 \rangle, \langle y, x_2 \rangle, \ldots, \langle y, x_n \rangle\rangle ; \perp$
Por ejemplo:
distl : $\langle A, \langle 1, 2, 3 \rangle\rangle$ resulta $\langle\langle A, 1 \rangle, \langle A, 2 \rangle, \langle A, 3 \rangle\rangle$

e) Distribuir desde la derecha

distr : $x \equiv x = \langle \emptyset\ \ y \rangle \rightarrow \emptyset$;
$x = \langle\langle x_1, \ldots, x_n \rangle, y \rangle \rightarrow \langle\langle x_1, y \rangle, \langle x_2, y \rangle, \ldots, \langle x_n, y \rangle\rangle ; \perp$
Por ejemplo:
distr : $\langle\langle 1, 2, 3 \rangle, A \rangle$ resulta $\langle\langle 1, A \rangle, \langle 2, A \rangle, \langle 3, A \rangle\rangle$

f) Concatenar a la izquierda

apndl : $x \equiv x = \langle y, \langle x_1, \ldots, x_n \rangle\rangle \rightarrow \langle y, x_1, \ldots, x_n \rangle$;
$x = \langle y, \emptyset \rangle \rightarrow \langle y \rangle ; \perp$
Por ejemplo:
apndl : $\langle\langle A, B \rangle, \langle C, D \rangle\rangle$ resulta $\langle\langle A, B \rangle, C, D \rangle$

g) Concatenar a la derecha

apndr : $x \equiv x = \langle\langle x_1, \ldots, x_n \rangle, y \rangle \rightarrow \langle x_1, \ldots, x_n, y \rangle$;
$x = \langle \emptyset, y \rangle \rightarrow \langle y \rangle ; \perp$
Por ejemplo:
apndr : $\langle\langle A, B \rangle, \langle C, D \rangle\rangle$ resulta $\langle A, B, \langle C, D \rangle\rangle$

h) Rotar hacia la izquierda

rotl : $x \equiv x = \langle x_1, \ldots, x_n \rangle \rightarrow \langle x_2, \ldots, x_n, x_1 \rangle$
$x = \emptyset \rightarrow \emptyset$;
$x = \langle x_1 \rangle \rightarrow \langle x_1 \rangle; \perp$
Por ejemplo:
rotl : $\langle A, B, C, D \rangle$ resulta $\langle B, C, D, A \rangle$

i) Rotar hacia la derecha

rotr : $x \equiv x = \langle x_1, \ldots, x_n \rangle \rightarrow \langle x_n, x_1, \ldots, x_{n-1} \rangle$
$x = \emptyset \rightarrow \emptyset$;
$x = \langle x_1 \rangle \rightarrow \langle x_1 \rangle; \perp$

Por ejemplo:
rotr : $\langle A, B, C, D \rangle$ resulta $\langle D, A, B, C \rangle$

4. FORMAS FUNCIONALES

4.1. Composición

$f \circ g : x \equiv f : (g : x)$

Por ejemplo:

$1 \circ tl : <A, B, C>$ resulta *B*

4.2. Construcción

$[f_1, ..., f_n] : x \equiv <f_1 : x, f_2 : x, ..., f_n : x>$

Por ejemplo:

$[tl, tlr] : <A, B, C>$ resulta $<<B, C>, <A, B>>$

4.3. Condición

$$(p \rightarrow f; g) : x \equiv (p : x) = T \rightarrow f : x ; \quad (p : x) = F \rightarrow g : x ; \perp$$

Por ejemplo:

$(not \circ atom \rightarrow 1; id) : <A, B, C>$ resulta *A*

4.4. Constante

$\overline{X} : y \equiv y = \perp \rightarrow \perp; X$

Por ejemplo:

$+ \circ [id, \overline{1}] : 3$ resulta *4*

4.5. Inserción

$$/f : x \equiv x = <x_1> \rightarrow x_1 ; \quad x = <x_1, ..., x_n> \rightarrow f : <x_1, /f : <x_2, ..., x_n>> ; \perp$$

Por ejemplo:

$/+ : <1, 2, 3> \equiv + : <1, + : <2, + : <3>>> \equiv + : <1, + : <2, 3>>$ resulta 6

4.6. Aplicación a todos

$$\alpha f : x \equiv x = \emptyset \rightarrow \emptyset; \quad x = <x_1, ..., x_n> \rightarrow <f : x_1, ..., f : x_n> ; \perp$$

Por ejemplo:

$\alpha 1 : <<A, B, C>, <4, 5, 6>>$ resulta $<A, 4>$

4.7. Binario a unario

$(bu\ f\ x) : y \equiv f : <x, y>$

Por ejemplo:

$(bu + 1) : 4 \equiv + : <1, 4>$ resulta *5*

4.8. While

$$(\text{while } p\ f) : x \equiv p : x = F \rightarrow x\ ;\quad p : x = T \rightarrow (\text{while } p\ f) : (f : x)\ ;\ \bot$$

Por ejemplo:

(while (not $\circ$ null $\circ$ tl) tl) : <*A, B, C, D, E, F, G, H*> resulta <*H*>

5. DEFINICIÓN DE FUNCIONES

5.1. Función iota

Def iota $\equiv$ funrec $\circ$ [id, $\overline{<>}$]

Def funrec $\equiv$ < $\circ$ [1, $\overline{1}$] $\rightarrow$ 2; funrec $\circ$ [- $\circ$ [1, $\overline{1}$], apndl]

Obs: Aquí se usa la función <

iota : *5* resulta <*1, 2, 3, 4, 5*>

5.2. Función factorial

Def ! $\equiv$ eq0 $\rightarrow$ $\overline{1}$; $\times$ $\circ$ [id, ! $\circ$ sub1]

Obs: Esta definición es explícitamente recursiva

Def eq0 $\equiv$ eq $\circ$ [id, $\overline{0}$]

Def sub1 $\equiv$ - $\circ$ [id, $\overline{1}$]

! : *4* resulta *24*

Def fact $\equiv$ eq0 $\rightarrow$ $\overline{1}$; (/$\times$) $\circ$ iota

Obs: Esta definición es más *funcional*

fact : *4* resulta *24*

5.3. Función producto interno

Def IP $\equiv$ (/+) $\circ$ ($\alpha\times$) $\circ$ trans

IP : <<1, 2, 3*>, <4, 5, 6>>* resulta *32*

5.4. Función producto matricial

Def MM $\equiv$ (α (α IP)) $\circ$ (α distl) $\circ$ distr $\circ$ [1, trans $\circ$ 2]

MM : <<<1, 2, 3*>, <4, 5, 6>, <7, 8, 9>>*, <<1, 1, 1>, <0, 0, 0>, <0, 1, 0>>>
resulta <*<1, 4, 1>, <4, 10, 4>, <7, 16, 7>>*

Ejercicios

1. Definir funciones que devuelvan como resultado:

   a) El máximo de dos números.
   b) El máximo de una secuencia.
   c) El primer átomo de una secuencia.
   d) El elemento mínimo entre los máximos por fila de una matriz (*minimax*).

2. Definir funciones que determinen:

   a) La pertenencia de un elemento a una secuencia.
   b) Si una secuencia tiene un solo componente.
   c) Si la cantidad de átomos de una secuencia es par.

3. Dada una secuencia con dos subsecuencias, definir funciones para determinar:

   a) La unión de ambas subsecuencias.
   b) La intersección de ambas subsecuencias.
   c) La diferencia de ambas subsecuencias.
   d) La diferencia simétrica de ambas subsecuencias.

4. Definir una función que aplicada sobre un número natural *n*; obtenga como resultado el máximo valor resultante de aplicar cierta función B (predefinida) sobre el intervalo natural que finaliza en *n* (Máximo entre B:1; B:2; ... B:*n*).

5. Definir funciones que permitan:

   a) Planchar una secuencia.
   b) Concatenar dos subsecuencias planchadas.
   c) Invertir totalmente una secuencia.
   d) Ordenar una secuencia.
   e) Calcular la profundidad de una secuencia (niveles de subsecuencias).

6. Dados dos vectores $p=(p_1, p_2, \ldots, p_n)$ y $q=(q_1, q_2, \ldots, q_n)$ del espacio euclídeo $\mathbb{R}^n$, la distancia entre p y q está dada por la siguiente fórmula:

$$d(p,q) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

   Definir la función *distancia al cuadrado*, aplicable a una secuencia compuesta por dos subsecuencias, cada una de las cuales representa un vector de $\mathbb{R}^n$.

   Ej: $<< x_1, x_2, x_3>, <y_1, y_2, y_3>> \rightarrow \sum_{i=1}^{3} (x_i - y_i)^2$

7. Dados dos vectores de un espacio n-dimensional, definir una función que determine si ambos vectores tienen al menos una componente en coincidencia.

8. Definir el producto de un escalar por una matriz.

9. Dada una matriz de números enteros, definir una función que obtenga la sumatoria de los números mayores que 0 de las columnas pares.

10. Definir una función *selector por izquierda* para arreglos de *n* dimensiones.
Ej: < <3, 2>, < <A, B, C>, <D, E, F>, <G, H, I> > > → <H>

11. Dados dos vectores *n*-dimensionales, obtener el vector suma (sin recursividad).

12. Dado un número *n*, generar la siguiente secuencia (sin recursividad):
<<1>, <1, 2>, <1, 2, 3>, <1, 2, 3, 4>, ... <1, 2, 3, 4, ... *n*>>

13. Dada una secuencia con dos elementos, donde el primero es un átomo o secuencia y el segundo es un número, obtener una secuencia que contenga el primer elemento tantas veces como indica el número.
Ej: < a, 4 > → < a, a, a, a >

    Utilizando la función anterior, escribir una función <u>no recursiva</u> que aplicada a un número *n* devuelva una matriz de $n \times n$ de la siguiente forma:
    Si *n*=4 → < <1, 2, 3, 4>, <1, 2, 3, 4>, <1, 2, 3, 4>, <1, 2, 3, 4> >

14. Dada una secuencia de pares ordenados donde la primera componente indica el equipo que resultó ganador y la segunda indica el perdedor y donde cada par ordenado indica un partido jugado (no hay empates) obtener:

    a) Los equipos invictos.
    b) Los que siempre perdieron.
    c) Los que ganaron más veces de las que perdieron.
    d) Los que perdieron más veces de las que ganaron.
    e) Los que perdieron y ganaron la misma cantidad de veces.