Cátedra de Lógica y Algoritmos
Licenciatura en Ciencias de la Computación
Universidad Nacional de Rosario.

Listas y Funciones de Listas
(Una propuesta de introducción de modelos de cálculo).

Resumen :
Sobre una idea desarrollada por G. Jacopini (Universidad de Roma), se introduce un modelo adicional a los clásicos de funciones recursivas y máquinas de Turing.
Este modelo, no común en la literatura, es de una enorme potencia didáctica.
En su aparente sencillez permite familiarizarse rápidamente con conceptos de gran profundidad. El presente es la base estructural de un apunte de cátedra mas completo.
Los alumnos llegan a esta parte de la materia con un manejo del concepto clásico de funciones recursivas.
El resultado principal que se ofrece es que este nuevo modelo "al menos " incluye al de las funciones recursivas.
El tema que sigue (que se esboza brevemente al final de este trabajo) es la presentación de las máquinas de Turing y la demostración de la capacidad de estas de "contemplar" entre sus posibilidades las funciones recursivas de listas.
La clásica demostración de que las máquinas de Turing pueden ser "representadas" por las funciones recursivas, cierra en belleza la presentación de estos tres modelos de cálculo, mostrando su equivalencia profunda.

Raúl Eduardo Kantor
Profesor Asociado
Lógica y Algoritmos.
Facultad de Ciencias Exactas e Ingeniería
Universidad Nacional de Rosario.

Agradecimiento : Deseo agradecer especialmente a la Lic. Ana Casali y al Sr. Mariano Suarez Alvarez por su colaboración en la redacción del presente trabajo.

Listas y Funciones de Listas

1. LISTAS

Las funciones recursivas definidas hasta ahora tenían como resultado un numero entero no negativo y como dominio k-uplas de números con k fijo y determinado.

$$f^{(k)}: N^k{}_o \longrightarrow N_o$$

En lo que sigue nuestro objeto de estudio serán funciones definidas sobre sucesiones finitas de números enteros no negativos y con resultados en el mismo conjunto.

Formalmente, presentaremos las siguientes definiciones y notaciones :

Una <u>lista</u> es una secuencia finita ordenada de cero o más elementos de N_o. (1.1)

Notaciones : La forma mas general de notación de una lista será explicitando sus elementos entre los símbolos $<$ y $>$.

$$< x_1, x_2 , ..., x_n >$$

La lista sin elementos, <u>lista vacia</u>, será notada en la forma $<>$. (1.2)

Cuando no estemos interesados en explicitar sus elementos indicaremos las listas con letras mayúsculas: **X**, **Y**.
A veces queremos distinguir elementos particulares de una lista para lo cual usaremos una notación combinada, así la lista **x**,**Y** indica una lista cuyo primer elemento es **x**, la notación **Y**,**z** representa una lista cuyo último elemento es **z**, etc..
El conjunto de todas las listas será notado con ***L***.
Llamamos <u>longitud</u> de una lista a la cantidad de elementos que posee. (1.3)
A veces notaremos X^k para indicar una lista de k elementos.
Para cada elemento **n** de N_o notaremos L^n al conjunto de todas las listas de longitud exactamente n. Notaremos $L^{>n+1}$ al conjunto de todas las listas que tienen al menos n elementos.

Definiremos en ***L*** la siguiente operación :

Dadas dos listas $X =< x_1, x_2 , ..., x_n >$, $Y = < y_1, y_2 , ..., y_n >$,llamamos <u>concatenación</u> de **X** e **Y** , que notaremos **XY**, a la lista:

$$XY = < x_1, x_2 , ..., x_n , y_1, y_2 , ..., y_n > \qquad (1.4)$$

obtenida por yuxtaposición de las listas originales.
A veces notamos **XY** como **X,Y**

Ejercicio :
Mostrar que ***L*** dotado de la operación concatenación resulta un semigrupo.

2. Funciones básicas y constructores

Consideraremos a continuación funciones que van de L en L, esto es funciones que asocian a una lista de n-números X una lista Y (no necesariamente del mismo largo)
En general una función de lista f, es parcial, esto es definida solo en un subconjunto Df de todas las listas.

Veamos algunos ejemplos:

2.1 Ejemplos - Funciones básicas:

Definiremos las siguientes funciones que llamaremos <u>básicas</u> :

Función Borra a Izquierda (Notación b_I)

b_I : y,X ——> X (Borra el elemento izquierdo de la lista) **(2.1.1)**

Función Borra a Derecha (Notación b_d)

b_d : X,y ——> X (Borra el elemento derecho de la lista) **(2.1.2)**

Función Cero a Izquierda (Notación 0_I)

0_I : X ——>0, X (Agrega un cero a la izquierda de la lista) **(2.1.3)**

Función Cero a Derecha (Notación 0_d)

0_d : X ——> X,0 (Agrega un cero a la derecha de la lista) **(2.1.4)**

Función Sucesor a Izquierda (Notación s_I)

s_I : y,X ——>y+1, X (Suma uno al elemento izquierdo de la lista) **(2.1.5)**

Función Sucesor a Derecha (Notación s_d)

s_d : X,y ——> X,y+1 (Suma uno al elemento derecho de la lista) **(2.1.6)**

Observemos que las funciones 0_I y 0_d son funciones totales, mientras que para las restantes el dominio es L - < > (Todas las listas menos la vacía)

Ejemplos :

1. 0_d < 3,2,4> = <3,2,4,0>
2 s_d < 3,2,4> = <3,2,5>
3. b_d < 3,2,4> = <3,2>

2.3 Operadores:

Definiremos a continuación dos operadores entre funciones de listas, a partir de los cuales construiremos (como lo hicimos en las funciones recursivas) nuevas funciones de lista.

2.3.1 Composición :

Dadas dos funciones de listas ***f,g*** definimos la <u>composición</u> de ***f*** con ***g*** (notaremos ***fg***) una nueva función ***h*** que se obtiene aplicando primero la función ***f*** y luego la función ***g***.

$$X \xrightarrow{f} Y \xrightarrow{g} Z \qquad\qquad fg\ X \longrightarrow Z$$

Cuando tanto ***f*** como ***g*** son funciones totales, ***fg*** resulta una función total, ya que en ese caso $g(f(X))$ siempre está definido; cuando éste no es el caso, se tiene que una lista **X** pertenece al dominio de la composición ***fg*** si ***X*** pertenece al ***Df*** y ***f(X)*** pertenece al ***Dg***.

En particular, el dominio de ***fg*** puede resultar vacío.

Ejemplos :

1. $s_d\, 0_d$ **< 3,2,4> = <3,2,5,0>**
2. $0_d\, s_d$ **< 3,2,4> = <3,2,4,1>**
3. $0_d\, b_d$ **< 3,2,4> = <3,2,4>**

Es sencillo mostrar que la composición de funciones parciales es asociativa.
Esto nos permite escribir la composición de una función ***f*** consigo misma **k** veces.
Esto lo notamos como f^k.
Como es usual se conviene en notar f^0 como la aplicación identidad

Por ejemplo, $s_d^{\,k}$ tiene dominio en $L^{>0}$ y actúa de la siguiente manera :

$$s_d^{\,k} < x_1, x_2, \ldots, x_n > = < x_1 + k, x_2, \ldots, x_n >$$

Ejercicio:
Construir, a partir de las funciones elementales y de la composición, una función tal que su dominio sea exactamente $L^{>2}$ (Listas de largo mayor o igual a dos) sin importar que valores allí tome.

2.3.2 Repetición

La repetición es una forma de construir una nueva función de listas a partir de la repetida aplicación de una función dada.
Podemos decir que cumple un rol análogo al de la construcción **mientras** (while) en los lenguajes procedurales.

Dada una función ***f*** que pertenece a $L^{>1}$ de listas definimos la <u>repetición</u> de ***f***, notada ***(f)***, a una nueva función ***h*** definida de la siguiente manera :

$h\langle x,X,y\rangle = \langle x,X,y\rangle$	*si sucede que* x es igual a y
$h\langle x,X,y\rangle = h[f\langle x,X,y\rangle]$	*si sucede que* x es distinto de y

donde X es una lista, eventualmente vacía.
En otras palabras, la función ***(f)*** actúa aplicando tantas veces ***f*** hasta que el primer y el último elemento de su argumento son iguales.(Lo que no se puede afirmar que suceda necesariamente ...)

Ejemplos :

$(s_d)\langle 3,2,1\rangle = (s_d)\,[s_d\,\langle 3,2,1\rangle] = (s_d)\,\langle 3,2,2\rangle = (s_d)[s_d\,\langle 3,2,2\rangle] = \; s_d)\,\langle 3,2,3\rangle = \langle 3,2,3\rangle$

Observemos que la función (s_d) <u>no esta definida</u> sobre cualquier k-upla tal que su primer elemento sea menor que el último, ya que la aplicación sobre un tal argumento hace que el ciclo de aplicación se repita indefinidamente.

3. Funciones Recursivas de Listas

Ahora estamos en condiciones de definir la <u>clase de las funciones recursivas de listas</u> **FRL**:

Las definiremos de la siguiente manera :

3.1 Las funciones básicas 0_i, 0_d, s_d, s_i, b_d, b_i son **FRL**.

3.2 Si las funciones ***f, g*** son **FRL** , entonces ***fg*** es **FRL**.

3.3 Si la función ***f*** es **FRL** , entonces ***(f)*** es **FRL**.

O sea son funciones recursivas de listas las funciones básicas definidas en 2.1 y las que se obtienen de estas aplicando un numero finito de veces las reglas 3.2 y 3.3 .

Ejemplos :

Sean las siguientes funciones de lista que son **FRL**

1. **—>** $= 0_d\,(s_d)\,b_i$

2. **<—** $= 0_i\,(s_i)\,b_d$

3. $\mathbf{D_i} = 0_d\,(s_d)$ **<—**

4. $\mathbf{D_d} = 0_i\,(s_i)$ **—>**

Se puede ver que aplicadas producen los siguientes efectos :

—> : y, X ——> X, y	**Pasa un elemento del extremo izquierdo al derecho.**
<— : X, y ——> y, X	**Pasa un elemento del extremo derecho al izquierdo.**
$\mathbf{D_i}$: y, X ——>y, y, X	**Duplica un elemento a izquierda.**
$\mathbf{D_d}$: X, y ——> X, y, y	**Duplica un elemento a derecha.**

Ejercicios :

1. Verificar las funciones del ejemplo anterior.

2. Construir la función **<—>** que realiza lo siguiente :

<—> : y , X, z ——> z, X, y (No es trivial...)

4. Representación de las funciones recursivas parciales en FRL

El objetivo de esta sección es mostrar que las funciones de lista son una herramienta de cálculo por lo menos tan poderosa como las funciones recursivas primitivas.
Probaremos que cualquier función recursiva puede ser calculada por una función de lista.
Para ello comenzaremos con la siguiente definición :

4.1 Definición :

Dada una $f^{(k)}$, con dominio Df contenido en N^k_o , definida f: $N^k_o \longrightarrow N_o$ diremos que una función de lista que notaremos A_f la <u>representa</u> si cumple con lo siguiente :
Para toda X, lista con k elementos, que pertenece al dominio de f,
Sea Y una lista cualquiera.
Entonces aplicando A_f a la lista X,Y se obtiene la lista : $f(X),X,Y$ (4. 1)

X en Df entonces A_f : X,Y ——> $f(X),X,Y$

Observemos que si A_f representa a f, entonces no solo es posible el cálculo con funciones de lista sino que, a diferencia de f, al operar con calcular A_f no perdemos los valores de los argumentos.

4.2 Teorema :

Para toda $f^{(k)}$ función recursiva parcial f: $N^k_o \longrightarrow N_o$ existe una función recursiva de lista asociada, que notaremos A_f tal que la representa.

Para probar este teorema aprovecharemos la definición precisamente "recursiva" de las funciones recursivas parciales.

Mostraremos primero :

4.2.1 Las funciones bases tienen representación en las funciones recursivas de lista.

Recordemos que las funciones base de las funciones recursivas eran :

$z^{(n)}$ funciones cero	$z(X) = 0$ para todo X
$S^{(1)}$ función sucesor	$S(x)= x+1$
$U^{(n)}_k$ funciones proyección	$U^{(n)}_k < x_1, x_2 , .x_k ,..., x_n >= x_k$

Efectivamente las funciones :

$A_z = 0_l$ representa a las funciones cero.

$A_s = D_l s_l$ representa a la función sucesor.

A_U = —>—>—>...k - 1 veces —>—> D_l <—><—....k-1 veces ... <—><—

Verificar.

4.2.2 Sea f función definida por composición , o sea :

$$f = \S\, [\, h, g_1 , g_2 , \dots g_m] \qquad f(X) = h(g_1(X), g_2(X), \dots g_m(X))$$

Si A_h, A_{g1}, A_{g2}, ... A_{gm} son **FRL** entonces existe A_f que representa a f y es **FRL**

La demostración es constructiva :

A_f = A_{g1} —> A_{g2}—> ... —> A_{gm} <— <—...m - 1 veces ..<— A_h —> b_i y... m veces ... b_i <—

Veamos como funciona aplicando por partes :

Dada X,Y aplicando A_{g1} —> A_{g2}—> ... —> A_{gm} <— <—...m - 1 veces... <— queda entonces

$g_1(X)$, $g_2(X)$, ...$g_m(X)$, X , Y aplicando ahora A_h —> b_i y... m veces ... b_i <—

queda f(X),X,Y .

4.2.3 Sea la función f definida por recursión, o sea f = P [g , h] , tal que g y h están representadas por las funciones de listas A_h, A_g que son FRL.
Entonces existe A_f que es FRL y que representa a f.
Aquí también la demostración es constructiva :

En efecto

A_f = —> A_g 0_i. (—><—><— A_h —>b_i. S_i <—><—) b_i. <—><—

Veamos que efectivamente cumple :
a y, X, Y le aplicamos primero —> A_g 0_i obteniendo 0, g(X), X, Y, y

Aplicamos ahora repetidamente la parte interna del operador (..)
Observemos que si y = 0 , entonces no corresponde operar ninguna vez

Si a t, f(t,X), X, Y, y le aplicamos —><—><— A_h —>b_i. S_i <—><— obtenemos :

t+1, f(t+1, X), X, Y, y . Reiteramos esta aplicación hasta llegar a y, f(y, X), X, Y, y.

Allí entonces aplicamos : b_i. <—><— para obtener entonces f(y, X), y,X, Y .

Por lo que entonces A_f representa a f.

4.2.4 Sea finalmente función f definida por minimización de la función g . (Donde se cumple la condición de regularidad)
O sea f = M [g] , tal que g esta representada por las función de lista A_g que es FRL.
Entonces existe A_f que es FRL y que representa a f.
Nuevamente la demostración es constructiva :
A_f = 0_i. A_g 0_d (b_i. S_i A_g) b_i. b_d
Ejercicio : Verificar que efectivamente A_f representa a f.

Dada la manera en que se definieron las funciones recursivas, surge inmediatamente de las demostraciones **4.2.1, 4.2.2, 4.2.3** y **4.2.4,** que toda función de ese tipo tiene un representante en las funciones recursivas de listas, .

En ese sentido se puede afirmar que las funciones recursivas parciales son particulares casos de las funciones recursivas de listas . **FRP** están contenidas en **FRL.**

==

Con este resultado termina esta parte de la materia. En las clases siguientes se introducen las máquinas de Turing.
Es relativamente sencillo asociar a cada lista de enteros no negativos, una particular secuencia de la cinta de una máquina de Turing.
(En la máquina de dos símbolos |,□ basta asociar a la lista $< x_1, x_2, ..., x_n >$ la sucesión de $x_1 + 1$ | , $x_2 + 1$ | , .etc. Separados por un □.)
Se puede mostrar que a toda función de lista se le puede asociar una adecuada máquina de Turing.
Las funciones de lista básicas son de representación trivial.
La composición coincide con la composición de máquinas.
Finalmente la repetición se realiza con la composición entre la máquina que representa la función a repetir y una máquina que verifica si los extremos de la secuencia coinciden.
(Tal vez sea muy sintético, pero la idea es trasmitir la potencia didáctica del recurso)

==

Algunos ejercicios finales :

Construir funciones de lista *f* tales que

1. $F < x+1, Y> = < x, Y>$
2. $F < x,y,Y> = < x-y, Y>$ si $x >= y$
3. $F < x,y,Z> = < Z, y,y,....x$ *veces* $...y,y>$
4. $F < x,Y> = <1,2,3,...x-1 ,x, Y>$
5. $F < x,y,Y> = < xy, Y>$
6. $F < x,Y> = < 1 + 2x + x^2, Y>$
7. $F < x,Y> = < z, Y>$ si $x = 2z$, *indefinida en otro caso.*
8. $F < x_1, x_2 , ..., x_n > = < n >$ *para nada trivial...*

Cual es el dominio de las siguientes funciones de lista :

1. $0_d (s_d^2)$
2. $0_d (<\!\!-\!\!-b_d\!\!-\!\!-\!\!>)$