Práctico 2: Especificación, Derivación y Verificación de Programas Funcionales

Algoritmos y Estructuras de Datos I
2$^{\text{do}}$ cuatrimestre 2016

Esta guía tiene como objetivo obtener las habilidades necesarias para llevar adelante un proceso de derivación o verificación de programas recursivos a partir de especificaciones formales.

Completan esta guía ejercicios algunos ejercicios para demostrar tanto por inducción sobre naturales, como por inducción ***estructural*** *sobre listas. Se busca mejorar la habilidad para utilizar esta técnica de prueba que es central para la tarea de cálculo de programas recursivos.*

Los ejercicios de cálculo de programas tienen una dificultad creciente: en los primeros, la derivación o verificación se obtiene de manera directa *a través de una demostración inductiva. Los ejercicios sucesivos son más complejos y requieren el uso de técnicas más avanzadas: modularización y generalización.*

1. Dado el **programa**

$$\begin{array}{|l} sum : [Num] \to Num \\ \hline sum.[\,] \doteq 0 \\ sum.(x \triangleright xs) \doteq x + sum.xs \end{array}$$

   a) ¿Qué hace esta función? Escriba en lenguaje natural el **problema** que resuelve.

   b) Escriba una **especificación** de la función con una expresión cuantificada.

   c) **Verifique** que esta especificación vale para toda lista xs.

   d) En la vida real, ¿en qué orden se realiza la implementación del **programa**, el enunciado del **problema**, la redacción de la **especificación** y el proceso de **verificación**?

   e) Ahora **derive** la definición de la función a partir de su especificación.
   ¿Esta derivación es parecida a la demostración en el punto 1*c*?

2. A partir de las siguientes especificaciones, dar el tipo de cada función y *derivar* las soluciones algorítmicas correspondientes.

   a) $sum\_cuad.xs = \langle \sum i \,:\, 0 \le i < \#xs \,:\, xs.i * xs.i \rangle$

   b) $iga.e.xs = \langle\, \forall\, i \,:0 \le i < \#xs :\; xs.i = e \,\rangle$

   c) $exp.x.n = x^n$

   d) $sum\_par.n = \langle \sum i \,:\, 0 \le i \le n \wedge par.i \,:\, i \rangle$

   e) $cuántos.p.xs = \langle \mathrm{N}\ i \,:\, 0 \le i < \#xs \,:\, p.(xs.i) \rangle$

   f) $busca.e.xs = \langle \mathrm{Min}\ \ i \,:0 \le i < \#xs \wedge xs.i = e :\; i \,\rangle$

3. Para todos los ítems del ejercicio anterior, dar un ejemplo de uso de la función, es decir: elegir valores concretos para los parámetros y calcular el resultado usando la solución algorítmica obtenida. Las listas deben tener por lo menos tres elementos.

4. Expresar en lenguaje natural cada una de las siguientes sentencias formales, donde xs y ys son listas de enteros:

   a) $\langle \exists x \,:\, x \in Num \,:\, \langle \forall y \,:\, y \in Num \,:\, x \le y \rangle \rangle$

   b) $\langle \forall y \,:\, y \in Num \,:\, \langle \exists x \,:\, x \in Num \,:\, x \le y \rangle \rangle$ ¿Cuál es la diferencia con la anterior?

   c) $\langle \forall x, z \,:\, x \in Num \wedge z \in Num \wedge x < z \,:\, \langle \exists y \,:\, y \in Num \,:\, x < y < z \rangle \rangle$

   d) $\langle \exists i \,:\, 0 \le i < \#xs \,:\, xs.i = 0 \rangle$

e) $\langle \forall i \,:\, 0 \leq i < \#xs \,:\, xs.i \geq 0 \rangle$

f) $\langle \exists i \,:\, 0 < i < \#xs \,:\, xs.(i-1) > xs.i \rangle$

g) $\#xs \neq \#ys \vee \langle \exists i \,:\, 0 \leq i < \#xs \wedge 0 \leq i < \#ys \,:\, xs.i \neq ys.i \rangle$

5. Derivar las siguientes funciones.

a) $f : Num \to Nat \to Num$ computa la suma de potencias de un número, esto es

$$f.x.n = \langle \sum i \,:\, 0 \leq i < n :\; x^i \,\rangle \quad .$$

b) $pi : Nat \to Num$ computa la aproximación del número π

$$pi.n = 4 * \langle \sum i \,:\, 0 \leq i < n :\; (-1)^i/(2*i+1) \,\rangle$$

Ayuda: Modularizar dos veces. La segunda con la función *exp* del ejercicio 2*c*

c) $f : Nat \to Nat$ computa el cubo de un número natural x utilizando únicamente sumas. La especificación es muy simple: $f.x = x^3$.

Ayuda: Usar inducción y modularización varias veces

d) $f.xs = \langle \exists i \,:\, 0 < i \leq \#xs :\; \langle \prod j \,:\, 0 \leq j < \#(xs{\downarrow}i) :\; (xs{\downarrow}i).j \,\rangle = xs.(i-1) \,\rangle$

6. Dada la definición recursiva de la función

$$\begin{array}{|l} f : Nat \to Nat \\ \hline f.0 \doteq 0 \\ f.1 \doteq 1 \\ f.(n+2) \doteq f.n + f.(n+1) \end{array}$$

demostrar que la siguiente propiedad

$$P.n : f.n < 2^n$$

vale para todo $n \in Nat$.

7. Especificar formalmente utilizando cuantificadores cada una de las siguientes funciones descriptas informalmente. Luego, *derivar* soluciones algorítmicas para cada una.

a) $iguales : [A] \to Bool$, que determina si los elementos de una lista de tipo A son todos iguales entre sí. Suponga que el operador $=$ es la igualdad para el tipo A.

b) $minimo : [Int] \to Int$, que calcula el mínimo elemento de una lista **no vacía** de enteros.

Nota: La función no debe devolver $\pm\infty$.

c) $creciente : [Int] \to Bool$, que determina si los elementos de una lista de enteros están ordenados en forma creciente.

d) $prod : [Num] \to [Num] \to Num$, que calcula el producto entre pares de elementos en iguales posiciones de las listas y suma estos resultados (producto punto). Si las listas tienen distinto tamaño se opera hasta la última posición de las más chica.

8. A partir de las siguientes especificaciones expresar en lenguaje natural qué devuelven las funciones, agregarles su tipo y derivarlas:

a) $psum.xs = \langle \forall i \,:\, 0 \leq i \leq \#xs :\; sum.(xs{\uparrow}i) \geq 0 \,\rangle$, con *sum* la función del ejercicio 1.

b) $sum\_ant.xs = \langle \exists i \,:\, 0 \leq i < \#xs :\; xs.i = sum.(xs{\uparrow}i) \,\rangle$

c) $sum8.xs = \langle \exists i \,:\, 0 \leq i \leq \#xs :\; sum.(xs{\uparrow}i) = 8 \,\rangle$.

d) $f.xs = \langle \text{Max } i \,:\, 0 \leq i < \#xs \wedge sum.(xs{\uparrow}i) = sum.(xs{\downarrow}i) :\; i \,\rangle$.

9. Especificar formalmente utilizando cuantificadores cada una de las siguientes funciones descriptas informalmente. Luego, *derivar* soluciones algorítmicas para cada una.

a) $cuad : Nat \to Bool$, que dado un natural determina si es el cuadrado de un número.

b) $n8 : [Num] \to Nat$, que cuenta la cantidad de segmentos iniciales de una lista cuya suma es igual a 8.

10. Expresar utilizando cuantificadores las siguientes sentencias del lenguaje natural:

    a) La lista xs es un segmento inicial de la lista ys.

    b) La lista xs es un segmento de la lista ys.

    c) La lista xs es un segmento final de la lista ys.

    d) Las listas xs y ys tienen en común un segmento.

    e) La lista xs de numeros enteros tienen la misma cantidad de elementos pares e impares.

    f) La lista xs posee un segmento **no** inicial y **no** final cuyos valores son mayores a los valores del resto de la misma.

11. Derivar funciones recursivas a partir de cada una de las especificaciones que escribiste para los ejercicios 10*a* y 10*b*.

12. Derivar las funciones especificadas como:

    a) $sumin.xs = \langle \text{Min } as, bs, cs \ : xs = as +\!\!+ bs +\!\!+ cs : \ sum.bs \rangle$
    (suma mínima de un segmento).

    b) $f.xs = \langle \text{N } as, bs, cs \ : xs = as +\!\!+ bs +\!\!+ cs : \ 8 = sum.bs \rangle$
    (cantidad de segmentos que suman eso).

    c) $maxiga.e.xs = \langle \text{Max } \ as, bs, cs \ : xs = as +\!\!+ bs +\!\!+ cs \wedge iga.e.bs : \ \#bs \rangle$
    (máxima longitud de iguales a e)
    donde iga es la función del ejercicio 2*b*.

13. Describir en lenguaje natural y dar el *tipo* de cada una de las siguientes funciones especificadas formalmente:

    a) $f.xs \doteq \langle \text{N } i, j \ : \ 0 \leq i < j < \#xs \ : \ xs.i = 0 \wedge xs.j = 0 \rangle$

    b) $g.xs \doteq \langle \text{Max } p, q \ : \ 0 \leq p < q < \#xs \ : \ xs.p + xs.q \rangle$

    c) $h.xs \doteq \langle \text{N } k \ : \ 0 \leq k < \#xs \ : \ \langle \forall i \ : \ 0 \leq i < k \ : \ xs.i < xs.k \rangle \rangle$

    d) $k.xs \doteq \langle \forall i, j \ : \ 0 \leq i \wedge 0 \leq j \wedge i + j = \#xs - 1 \ : \ xs.i = xs.j \rangle$

    e) $l.xs \doteq \langle \text{Max } p, q \ : \ 0 \leq p \leq q < \#xs \wedge \langle \forall i \ : \ p \leq i < q \ : \ xs.i \geq 0 \rangle \ : \ q - p \rangle$

14. Derivar la función definida en el ejercicio 13*b*.

15. Derivar una definición recursiva para la función $f : [Num] \to [Num] \to Num$, que calcula la mínima distancia entre valores de las listas xs y ys, y cuya especificación es la siguiente:

$$f.xs.ys = \langle \text{Min } i, j \ : \ 0 \leq i < \#xs \wedge 0 \leq j < \#ys \ : \ |xs.i - ys.j| \rangle$$

16. Derivar una definición recursiva para la función $g : [Char] \to [Char] \to Bool$, especificada como sigue:

$$g.xs.ys = \langle \exists as, bs, c, cs \ : \ xs = as +\!\!+ bs \wedge ys = as +\!\!+ (c \triangleright cs) \ : \ bs = [\,] \vee bs.0 < c \rangle$$

    Decir en palabras cuándo $g.xs.ys$ es $True$.