12/08/14

- Algoritmos I -

Franco Luque   OF. 399   Libro: Cálculo de progr
Matias "Chun" Lee

* 2 parciales   * Laboratorio → 5 proyectos.

http://cs.famaf.unc.edu.ar/wiki

Problema - informal
- poco detallado

↓

Programa - formal
- muy detallado.

- Crisis del software → lleva a la especificación de un programa
  ↓
  Lenguaje formar p/ especificar un problema.

Nuevo esquema:

Problema - Informal

↓

Especificación formal - Formal
- poco detallada → falta de datos específicos

↓

Programa - Formal
- muy detallado } → un algoritmo cumple con det. especificaciones

Derivar el programa, a través de especificación formal

Repaso Cálculo Proposicional

A12: Regla Dorada
A9: Idemp. de ∨
A2: conmut. ≡

1 a) $p \wedge p \equiv p$

$p \wedge p$

$\equiv \{A12 (P, Q := p, p)\}$

$p \equiv p \equiv p \vee p$

$\equiv \{A2 (P, Q := p, p \vee p)\}$

$p \equiv p \vee p \equiv p$

$\equiv \{A9 (P := p)\}$

$p \equiv True$

$\equiv \{A3 (P := p)\}$

p #

$p \wedge p \equiv p$

$\equiv$ { R. Dorada }

$p \equiv p \vee p$

$\equiv$ {conmut. e Idemp.}

True

#

CUANTIFICADORES

todosPares.xs =

$$\langle \forall i : 0 \leq i < \#xs : espar.(xs, i) \rangle$$

Tipos

espar :: Num → Bool
todospares :: [Num] → Bool

todospares . [2,4,7] = espar.(xs,0) ∧ espar.(xs.1) ∧ espar.(xs.2).

operador.	cuantificador	
$\wedge$	$\forall$	[] ≡ True
$\vee$	$\exists$	[] ≡ False
$+$	Σ	0
$*$	Π	1
$\cap$	$\bigcap$	Universal
$\cup$	$\bigcup$	$\emptyset$
min	Min	∞
max	Max	$-\infty$

sea ⊕ un operador

$\oplus : A \to A \to A$

⊕ asociativo

⊕ conmutativo

$$\langle \underbrace{\oplus i}_{\text{variable cuantificada}} : \overbrace{R.i}^{\text{rango}} : \underbrace{T.i}_{\text{término}} \rangle$$

Intuición cuantificadora

1. Calcular valores $i_1, \ldots, i_n$ que satisfacen $R.i$
2. Aplicar $T.i$ a $i_1, \ldots, i_n$.
3. Juntar todo con operador ⊕

$$T.i_1 \oplus T.i_2 \oplus T.i_3 \oplus \ldots \oplus T.i_n$$

Alto Orden

Función

③ a) $ParaTodo :: [A] \to \underbrace{(A \to Bool)} \to Bool$

$paraTodo.R.T \equiv \langle \forall i :: i \in R : T.i \rangle$ a donde $\in :: A \to [A] \to$

$i \in [\,] \equiv Fals$

$i \in (x \triangleright xs) \equiv (i = x$

$i \in$

$paraTodo.[\,].T \equiv True$

$paratodo.(x \triangleright xs).T \equiv T.x \wedge paraTodo.xs.T$

Especificar y definir

$sumacuadrados :: Int \to Int$

Ej: sumacuadrados. 3 = $0^2 + 1^2 + 2^2 + 3^2$

$sumacuadrados.n =$

$\langle \sum i : 0 \leq i \leq n : i^2 \rangle \equiv P.n$

$sumacuados.0 = 0$

$suma cuadrados.(n+1) = (n+1)^2 + sumacuadrados.n$

suponemos $P.n$ y consideremos $sumacuadrado.(n+1)$

$sumacuadrado.(n+1)$

$\equiv$ {especif}

$\langle \sum i : 0 \leq i \leq n+1 : i^2 \rangle$

$\equiv$ {reescribir rango}

$\langle \sum i : 0 \leq i \leq n \vee i = n+1 : i^2 \rangle$

$\equiv$ {Partición rango} → solo se pueden hacer si son disjuntos. (se sumaría varias veces)

$\langle \sum i : 0 \leq i \leq n : i^2 \rangle + \langle \sum i : i = n+1 : i^2 \rangle$

$\equiv$ {HI}

$sumacuadrado.n + \langle \sum i : i = n+1 : i^2 \rangle$

$\equiv$ {Rango Unitario}

$sumacuadrado.n + (n+1)^2$

#

(3) b) existe [].T ≡ False
existe (x▷xs).T ≡ T.x ∨ existe.xs.T

c) sumatoria [].T = 0
sumatoria (x▷xs).T = T.x + sumatoria.xs.T

d) productoria [].T = 1
productoria (x▷xs).T = T.x * productoria.xs.T

Existe:
(4) cuantGen ⊕.z.[].T = z

cuantGen ⊕.z.(x▷xs).T = T.x ⊕ cuantGen ⊕.z.xs.T

Sumatoria:
sumatoria R.T = cuantGen +.0.R.T

Productoria:
productoria R.T = cuantGen *.1.R.T

(5) (3) b) existe.[].T

≡ {especificación}

$\langle \exists i : i \in [\,] : T.i \rangle$

≡ {Def. recursiva ∈}

$\langle \exists i : False : T.i \rangle$

≡ {Rango vacío para ∃, neutro ∨ es False}

False.

#

existe [N].T

≡ {constructor ▷}

existe.(N▷[]).T

≡ {Desplegando función caso ind}

T.N ∨ existe.[].T

≡ {Desplegando función (caso base)}

T.N ∨ False

≡ {Neutro disyunción}

T.N

#

Sumatoria [1, 2, 3] $\frac{1+2}{1+0}$

sumatoria . [ ] . T
$\equiv$ {especificación}
$\langle \sum i : i \in [\,] : T.i \rangle$
$\equiv$ {Def. rec. $\in$}
$\langle \sum i : False : T.i \rangle$
$\equiv$ {R. V sumat.}
0 #

sumatoria [N] . T
$\equiv$ {constructor $\triangleright$}
sumatoria (N $\triangleright$ [ ]) . T
$\equiv$ {Desplegando función (caso inductivo
T.N + sumatoria [ ] . T
$\equiv$ {Desplegando función (caso base
TN + 0
$\equiv$ {Neutro sumatoria}
TN #

productoria . [ ] . T
$\equiv$ {especificación}
$\langle \prod i : i \in [\,] : T.i \rangle$
$\equiv$ {Def. recursiva $\in$}
$\langle \prod i : False : T.i \rangle$
$\equiv$ {R. V productoria}
1 #

productoria [N] . T
$\equiv$ {constructor $\triangleright$}
productoria (N $\triangleright$ [ ]) . T
$\equiv$ {Desplegando función (caso
TN $*$ productoria [ ] . T
$\equiv$ {Despleg. función (caso base
TN $*$ 1
$\equiv$ {Neutro productoria}
TN. #

cuantGen . $\oplus$ z [ ] . T
$\equiv$ {especificación}
$\langle \oplus i : i \in [\,] : T.i \rangle$
$\equiv$ {Def recursiva $\in$}
$\langle \oplus i : False : T.i \rangle$
$\equiv$ {R. V cuant. Gen}
z

cuantGen . $\oplus$ z . [N] . T
$\equiv$ {constructor $\triangleright$}
cuantGen $\oplus$ z (N $\triangleright$ [ ]) . T
$\equiv$ {Despleg. función (caso ind)}
TN $\oplus$ cuant. Gen $\oplus$ z [ ] . T
$\equiv$ {Despleg. función (caso base)}
TN $\oplus$ z
$\equiv$ {Neutro cuantGen}
TN #

7 $\langle \forall i : R.i : T.i \rangle \equiv \langle \forall i :: R.i \Rightarrow T.i \rangle$

$\langle \exists i : R.i : T.i \rangle \equiv \langle \exists i :: R.i \wedge T.i \rangle$

$\equiv$ {Def. de $\exists$}

$\neg \langle \forall i : R.i : \neg T.i \rangle$

$\equiv$ {Intercambio}

$\neg \langle \forall i :: R.i \Rightarrow \neg T.i \rangle$

$\equiv$ {caract. de $\Rightarrow$}

$\neg \langle \forall i :: \neg R.i \vee \neg T.i \rangle$

$\equiv$ {De Morgan}

$\neg \langle \forall i :: \neg (R.i \wedge T.i) \rangle$

$\equiv$ {Def. de $\exists$}

$\neg\neg \langle \exists i :: \neg\neg (R.i \wedge T.i) \rangle$

$\equiv$ {Doble Neg.}

$\langle \exists i :: R.i \wedge T.i \rangle$ #

14/08/14

- Repaso -
- $A \equiv B$
- Cuantificación Gral
  - operador $\oplus : A \to A \to A$
  - asociativo y conmutativo

$$\langle \oplus i : R.i : T.i \rangle$$

- intuición: $i_1, i_2, \ldots, i_n$ → computamos los valores posibles

$$\begin{array}{cccc} i_1, & i_2, & \ldots, & i_n \\ \downarrow & \downarrow & & \\ T.i_1 & T.i_2 & \ldots & T.i_n \end{array}$$

$T.i_n$ → término

$T.i_1 \oplus T.i_2 \oplus \ldots \oplus T.i_n$ → aplicación de la operac

$$\sum_{i=0}^{n} i^2 = \langle \sum i : 0 \leq i \leq n : i^2 \rangle$$ → Mejor notación.

$$\underbrace{\text{espar}.i \wedge i^2 \leq n}_{R.i}$$

Reglas para la cuantificación general

A16: $\langle \oplus i : False : T.i \rangle = e$

donde e es el neutro de $\oplus$: $x \oplus e = x$.

Ejemplos:

$$\langle \prod i : \underbrace{i^2 \geq 5 \wedge |i| \leq 1}_{\text{no existen valores que cumplan: } False} : i! \rangle$$

$\equiv$ {"pasos"}

$$\langle \prod i : False : i! \rangle$$

$\equiv$ {Rango Vacío}

A17: $\langle \oplus i : i = N : T.i \rangle = T.N$ — Rango Unitario

$(i+2)$ ✗ → no se puede

predicados — término tiene el tipo del operador.

A18: $\langle \oplus i : \underbrace{R \vee S}_{\text{Disjuntos}} : T \rangle = \langle \oplus i : R : T \rangle \oplus \langle \oplus i : S : T \rangle$

Ejemplo: $\langle \sum i : 0 \leq i < 3 \vee 1 \leq i < 5 : i^2 \rangle$ $(=10)$ ¡No es lo mismo!

$$\neq \langle \sum i : \underbrace{0 \leq i < 3}_{0,1,2} : i \rangle + \langle \sum i : \underbrace{1 \leq i < 5}_{1,2,3,4} : i \rangle$$

Deben ser Disjuntos!

Requisito: $\underbrace{\oplus \text{ idempotentes}}_{(\forall x : x \oplus x = x)}$ y R y S disjuntos $(R \wedge S \equiv False)$

A20: $\langle \oplus i : R_i : T_i \oplus G_i \rangle = \langle \oplus i : R_i : T_i \rangle \oplus \langle \oplus i : R_i : G_i \rangle$

↳ Regla del término.

$$(T.i_1 \oplus G.i_1) \oplus (T.i_2 \oplus G.i_2) \oplus \ldots \oplus (T.i_n \oplus G.i_n)$$

$$= (T.i_1 \oplus T.i_2 \oplus \ldots \oplus T.i_n) \oplus (G.i_1 \oplus \ldots \oplus G.i_n).$$

A21: término constante

$\langle \oplus i : R.i : C \rangle = C$

R ↳ predicado, expresión booleana.

sumatoria y productoria el neutro va a ser idempotente

$i_1, i_2, \ldots, i_n$ — i no aparece en C

$C \oplus C \oplus \ldots \oplus C = C$

↳ operador es idempotente para C $(C \oplus C = C)$

- R vacío → no puede hacerse sino que se utiliza rango vacío, y devuelve el término

R debe ser <u>no</u> vacío $(\exists i : R.i) \rightarrow \exists i : R.i$

↳ $R \not\equiv False$.

A22: Distributividad

$$\langle \oplus i : R_i : x \otimes T_i \rangle = x \otimes \langle \oplus i : R_i : T.i \rangle$$

<u>Intuición:</u>

$$(C \otimes T.i_1) \oplus (C \otimes T.i_2) \oplus \ldots \oplus (C \otimes T.i_n)$$

$$C \otimes (T.i_1 \oplus \ldots + T.i_n)$$

- $\otimes$ distribuye a $\oplus$

↳ $\forall$ y $\exists$ → funciona ✓ (de los dos lados)

Si $\oplus$ es productoria y $\otimes$ sumatoria → no funcionaría

<u>Ejemplo:</u>

$(x \max y) + z = (x + z \max y + z$

- Si R es no vacío o el neutro $\oplus$ absorve $\otimes$.

$p \vee True \equiv True$ $(C \otimes e = e)$

$x \min -\infty = -\infty$
$x \max \infty = \infty$ → en casos especiales.

9 a) Distributividad de $\forall$ respecto a $\Rightarrow$:

$$\langle \forall i : R.i : Z \Rightarrow T.i \rangle \equiv Z \Rightarrow \langle \forall i : R.i : T.i \rangle$$

$\equiv$ { Caracterización $\Rightarrow$ }

$\langle \forall i : R.i : \neg Z \vee T.i \rangle$

$\equiv$ { Distributiva de $\vee$ con $\forall$ }

$\neg Z \vee \langle \forall i : R.i : T.i \rangle$

$\equiv$ { Caract $\Rightarrow$ }

$Z \Rightarrow \langle \forall i : R.i : T.i \rangle$

#

Rta: i no debe estar cuantificada en Z }

b) Instanciación de $\forall$

$$\langle \forall i :: T.i \rangle \Rightarrow T.x$$

Regla de instanc. de $\exists$:

$$T.x \Rightarrow \langle \exists i :: T.i \rangle$$

Demostración:

$\langle \forall i :: T.i \rangle \Rightarrow T.x$

$\equiv$ { Def. dual de $\Rightarrow$ }

$\langle \forall i :: T.i \rangle \wedge T.x \equiv \langle \forall i :: T.i \rangle$

$\equiv$ { Rango Unitario }

$\langle \forall i :: T.i \rangle \wedge \langle \forall i : i = x : T.i \rangle \equiv \langle \forall i :: T.i \rangle$

$\equiv$ { Partición de Rango }

$\langle \forall i : True \vee i = x : T.i \rangle \equiv \langle \forall i :: T.i \rangle$

$\equiv$ { Absorción }

$\langle \forall i :: T.i \rangle \equiv \langle \forall i :: T.i \rangle$ True #

$T.x \Rightarrow \langle \exists i :: T.i \rangle$

$\equiv$ {Def. de $\exists$}

$T.x \Rightarrow \neg \langle \forall i :: \neg T.i \rangle$

$\equiv$ {contrarrecíproca}

$\neg(\neg \langle \forall i :: \neg T.i \rangle) \Rightarrow \neg T.x$

$\equiv$ {Doble Neg}

$\langle \forall i :: \neg T.i \rangle \Rightarrow \neg T.x$

$\equiv$ {Sea $T'.i \equiv \neg T.i$}

$\langle \forall i :: T'.i \rangle \Rightarrow T'.x$

$\equiv$ {instanciación de $\forall$}

True.

#

9) c) Intercambio para $\forall$ (generalizada)

$\langle \forall i : R.i \wedge S.i : T.i \rangle \equiv \langle \forall i : R.i : S.i \Rightarrow T.i \rangle$

$\langle \forall i : R.i : S.i \Rightarrow T.i \rangle$

$\equiv$ {A24: Intercambio}

$\langle \forall i :: R.i \Rightarrow (S.i \Rightarrow T.i) \rangle$

$\equiv$ {currificación}

$\langle \forall i :: R.i \wedge S.i \Rightarrow T.i \rangle$

$\equiv$ {Intercambio $\forall$}

$\langle \forall i : R.i \wedge S.i : T.i \rangle$

$\equiv$ True

#

19/08/14

Partición de rango: $\langle \oplus i : R \vee S : T \rangle \equiv \langle \oplus i : R.i : T.i \rangle \oplus \langle \oplus i : S.i : T.i \rangle$

Condiciones:

- $\oplus$ es idempotente ó R y S son disjuntos
  ($\forall x,\ x \oplus x = x$) ($\nexists i : R.i \wedge S.i$)

A23. Anidado: $\langle \oplus i,j : R.i \wedge S.i.j : T.i.j \rangle \equiv$
$\langle \oplus i : R.i : \langle \oplus j : S.i.j : T.i.j \rangle \rangle$

i constante que viene de afuera.

$T!i$

Ejemplo: $R.i \equiv 7 \leq i \leq 10$
$S.i.j \equiv i \bmod j = 0$

$\langle \oplus i,j : 7 \leq i \leq 10 \wedge i \bmod j = 0 : T.i.j \rangle$

$(i,j) \in \{(7,1), (7,7), (8,1), (8,2), (8,4), (8,8), (9,1), (9,$
$(9,9)\}$

↳ Aplicamos el anidado, empezamos fijando i, para después computar j.

$\langle \oplus i : 7 \leq i < 10 : \langle \oplus j : i \bmod j = 0 \wedge j > 0 : T.i.j \rangle \rangle$

$i \in \{7, 8, 9\}$

* $i = 7$: $j \in \{1, 7\}$
* $i = 8$: $j \in \{1, 2, 4, 8\}$
* $i = 9$: $j \in \{1, 3, 9\}$

función que va de i a j.

T9: Cambio de Variable:

$$\langle \oplus i : R.i : T.i \rangle \equiv \langle \oplus j : R(f.j) : T.(f.j) \rangle$$

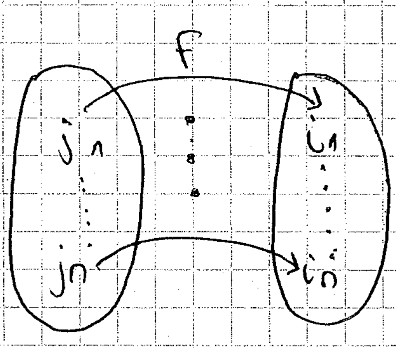
$i \in I$, $j \in J$, $f : J \to I$

$f(j_1)$ $f(j_2)$ $f(j_n)$

↳ i_1 i_2 ... i_n $(\in i)$

(resultado aplicar f.)

$T.i_1 \oplus T.i_2 \oplus \ldots \oplus T.i_n$



Condiciones:

- f tiene inversa en R
  ↳ en el conj. de valores que satisfacen a R (en este caso i)

- si no es inyectiva ⟹ se va a repetir un elemento, aunque sólo se podría cumplir si ⊕ es suryectiva.
- si no es suryectiva ⟹ va a contarse un i al principio y luego no.

Ejemplo bueno: $R.i = -4 \leq i \leq 4$ $f.j = j+1$ $T.i = i$ $\oplus = \sum$

$\langle \oplus i : -4 \leq i \leq 4 : T.i \rangle$

$= \{$cambio variable con $f.j = j+1\}$

$\langle \oplus j : \underbrace{-4 \leq j+1 \leq 4}_{R(f.j)} : T.(j+1) \rangle$

$= \{$aritmetica$\}$

$\langle \oplus j : -5 \leq j \leq 3 : T.(j+1) \rangle$

$= \langle \sum i : -4 \leq i \leq 4 : i \rangle = \langle \sum j : -5 \leq i \leq 3 : j+1 \rangle$

$= 0$ $= 0$

Ejemplo malo:

$f.j = j^2$

$\langle \sum i : -4 \leq i \leq 4 : i \rangle$

$\overset{?}{\equiv}$ {cambio variable mal}

$\langle \sum j : -4 \leq j^2 \leq 4 : j^2 \rangle$

$j \in \{-2, -1, 0, 1, 2\}$

$\underbrace{4 + 1}_{\text{se suman 2 veces porque no es inyectiva}} + 0 + 1 + 4 = 10$

→ no es suryectiva porque ningún ~~n~~ elevado al 2 va a -3.

T10: Separación de un Término

$\langle \oplus i : 0 \leq i < n+1 : T.i \rangle = T.0 \oplus \underbrace{\langle \oplus i : 0 \leq i < n : T.(i+1) \rangle}$

$$\begin{array}{ccccccc} 0 & & 1 & \dots & & & n \\ \downarrow & & \downarrow & & & & \downarrow \\ T.0 & \oplus & T.1 & \oplus & \dots & \oplus & T.n \end{array} \qquad \begin{array}{ccccccccc} & & 0 & & 1 & \dots & & & n-1 \\ & & \downarrow & & \downarrow & & & & \downarrow \\ \cancel{T.0 \oplus} & & T.1 & \oplus & T.2 & \oplus & \dots & \oplus & T.n \end{array}$$

$\boxed{0 \mid \dots \mid n} \Longrightarrow \underset{0}{\boxed{}}\ \underset{1 \quad\quad n}{\boxed{\dots}} \Longrightarrow \underset{0 \quad\quad n-1}{\boxed{\dots}}\ \underset{n}{\boxed{}}$

Demostración:

$\langle \oplus i : 0 \leq i < n+1 : T.i \rangle$

$\equiv$ {Lógica}

$\langle \oplus i : i = 0 \vee 0 < i < n+1 : T.i \rangle$

$\equiv$ {Partición de Rango}

$\langle \oplus i : i = 0 : T.i \rangle \oplus \langle \oplus i : 0 < i < n+1 : T.i \rangle$

$\equiv$ {Rango unitario}

$T.0 \oplus \langle \oplus i : 0 < i < n+1 : T.i \rangle$

$\equiv$ {cambio de variable} $f.i = i+1$

$T.0 \oplus \langle \oplus i : 0 < i+1 < n+1 : T.(i+1) \rangle$

$\equiv$ {Aritmética: $-1 < i < n \equiv 0 \leq i < n$}

$T.0 \oplus \langle \oplus i : 0 \leq i < n . T.(i+1) \rangle$ #

Práctico 1 - Ejercicio extra -

3 a)
$$\langle \oplus i,j : 0 \leq i \leq j < n+1 : T.i.j \rangle = \langle \oplus i,j : 0 \leq i \leq j < n : T.i.j \rangle$$
$$\oplus \langle \oplus i : 0 \leq i < n+1 : T.i.\underset{\text{última fila}}{n} \rangle$$

j\i	0	1	2	...	n-1	n
0	X					
1	X	X				
2	X	X	X			
⋮						
n-1	X	X	X	X	X	
n	X	X	X	X	X	X

Una forma: separo por un lado este pedazo de triángulo y ~~el otro~~ lo que sobra por el otro lado.

Otra forma: separo esta columna

(utilizo Anidado).

Cuantificador de conteo: (N)

$$\langle \sum i : i \bmod 7 = 0 \wedge 0 \leq i < 100 : 1 \rangle$$ → voy sumando 1, a medida que voy encontrando otro n° divisible por 7.

$$= \langle N i : \underbrace{0 \leq i < 100}_{} : \underbrace{i \bmod 7 = 0}_{} \rangle$$
↳ el término es un predicado.

Definición: $\boxed{\langle N i : R.i : T.i \rangle = \langle \sum i : R.i \wedge T.i : 1 \rangle}$ → reglas aparte en un minidigesto. (Teoremas).

Rango vacío:

$\langle N i : False : T.i \rangle = 0$

Rango unitario:

$\langle N i : i = C : T.i \rangle = (T.C \Rightarrow 1$

alternativa ← $\square\ \neg T.C \Rightarrow 0$

$)$ → cerrado

$\langle N i : i = C : T.i \rangle = (T.C \rightarrow 1$

$\square\ \neg T.C \rightarrow 0$

$)$

Demostración: $\langle N i : i = C : T.i \rangle$

Leibniz:

$\bullet\ x = y \wedge P.x \equiv x = y \wedge P.y$

$= \{\text{Def. de } N\}$

$\langle \sum i : i = C \wedge T.i : 1 \rangle$

$= \{\text{Leibniz}\}$ → Análisis por casos

$\langle \sum i : i = C \wedge T.C : 1 \rangle$

* Si $T.C$:

$= \{\text{Sustitución}\}$

$\langle \sum i : i = C \wedge True : 1 \rangle$

$= \{\text{Neutro } \wedge\}$

$\langle \sum i : i = C : 1 \rangle$

$= \{\text{Rango unitario}\}$

1 $\blacksquare$

* Si $\neg T.C$:

$= \{\text{sustitución}\}$

$\langle \sum i : i = C \wedge False : 1 \rangle$

$= \{\text{Absorbente } \wedge\}$

$\langle \sum i : False : 1 \rangle$

$= \{\text{Rango vacío}\}$

0 $\blacksquare$

(8) $\langle N i : R.i : T.i \rangle \equiv \langle \sum i : R.i \wedge T.i : 1 \rangle$

a) Rango vacío:

$\langle N i : False : Ti \rangle = 0$

$\equiv$ { Def. de conteo }

$\langle \sum i : False \wedge Ti : 1 \rangle$

$=$ { Absorbente de $\wedge$ }

$\langle \sum i : False : 1 \rangle$

$=$ { Rango vacío $\sum$ }

0 #

Partición de rango:

$\langle N.i : S.i \vee R.i : Ti \rangle = \langle \sum i : Si \wedge T.i : 1 \rangle + \langle \sum i : R.i \wedge T.i : 1 \rangle$

$\langle N.i : S.i \vee R.i : Ti \rangle$

$=$ { Def de conteo }

$\langle \sum i : (Si \vee Ri) \wedge Ti : 1 \rangle$

$=$ { Distrib. de $\vee$ con $\wedge$ }

$\langle \sum i : (Si \wedge Ti) \vee (R.i \wedge Ti) : 1 \rangle$

$=$ { Partición de rango }

$\langle \sum i : Si \wedge Ti : 1 \rangle + \langle \sum i : Ri \wedge Ti : 1 \rangle$

$=$ { Def. de conteo }

$\langle N i : Si : Ti \rangle + \langle N i : Ri : Ti \rangle$ #

(8) b) $\langle \Sigma i : R.i \wedge T.i : k \rangle = \langle N i : R.i : T.i \rangle \times K$

$= \{\text{aritmética}\}$

$\langle \Sigma i : R.i \wedge T.i : 1 * k \rangle$, K constante

$= \{\text{Distrib}\}$

$k * \langle \Sigma i : R.i \wedge T.i : 1 \rangle$

$= \{\text{Def. } N\}$

$k * \langle N i : R.i : T.i \rangle$ #

(10) $\langle \text{Min } i : R.i : -T.i \rangle = -\langle \text{Max } i : R.i : T.i \rangle$

$z = \langle \text{Max } i : R.i : f.i \rangle \equiv \langle \exists i : R.i : z = f.i \rangle \wedge \langle \forall i : R.i : f.i \leq z \rangle$

$z = \langle \text{Min } i : R.i : f.i \rangle \equiv \langle \exists i : R.i : z = f.i \rangle \wedge \langle \forall i : R.i : z \leq f.i \rangle$

$m = \langle \text{Min } i : R.i : -T.i \rangle$

$\equiv \{\text{Def. Min}\}$

$m = \langle \exists i : R.i : m = -T.i \rangle \wedge \langle \forall i : R.i : -T.i \geq m \rangle$

$\equiv \{\text{Aritmética } * (-1)\}$

$m = \langle \exists i : R.i : -m = T.i \rangle \wedge \langle \forall i : R.i : T.i \leq -m \rangle$

$\equiv \{\text{Def. Max}\}$

$-m = \langle \text{Max } i : R.i : T.i \rangle$

$\equiv \{\text{Aritmética}\}$

$m = -\langle \text{Max } i : R.i : T.i \rangle$

#

21/08/14

Demostración y derivación de algoritmos

<u>Inducción:</u>

Sea $P : Num \to Bool$

Luego, $\forall n \geq 0 : P.n$, si y sólo si:

$P.0 \wedge (P.0 \Rightarrow P.1) \wedge (P.1 \Rightarrow P.2) \wedge \ldots$

i) $P.0$

ii) $\forall n : P.n \Rightarrow P.(n+1)$

$P.0 \wedge P.1 \wedge P.2 \ldots$

- Ejemplo:

Sea $P.n \equiv \langle \Sigma i : 0 \leq i \leq n : i \rangle = \frac{n * (n+1)}{2}$

- Demostración: Por inducción en n

* Caso base:

$$
\begin{aligned}
& P.0 \\
\equiv & \{ \text{Def. } P \} \\
& \langle \Sigma i : 0 \leq i \leq 0 : i \rangle = \frac{0 * (0+1)}{2} \\
\equiv & \{ \ldots \} \\
& True
\end{aligned}
$$

* Paso inductivo: Debo probar $P.n \Rightarrow P(n+1)$. Sup. $P.n$ (Hip. Ind). Luego, $P(n+1)$.

$$
\begin{aligned}
& P.(n+1) \\
\equiv & \{ \text{Def. } P \} \\
& \langle \Sigma i : 0 \leq i \leq n+1 : i \rangle = \frac{(n+1) * ((n+1)+1)}{2} \\
\equiv & \{ \text{Álgebra } 0 \leq i \leq n+1 \equiv 0 \leq i \leq n \vee i = n+1, \text{ Part. rango, R.U} \} \\
& \langle \Sigma i : 0 \leq i \leq n : i \rangle + (n+1) = \frac{(n+1) * ((n+1)+1)}{2} \\
\equiv & \{ HI \} \\
& \frac{n * (n+1)}{2} + (n+1) = \frac{(n+1) * (n+2)}{2} \\
\equiv & \{ \ldots \} \\
& \underline{True}
\end{aligned}
$$

Otras formas de inducción:

- Fuerte: i) $P.0$
  ii) $\forall n > 0 : (\forall m < n : P.m) \Rightarrow P.n$ $\Rightarrow$ $P.0 \wedge (P.0 \Rightarrow P.1) \wedge (P.0 \wedge P.1 \Rightarrow P.2) \wedge (P.0 \wedge P.1 \wedge P.2 \Rightarrow P.3) \ldots$

  si y sólo si $\forall n \geq 0 : P.n$
  $\hookrightarrow P.0 \wedge P.1 \wedge \ldots$

- Otra: i) $P.0$
  ii) $P.1$
  iii) $\forall n : P.n \wedge P(n+1) \Rightarrow P(n+2)$

  si y sólo si $\forall n \geq 0 : P.n$

Inducción sobre listas:

Sea $P : [A] \rightarrow Bool$. Luego, $\forall xs : P.xs$
si y sólo si:
i) $P.[\,]$
ii) $\forall x, xs : P.xs \Rightarrow P.(x \triangleright xs)$

- Ejemplo:
  Sea $sum : [Num] \rightarrow Num$ tal que:
  $sum.[\,] = 0$
  $sum(x \triangleright xs) = x + sum.xs$

Sea P tal que: $P.xs \equiv sum.xs = \langle \sum i : 0 \leq i < \#xs : xs.i \rangle$

Teorema: $\forall xs : P.xs$

Demostración: Por inducción en xs.

* **Caso base:** Probar $P.[\,]$ $\quad xs = [\,]$

$\langle \sum i : 0 \leq i < \#[\,] : [\,].i \rangle$
$= \{Def. \#\}$
$\langle \sum i : 0 \leq i < 0 : \boxed{[\,].i} \rangle$ → no está definido
$= \{$Lógica, Rango vacío$\}$
0
$= \{Def. sum\}$
$sum.[\,]$ ■

* Paso inductivo: Hay que probar $P.xs \Rightarrow P(x \triangleright xs)$

Nunca $xs = x \triangleright xs$

Sup. $P.xs$. Veamos que vale $P(x \triangleright xs)$

$\langle \sum i : 0 \leq i < \#(x \triangleright xs) : (x \triangleright xs).i \rangle$

$= \{ \text{Def. } \# \}$

$\langle \sum i : 0 \leq i < \#xs + 1 : (x \triangleright xs).i \rangle$

$= \{ \text{Álgebra} \}$

$\langle \sum i : 1 \leq i < \#xs + 1 \lor i = 0 : (x \triangleright xs).i \rangle$

$= \{ \text{Part. de rango, Rango unitario} \}$

$\langle \sum i : 1 \leq i < \#xs + 1 : (x \triangleright xs).i \rangle + (x \triangleright xs).0$

$= \{ \text{Def. } . \}$

$x + \langle \sum i : 1 \leq i < \#xs + 1 : (x \triangleright xs).i \rangle$

$= \{ \text{Restar } 1 \}$

$x + \langle \sum i : 0 \leq i - 1 < \#xs : (x \triangleright xs).i \rangle$

$= \{ \text{Sea } f.i = i + 1 \}$

$x + \langle \sum i : 0 \leq (i+1) - 1 < \#xs : (x \triangleright xs)(i+1) \rangle$

$= \{ \text{Aritmética, def. } . \}$

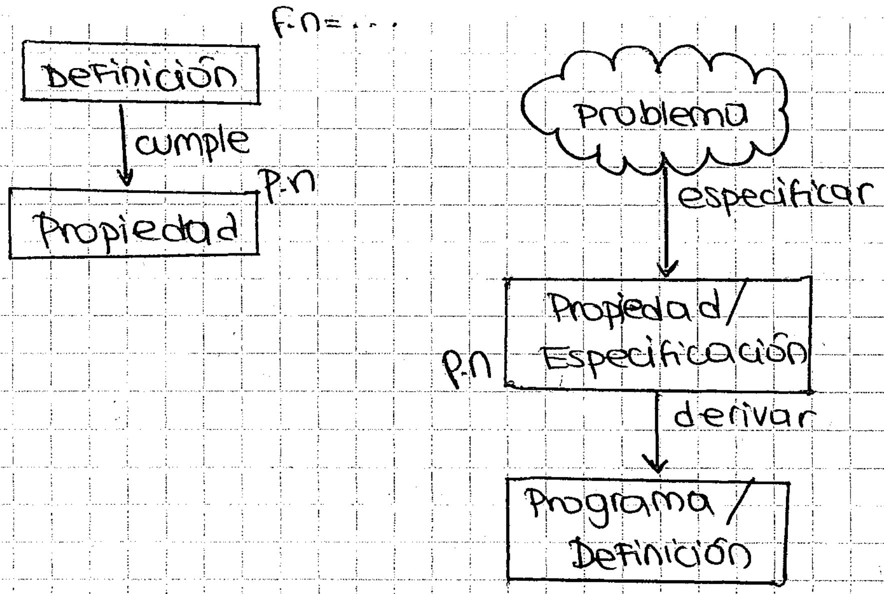
$x + \langle \sum i : 0 \leq i < \#xs : xs.i \rangle$

$= \{ \text{HI} \}$

$x + sum.xs$

$= \{ \text{Def. sum} \}$

$sum.(x \triangleright xs)$ ■



Especificación: $sum : [Num] \to Num$

Tal que cumple con la especificación

$P.xs \equiv sum.xs = \langle \sum i : 0 \leq i < \#xs : xs.i \rangle$

– Ejercicio: Obtener una def. p/ sum tal que cumpla P.xs

* Paso inductivo: $P.xs \Rightarrow P(x \triangleright xs)$

Sup. $P.xs$ y veamos $P.(x \triangleright xs)$

¿Cómo defino sum p/ llegar a True?

$$
\begin{aligned}
& sum.(x \triangleright xs) \\
= & \{ \text{Especificación} \} \\
& \langle \sum i : 0 \leq i < \#(x \triangleright xs) : (x \triangleright xs).i \rangle \\
= & \{ \ldots \} \\
& x + \langle \sum i : 0 \leq i < \#xs : xs.i \rangle \\
= & \{ HI \} \\
& x + sum.xs
\end{aligned}
$$

Luego, si defino: $sum.[\,] = 0$

$sum.(x \triangleright xs) = x + sum\,xs$

vale $P.xs$.

26/08/14

Repaso:

- Inducción con números y listas
- Dif. esquemas de inducción.

Teorema: sea $P : [A] \rightarrow Bool$.
Luego, $\forall xs : P.xs$ si y solo si

i) $P.[]$
ii) $\forall x, xs : P.xs \Rightarrow P.(x \triangleright xs)$

Teorema: $\forall xs : P.xs$ si y solo si

i) $P.[]$
ii) $\forall x .. P.[x]$
iii) $\forall x, y, xs : P.xs \wedge P(y \triangleright xs) \Rightarrow P.(x \triangleright y \triangleright xs)$

con números:
i) $P.0$
ii) $P.1$
iii) $P.n \wedge P(n+1) \Rightarrow P.(n+2)$.

1 c) $P.n : fib.n < 2^n$

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

- Ayuda 1: Ind. $P.0$, $P.1$, $P.n \wedge P.(n+1) \Rightarrow P(n+2)$
- Ayuda 2:
$fib.(n+2) < 2^{n+2}$

Hipótesis.
$fib.n < 2^n$
$fib.(n+1) < 2^{n+1}$

$\boxed{\text{Derivación}}$

- Especificación: sea $sum : Num \rightarrow Num$:

Teorema aparte
→ complejidad

$$sum.n = \langle \sum i : 0 \leq i \leq n : i \rangle = \frac{n \times (n \times 1)}{2}$$

- Derivación: Por inducción en n

caso base: $n = 0$

$sum.0$
$= \{esp.\}$
$\langle \sum i : 0 \leq i \leq 0 : i \rangle$
$= \{R.U\}$
0.

Paso inductivo: $P(n+1)$

$sum.(n+1)$
$= \{esp.\}$
$\langle \sum i : 0 \leq i \leq n+1 : i \rangle$
$= \{P. \text{ de rango y } R.U\}$
$\langle \sum i : 0 \leq i \leq n : i \rangle + (n+1)$
$= \{HI\}$
$sum.n + (n+1)$

#

Luego, si def.

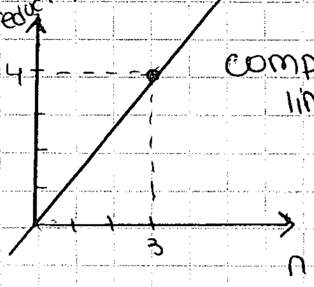
$$\text{sum}.0 = 0$$
$$\text{sum}(n+1) = \text{sum}.n + (n+1)$$

Puedo demostrar que vale $\forall n : P.n$

$$\begin{aligned}\text{sum}.3 &= \text{sum}.(2+1)\\ &= \{\text{def}\}\\ &= \text{sum}.2 + (2+1)\\ &= \text{sum}(1+1) + 3\\ &= \text{sum}.1 + (1+1) + 3\\ &= \text{sum}.(0+1) + 5\\ &= \text{sum}.0 + 1 + 5\\ &= 6\end{aligned}$$

\#pasos 6

\#reducciones 4



complejidad lineal.

Teorema: $\text{sum}.n = \dfrac{n \times (n+1)}{2}$

- Deriv:

$$\begin{aligned}&\text{sum}.n\\ =\ &\{\text{esp.}\}\\ &\langle \sum i : 0 \leq i \leq n : i \rangle\\ =\ &\{\text{teo. clase pasada}\}\\ &\frac{n \times (n+1)}{2} \quad \#\end{aligned}$$

Def: $\text{sum}.n = \dfrac{n \times (n+1)}{2}$

$$P.n \equiv prod.n = \langle \prod i : 0 < i \leq n : i \rangle$$
$$= n!$$

Definición:

$$prod.0 = 1$$

$$prod.(n+1) = (n+1) * prod.n$$

<u>Pattern matching vs Análisis por casos</u>

- $sum.[\,] = 0$
  $sum.(x \triangleright xs) = x + sum.xs$

$$sum.xs = (\,xs = [\,] \rightarrow 0$$
$$\square\; xs \neq [\,] \rightarrow$$
$$hd(xs) + sum.(tl\,(xs))$$
$$)$$

$= xs.0 + sum\,(xs \downarrow 1)$

- $fib.0 = 0$

$$fib.1 = 1$$
$$fib.(n+2) = fib.n + fib(n+1)$$

$$fib.n = (\,n = 0 \rightarrow 0$$
$$\square\; n = 1 \rightarrow 1$$
$$\square\; n \geq 2 \rightarrow$$
$$fib.(n-2) + fib.(n-1)$$
$$)$$

- $iga.e.xs = (\; xs = [\,] \rightarrow True$

$$\square\; xs \neq [\,] \rightarrow$$
$$(\,hd.xs = e \rightarrow iga.e(tl).xs$$
$$\square\; hd\,xs \neq e \rightarrow False$$
$$)$$
$$)$$

$|[\,ys = tl.xs\,]|$

en lugar de hd, puedo poner x, y poner $|[\,x = hd.xs\,]|$ → definición local.

complejidad constante

28/08/14

Propiedad: predicado en lenguaje matemático que afirma algo acerca de una función

Ej: (1): $fib.n < 2^n$

Ej: (2): $sum.xs = \langle \sum i : 0 \leq i < \#xs : xs.i \rangle$

Especificación: una propiedad que formaliza un problema que se quiere solucionar a través de una función. (me dice el "qué")

Programa: una definición de la función que resuelve el problema usando el lenguaje de programación. (me dice el "como").

Demostración: dada una propiedad y un programa probar que el programa satisface la propiedad.

Derivación: dada una especificación obtener un programa que la satisface.

Observación: una derivación me da una demostración.

Ejemplo:

Problema: Una función que me dice si un número es par

Esp: $es\_par : Num \rightarrow Bool$
$es\_par.n \equiv n \bmod 2 = 0$

Derivación:

$$\begin{aligned} & es\_par.n \\ = & \{esp.\} \\ & n \bmod 2 = 0 \quad \blacksquare \end{aligned}$$

Definición:
$es\_par.n \doteq n \bmod 2 = 0$

Otra Esp: $es\_par.(2n) \wedge \neg es\_par.(2n+1)$

<u>Modularización</u>

Ejercicio. 5 a)

$f.x.n = \langle \sum i : 0 \leq i < n : x^i \rangle$ → suma de las primeras n poten
de x.

<u>Derivación</u>: Por inducción en n

* Caso base: $n=0$

$f.x.0$
$= \{esp.\}$
$\langle \sum i : 0 \leq i < 0 : x^i \rangle$
$= \{\text{Aritmética}\}$
$\langle \sum i : False : x^i \rangle$
$= \{\text{Rango vacío}\}$
$0.$

* Caso inductivo: sop. P.n. Ver P.(n+1) → Usar P.(n+1) para obtener el programa.

$f.x.(n+1)$
$= \{esp.\}$
$\langle \sum i : 0 \leq i < n+1 : x^i \rangle$
$= \{\text{aritmética}\}$
$\langle \sum i : 0 \leq i < n \lor i=n : x^i \rangle$
$= \{\text{partición de rango}\}$
$\langle \sum i : 0 \leq i < n : x^i \rangle + \langle \sum i : i=n : x^i \rangle$
$= \{\text{HI y rango unitario}\}$
$f.x.n + x^n$ → no es programa
$= \{\text{Modularización } g.x.n = x^n\}$
$f.x.n + g.x.n$ ∎

Programa:

$f.x.0 = 0$
$f.x.(n+1) = f.x.n + g.x.n$
$g.x.0 = 1$
$g.x.n = x * g.x.n$ (Ejercicio: derivar g).

Modularización: es la identificación de un sub-problema que me ayuda a solucionar el problema actual. El sub-problema introduce una nueva función con su especificación, y debe derivarse el subprograma (en el caso anterior el subprograma es g).

Ejercicio. 5 b)

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

Derivación:

$pi.n$

$= \{$Modularización $pi'.n = \langle \sum i : 0 \leq i < n : (-1)^i / (2 * i + 1) \rangle \}$

$4 * pi'.n$ → es un programa si yo derivo pi'.n (Ejercicio).

5 c) $f.x = x^3$ (Ejercicio si no tengo producto)

$f.x$

$= \{esp.\}$

x^3

$= \{$aritmética$\}$

$x * x * x$

6 b) creciente : [Int] → Bool

$creciente.xs \equiv \langle \forall i : 0 \leq i < \#xs - 1 : xs.i < xs.(i+1) \rangle$

$\equiv \langle \forall i,j : 0 \leq i < j < \#xs : xs.i < xs.j \rangle$ → otra opción.

Derivación: Por inducción en xs

* caso base: xs = []

creciente.[]

$= \{esp.\}$

$\langle \forall i : 0 \leq i < \#[] - 1 : [].i < [].(i+1) \rangle$

$\equiv \{$Rango vacío$\}$

True

* Paso inductivo:

crecienté$(x \triangleright xs)$ No necesariamente esto es igual a $i=0 \vee 1 \leq i < \#xs$.

$= \{esp.\}$

$\langle \forall i : \overbrace{0 \leq i < \#(x \triangleright xs) - 1}^{R.V.} : (x \triangleright xs).i < (x \triangleright xs).(i+1) \rangle$

$= \{$Aritmética y def. $\#\}$

$\langle \forall i : i = 0 \vee 1 \leq i < \#xs : (x \triangleright xs).i < (x \triangleright xs).(i+1) \rangle$

No puedo seguir. Necesito ver que es xs. Hago inducción en xs. Equivalentemente, uso esquema de inducción siguiente:

i) $P.[\,]$
ii) $P.[x]$
iii) $P.(x \triangleright xs) \Rightarrow P.(y \triangleright x \triangleright xs)$.

* <u>caso base 2</u>: $xs = [x]$

creciente. $[x]$

$= \{esp.\}$

$\langle \forall i : 0 \leq i < \#[x] - 1 : [x].i < [x].(i+1) \rangle$

$= \{$def. $\#$ y R.V$\}$

<u>True</u>

* <u>Paso inductivo</u>: Sup. $P.(x \triangleright xs)$. ver $P.(y \triangleright x \triangleright xs)$

creciente$(y \triangleright x \triangleright xs)$

$= \{esp.\}$

$\langle \forall i : 0 \leq i < \#(y \triangleright x \triangleright xs) - 1 : (y \triangleright x \triangleright xs).i < (y \triangleright x \triangleright xs).(i+1)$

$= \{$Def $\#$ 2 veces$\}$

$\langle \forall i : 0 \leq i < \#xs + 1 : (y \triangleright x \triangleright xs).i < (y \triangleright x \triangleright xs).(i+1) \rangle$

$= \{$Aritmética$\}$

$\langle \forall i : i = 0 \vee 1 \leq i < \#xs + 1 : (y \triangleright x \triangleright xs).i < (y \triangleright x \triangleright xs).($

$= \{$Partición de rango y Rango unitario$\}$

$(y \triangleright x \triangleright xs).0 < (y \triangleright x \triangleright xs).1 \wedge \langle \forall i : 1 \leq i < \#xs + 1 : (y \triangleright x \triangleright x$

$< (y \triangleright x \triangleright xs).(i+1$

$\leadsto$

$= \{\text{def} \cdot\}$

$y < x \wedge \langle \forall i : 1 \leq i < \#xs+1 : (y \triangleright x \triangleright xs)\, i < (y \triangleright x \triangleright xs).(i+1) \rangle$

$= \{ i = i+1 \}$

$y < x \wedge \langle \forall i : 1 \leq i+1 \leq \#xs+1 : (y \triangleright x \triangleright xs)(i+1) < (y \triangleright x \triangleright xs).(i+2) \rangle$

$= \{\text{Aritmética y def} \cdot\}$

$y < x \wedge \langle \forall i : 0 \leq i < \#(x \triangleright xs)-1 : (x \triangleright xs) \cdot i < (x \triangleright xs).(i+1) \rangle$

$= \{HI\}$

$y < x \wedge \text{creciente}.(x \triangleright xs)$

Luego, defino:

creciente.[] = True
creciente.[x] = True
creciente.$(y \triangleright x \triangleright xs) = y \leq x \wedge$ creciente.$(x \triangleright xs)$.

2/09/14

Repaso

- Modularización: Identificar y resolver un sub-problema de mi problema.

- Derivación por inducción. Más esquemas. Ejercicio 6 b):

i) P.[]

ii) P.[x]

iii) P.$(x \triangleright xs) \Rightarrow$ P.$(y \triangleright x \triangleright xs)$

creciente.[] = True
creciente.[x] = True
creciente.$(y \triangleright x \triangleright xs) = y \leq x \wedge$ creciente.$(x \triangleright xs)$

creciente.[-1,2,5] = (corrida).

Ejercicio 7 a) psum: [Num] → Bool

$$psum.xs = \langle \forall i : 0 \leq i \leq \#xs : sum.(xs \uparrow i) \geq 0 \rangle$$

•) Tomar: $\uparrow : [A] \to Num \to [A]$

se solapan
$$\begin{cases} xs \uparrow 0 = [\,] \\ [\,] \uparrow n = [\,] \end{cases}$$
$$(x \triangleright xs) \uparrow (n+1) = x \triangleright (xs \uparrow n)$$

$$\begin{cases} [\,] \uparrow n = [\,] \\ (x \triangleright xs) \uparrow 0 = [\,] \\ (x \triangleright xs) \uparrow (n+1) = x \triangleright (xs \uparrow n) \end{cases}$$

$$[\underset{x}{1}, \underbrace{3,5,7}_{xs}] \uparrow 2 = 1 \triangleright ([3,5,7] \uparrow 1)$$
$$= 1 \triangleright (3 \triangleright ([5,7] \uparrow 0))$$
$$= 1 \triangleright (3 \triangleright [\,]) = [1,3]$$

Tomar $xs \uparrow n$ devuelve la lista de los primeros n elementos de

•) $sum.[\,] = 0$
$sum.(x \triangleright xs) = x + sum.xs$.

psum [3,-1] = ?

$i \in \{0,1,2\}$ $\quad sum(\underbrace{xs \uparrow 0}_{[\,]}) \geq 0 \wedge sum.(\underbrace{xs \uparrow 1}_{[3]}) \geq 0 \wedge sum.(\underbrace{x \uparrow 2}_{[3,-1]}) \geq$

$0 \geq 0 \quad \wedge \quad 3 \geq 0 \quad \wedge \quad 2 \geq 0$

True

"prefijos inic
"listas inicia
"segmentos i
cia

* suma los elementos de los prefijos iniciales.

¿Qué hace psum?

Se fija si todas las sumas de los elementos de los segment iniciales ej ≥ 0

↳

Derivación: Por inducción en xs

*** Caso base:**

$$\mathrm{psum}.[\,]$$
$$= \{\text{especif.}\}$$
$$\langle \forall i : 0 \leq i \leq \#[\,] : \mathrm{sum}.([\,]\uparrow i) \geq 0 \rangle$$
$$= \{\text{def. } \#\}$$
$$\langle \forall i : 0 \leq i \leq 0 : \mathrm{sum}.([\,]\uparrow i) \geq 0 \rangle$$
$$= \{\text{Rango vacío}\}$$
$$\mathrm{sum}.([\,]\uparrow 0) \geq 0$$
$$= \{\text{def. } \uparrow\}$$
$$\mathrm{sum}.[\,] \geq 0$$
$$= \{\text{def. sum}\}$$
$$0 \geq 0$$
$$= \{\text{Aritmética}\}$$
$$\underline{\text{True}}$$

*** Paso inductivo:**

$$\mathrm{psum}.(x \triangleright xs)$$
$$= \{\text{especif.}\}$$
$$\langle \forall i : 0 \leq i \leq \#(x \triangleright xs) : \mathrm{sum}.((x \triangleright xs)\uparrow i) \geq 0 \rangle$$
$$= \{\text{def. } \#, \text{ Lógica}\}$$
$$\langle \forall i : i = 0 \vee 1 \leq i \leq \#xs + 1 : \mathrm{sum}.((x \triangleright xs)\uparrow i) \geq 0 \rangle$$
$$= \{\text{Partición de Rango, y R. unitario en el primero}\}$$
$$\mathrm{sum}((x \triangleright xs)\uparrow 0) \geq 0 \ \wedge \langle \forall i : 1 \leq i \leq \#xs+1 : \mathrm{sum}((x \triangleright xs)\uparrow i$$
$$= \{\text{Varios pasos en el } 1^{ro}, \text{ y cambio variable } i \leftarrow i+1 \text{ en el 2do}\}$$
$$\text{True} \ \wedge \langle \forall i : 1 \leq i+1 \leq \#xs+1 : \mathrm{sum}.((x \triangleright xs)\uparrow(i+1)) \geq 0 \rangle$$
$$= \{\text{Neutro } \wedge, \text{ arit. en rango, def. } \uparrow\}$$
$$\langle \forall i : 0 \leq i \leq \#xs : \underline{\mathrm{sum}.(x \triangleright (xs \uparrow i))} \geq 0 \rangle$$
$$= \{\text{Def. sum}\}$$
$$\langle \forall i : 0 \leq i \leq \#xs : \underline{x + \mathrm{sum}(xs \uparrow i)} \geq 0 \rangle$$

No puedo seguir!

No puedo llegar a la HI porque molesta el "x+". Si el problema tuviera ese término, podría aplicar HI.

Nuevo problema: $gpsum : Num \rightarrow [Num] \rightarrow Bool$

$gpsum.n.xs = \langle \forall i : 0 \leq i \leq \#xs : n + sum.(xs \uparrow i) \geq 0 \rangle$

1. Derivar desde cero
2. Ver qué tiene que ver con psum.

Derivación gpsum: Ind. en xs

* Caso base: $gpsum.n.[]$

$= \{ \text{esp. y varios pasos} \}$

$\langle \forall i : i = 0 : n + sum.([] \uparrow i) \geq 0 \rangle$

$= \{ \text{Rango unitario} \}$

$n + sum.([] \uparrow 0) \geq 0$

$= \{ \text{def. } \uparrow \text{ y sum} \}$

$n + 0 \geq 0$

$= \{ \text{aritm.} \}$

$n \geq 0$ → se puede programar, está terminado. ■

* Paso inductivo:

$gpsum.n.(x \triangleright xs)$

$= \{ \text{especif.} \}$

$\langle \forall i : 0 \leq i \leq \#(x \triangleright xs) : n + sum((x \triangleright xs) \uparrow i) \geq 0 \rangle$

$= \{ \text{def } \# \text{ y aritmética en rango} \}$

$\langle \forall i : i = 0 \vee 1 \leq i \leq \#xs + 1 : n + sum((x \triangleright xs) \uparrow i) \geq 0 \rangle$

$= \{ \text{Partición de rango y rango unitario} \}$

$n + sum((x \triangleright xs) \uparrow 0) \geq 0 \wedge \langle \forall i : 1 \leq i \leq \#xs + 1 : n + sum((x \triangleright xs) \uparrow i) \geq 0 \rangle$

$= \{ \text{def } \uparrow, \text{def sum y álgebra} \}$

$n \geq 0 \wedge \langle \forall i : 1 \leq i \leq \#xs + 1 : n + sum((x \triangleright xs) \uparrow i) \geq 0 \rangle$

$= \{ \text{cambio var } i \leftarrow i+1, \text{ arit.} \}$

$n \geq 0 \wedge \langle \forall i : 0 \leq i \leq \#xs : n + sum((x \triangleright xs) \uparrow (i+1)) \geq 0 \rangle$

$= \{ \text{def. } \uparrow \}$

$n \geq 0 \wedge \langle \forall i : 0 \leq i \leq \#xs : n + sum(x \triangleright (xs \uparrow i)) \geq 0 \rangle$

↳

$= \{\text{def sum}\}$

$n \geq 0 \wedge \langle \forall i : 0 \leq i \leq \# xs : n + (x + sum(xs \uparrow i)) \geq 0 \rangle$

$= \{\text{asociatividad } +\}$

$n \geq 0 \wedge \langle \forall i : 0 \leq i \leq \# xs : (n + x) + sum(xs \uparrow i) \geq 0 \rangle$

$= \{HI\}$

$n \geq 0 \wedge gpsum.(n+x).xs$.

i) P.[ ]
ii) P.xs ⇒ P.(x ▷ xs)
¿Qué es P?
$P.x \equiv \forall n . gpsum.n.x$
$:= \langle \forall i \ldots \rangle$ (esp.)
n vale para cualquier cosa, en particular (n+x).

$P.xs \equiv \forall n : gpsum.n.xs = \langle \forall i : 0 \leq i \leq \# xs : n + sum.(xs \uparrow i) \geq 0 \rangle$

* paso inductivo: supongo que vale P.xs. veamos P.(x ▷ xs).
Dado n, veamos gpsum.n.(x ▷ xs). Por H.I, en particular,
$gpsum.(n+x).xs = \langle \forall i : 0 \leq i \leq \# xs : (n+x) + sum.(xs \uparrow i) \geq 0 \rangle$

Definición:

$gpsum.n.[\,] = n \geq 0$

$gpsum.n.(x \triangleright xs) = n \geq 0 \wedge gpsum.(n+x).xs$

(Respondiendo preguntas del principio)

2. ¿Qué relación hay en psum?

psum.xs = gpsum.0.xs

- psum es una forma particular de gpsum.
- gpsum generaliza a psum.

$psum.[\underbrace{2}_{x}, \underbrace{-1, -2}_{xs}] = gpsum.0.[2, -1, -2]$

$= 0 \geq 0 \wedge gpsum.2.[-1, -2]$

$= 0 \geq 0 \wedge 2 \geq 0 \wedge gpsum.+1.[-2]$

$= 0 \geq 0 \wedge 2 \geq 0 \wedge 1 \geq 0 \wedge gpsum.-1.[\,]$

$= 0 \geq 0 \wedge 2 \geq 0 \wedge 1 \geq 0 \wedge -1 \geq 0$

$= true \wedge True \wedge True \wedge false$

$= False.$

FECHA 9/09/14

Repaso:

- Generalización
- Ejercicio 7b (sum-ant)
- Ejercicio 8a (cuad).

$$cuad.n = \langle \exists i : 0 \leq i \leq n : i^2 = n \rangle$$
$$gcuad.m.n = \langle \exists i : 0 \leq i \leq n : i^2 = m+n \rangle$$

$$cuad.n \doteq gcuad.0.n$$
$$gcuad.m.0 \doteq (m=0)$$
$$gcuad.m.(n+1) = (n+1)^2 = n+m+1 \vee gcuad.(m+1).n$$

$$cuad.6 = gcuad.0.6$$
$$= 6^2 = 6 \vee gcuad.1.5$$
$$= False \vee 5^2 = 6 \vee g.cuad.2.4$$
$$= \underbrace{False \vee \underbrace{False}} \vee \underbrace{4^2 = 6} \vee gcuad.3.3$$
$$= \underbrace{False \vee \cancel{gcu}\ 3^2 = 6} \vee gcuad.4.2$$
$$= \underbrace{False \vee 2^2 = 6} \vee gcuad.5.1$$
$$= \underbrace{False \vee 1^2 = 6} \vee gcuad.6.0$$
$$= False \vee (6=0)$$
$$= \underline{False}$$

#

Segmentos de lista

Ej. 9 b) Especificar "la lista xs es un segmento de la lista ys".

$$\langle \exists i,j : 0 \leq i \leq j \leq \#ys : xs = (ys \downarrow j) \uparrow i \rangle$$

$$\langle \exists i : 0 \leq i \leq \#ys : xs = (ys \downarrow i) \uparrow \#xs \rangle$$

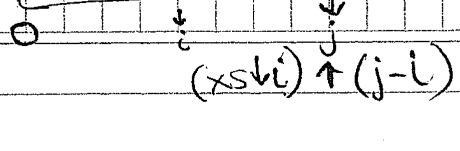
"La suma de todos los segmentos de xs"

$$\langle \sum i,j : 0 \leq i \leq j \leq \#xs : sum.((xs \downarrow i) \uparrow (j-i)) \rangle$$

$$\boxed{\langle \sum as,bs,cs : xs = as +\!\!+ bs +\!\!+ cs : sum.bs \rangle}$$

concatenación

$$\underbrace{as \quad bs \quad cs}_{xs}$$



Utilizando la concatenación, especifico:

"la lista xs es un segmento de la lista ys"

•) $\langle \exists as, bs, cs : ys = as ++ bs ++ cs : xs = bs \rangle$

•) $\langle \exists as, cs :: ys = as ++ xs ++ cs \rangle$

Definiciones:

- La lista ys es segmento de la lista xs sii $\exists as, bs$:

$$xs = as ++ ys ++ bs$$

- La lista ys es segmento inicial de xs sii $\exists as$:

$$xs = ys ++ as$$

- La lista ys es segmento final de xs sii $\exists as$:

$$xs = as ++ ys$$

• segmentos iniciales:

$\#xs + 1$ (igual con los finales)

• cant. de segmentos (segmentos "normales")

$\binom{\#xs}{2}$

Ejercicio 11 a): segmento de suma mínima

$sumin.xs = \langle Min\ as, bs, cs : xs = as ++ bs ++ cs : sum.bs \rangle$

Por inducción en xs.

$xs ++ ys = []$
$\Leftrightarrow xs = [] \wedge ys = [$

Caso base = $sumin.[]$

$= \{especif.\}$

$\langle Min\ as, bs, cs : [] = as ++ bs ++ cs : sum.bs \rangle$

$= \{prop. []\}$

$\langle Min\ as, bs, cs : as = [] \wedge bs = [] \wedge cs = [] : sum.bs \rangle$

$= \{Anidado\}$

$\langle Min\ bs : bs = [] : \langle Min\ as, cs : as = [] \wedge cs = [] : sum.bs \rangle \rangle$

NOTA

HOJA

FECHA

$= \{\text{Rango unitario}\}$ No es vacío, puedo aplicar término constante.

$\langle \text{Min } as, cs : as = [\,] \wedge cs = [\,] : sum.[\,] \rangle$

(*) $= \{\text{término constante}\}$

$sum.[\,]$

$= \{\text{Def. sum}\}$

0 . ■

otra forma desde (*)

$= \{\text{Anidado y Rango unitario con } as = [\,]\}$

$\langle \text{Min } as : cs = [\,] : sum.[\,] \rangle$

$= \{\text{Rango unitario}\}$

$sum.[\,]$

$= \{\text{Def. sum}\}$

0

<u>Paso inductivo:</u>

$sumin.(x \triangleright xs)$

$= \{\text{especif.}\}$

$\langle \text{Min } as, bs, cs : x \triangleright xs = as +\!\!+ bs +\!\!+ cs : sum.bs \rangle$

$= \{3\underline{ro} \text{ excluido}\}$

$\langle \text{Min } as, bs, cs : x \triangleright xs = as +\!\!+ bs +\!\!+ cs \wedge (as = [\,] \vee as \neq [\,]) : sum.bs \rangle$

$= \{\text{distributividad y partición de rango}\}$ deshacerme de la variable: anid Y RU

$\langle \text{Min } as, bs, cs : x \triangleright xs = as +\!\!+ bs +\!\!+ cs \wedge as = [\,] : sum.bs \rangle$ min

$\langle \text{Min } as, bs, cs : x \triangleright xs = as +\!\!+ bs +\!\!+ cs \wedge as \neq [\,] : sum.bs \rangle$

$= \{\text{Anidado}\}$

$\langle \text{Min } as : as = [\,] : \langle \text{Min } bs, cs : x \triangleright xs = as +\!\!+ bs +\!\!+ cs : sum.bs \rangle \rangle$ min

$\langle \text{Min } as, bs, cs : x \triangleright xs = as +\!\!+ bs +\!\!+ cs \wedge as \neq [\,] : sum.bs \rangle$

$= \{\text{Rango unitario}\}$ solo considera segmentos iniciales

subproblema

$\langle \text{Min } bs, cs : x \triangleright xs = bs +\!\!+ cs : sum.bs \rangle$ min

$\langle \text{Min } as, bs, cs : x \triangleright xs = as +\!\!+ bs +\!\!+ cs \wedge as \neq [\,] : sum.bs \rangle$

NOTA

$$= \{\text{Modularización}\ \ sumin2.xs = \langle \text{Min } bs, cs : xs = bs +\!\!+ cs : sum.bs\rangle\}$$

$$sumin\,2\,(x \triangleright xs)\ \min\ \langle \text{Min } as, bs, cs : x \triangleright xs = as +\!\!+ bs +\!\!+ cs \wedge as \neq [\,] : sum.bs\rangle.$$

$$= \{\text{cambio variable } as \leftarrow a \triangleright as'\}$$ → lo puedo hacer porque $as \neq [\,]$

Prop: $x \triangleright xs = y \triangleright ys \iff x = y \wedge xs = ys$

$$sumin\,2.(x \triangleright xs)\ \min\ \langle \text{Min } a, as, bs, cs : x \triangleright xs = (a \triangleright as) +\!\!+ bs +\!\!+ cs \wedge (a \triangleright as) \neq [\,] : sum.bs\rangle$$

Prop: $x \triangleright xs \neq [\,] \equiv True$

$$= \{\text{propiedades de listas}\}$$

$$sumin\,2(x \triangleright xs)\ \min\ \langle \text{Min } a, as, bs, cs : x \triangleright xs = a \triangleright (as +\!\!+ bs +\!\!+ cs) \wedge True : sum.bs\rangle$$

$$= \{\text{Prop. listas}\}$$

$$sumin.2.(x \triangleright xs)\ \min\ \langle \text{Min } a, as, bs, cs : x = a \wedge xs = as +\!\!+ bs +\!\!+ cs : sum.bs\rangle$$

$$= \{\text{Anidado}\}$$

$$sumin.2.(x \triangleright xs)\ \min\ \langle \text{Min } a, a = x : \langle \text{Min } as, bs, cs : xs = as +\!\!+ bs +\!\!+ cs : sum.bs\rangle\rangle$$

$$= \{\text{rango unitario } a = x\}$$

$$sumin.2.(x \triangleright xs)\ \min\ \langle \text{Min } as, bs, cs : xs = as +\!\!+ bs +\!\!+ cs : sum.bs\rangle$$

$$= \{HI\}$$

$$sumin.2.(x \triangleright xs)\ \min\ sumin.xs$$

<u>Resultado parcial:</u>

$$sumin.[\,] = 0$$

$$sumin.(x \triangleright xs) = sumin.2.(x \triangleright xs)\ \min\ sumin.xs$$

<u>Ejercicio</u>: derivar sumin.2

→ segmentos iniciales

$$sumin.[1,2,3] = sumin.2.[1,2,3]\ \min\ sumin.[2,3]$$

$$= sumin.2[1,2,3]\ \min\ sumin.2.[2,3]\ \min\ sumin.[3]$$

11/09/14

Repaso: segmentos de lista

* Definición de segmento, segmento inicial, segmento final.
* Especificaciones con segmentos

$$sumin.xs = \langle Min\ as, bs, cs : xs = as +\!\!+ bs +\!\!+ cs : sum.bs \rangle$$

* Derivaciones con segmentos

P.1: Partir rango con $as = []$
$\vee as \neq []$

- En $as \neq []$, cambio variable $as \leftarrow a \triangleright as$
- Anidar $x = a$ y rango unitario para eliminar a.

Complejidad Algorítmica

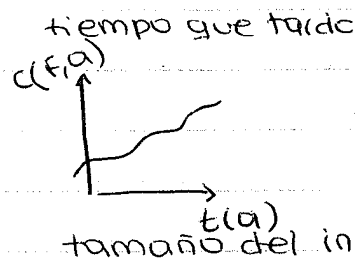
El tiempo que tarda en calcularse una función de acuerdo al tam[año] de sus parámetros de entrada.

{ también trata del espacio que ocupa en memoria - complejidad en espacio)

- Tiempo: Se puede medir en cantidad de "cuentas", operaciones elementales o pasos de reducción hasta alcanzar un valor irreducible.
- Tamaño: Depende del tipo.
    - Números: valor del número
    - Listas: largo
    - Árboles: cantidad nodos o aristas.
    - Combinaciones de las anteriores.

$f : A \rightarrow B$

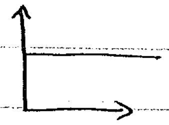
- tamaño: $t : A \rightarrow Num$
- tiempo: $c : (A \rightarrow B) \rightarrow A \rightarrow Num$



Complejidad → asociada al programa y no a lo que calcule

Ejemplos:

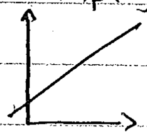
① Complejidad constante:



Sin importar el tamaño del input, va a tardar el mismo tiempo.

- ~~[illegible]~~ (sum) $= \frac{n * (n+1)}{2}$
- constantes: 0, 1, . . ., True, False, [], [0, 1], . . .
- operaciones elementales: +, *, <, ≤, =, ▷, . . . (no siempre constante para comp. arbitraria).

② Complejidad Lineal:



Una llamada recursiva y el parámetro se achica en 1 (o en una cant. ctte) (inducción básica).

fib no porque tiene 2 llamadas recursivas

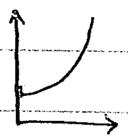
- sum
- fac
- exp. x. 0 = 1
  exp. x. (n+1) = x * exp. x. n
- concatenar → lineal en la primera lista

  [] ++ ys = ys

  (x▷xs) ++ ys = x▷ (xs ++ ys)

Todas las ~~expresiones~~ otras op. usadas deben tener comp. constante.

③ Complejidad cuadrática:

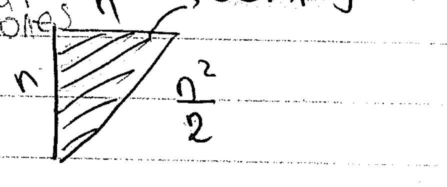


(Modularización): Una recursiva lineal que en cada paso llama a una función de complej. lineal.

- pi. n = 4 * pi'. n
  pi'. 0 = 0
  pi'. (n+1) = exp. (-1). n / (2 * n+1) + pi'. n

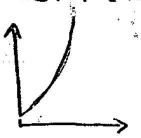
pi'. 5 → pi'. 4 → pi'. 3 → pi'. 2 → pi'. 1 → pi'. 0

exp(-1). 4	exp. 3	exp. 2	exp. 1	exp. 0
exp. (-1). 3	2	1	0	evitar repeticiones
exp. (-1). 2	1	0		
exp. (-1). 1	0			
exp. (-1). 0				



④ Complejidad exponencial:

Dos o más llamadas recursivas y el parámetro se achica en cantidad constante.

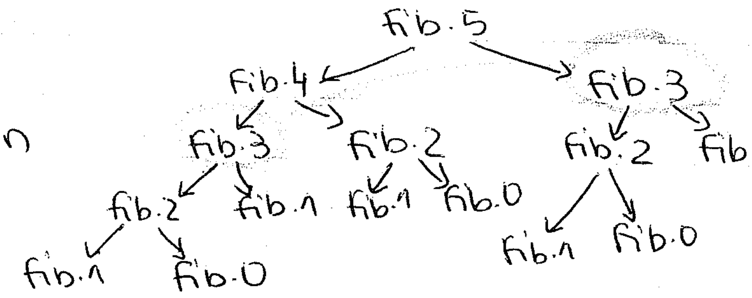


▶ fibonacci

fib.0 = 0

fib.1 = 1

fib.(n+2) = fib.(n+1) + fib.n



Árbol binario altura n.

Nodos = 2^n.

Técnica de tuplas: bajamos la complejidad

$$pi'.5 = exp.(-1).4 / (2*4+1) + pi'.4$$
$$= (-1) * exp.(-1).3 / 9 + \underline{exp.(-1).3} / 7 + \underline{pi'.3}$$
$$= (-1) * b / 9 + b / 7 + a$$
$$|[a = pi'.3$$
$$b = exp.(-1).3]|$$
$$= (-1) * b / 9 + b / 7 + a$$
$$|[(a,b) = \underbrace{(pi'.3, exp(-1).3)}_{tpi'.3}]|$$

Especificación: tpi' : Num → (Num, Num)

$$tpi'.n = (pi'.n, exp.(-1).n)$$

*Derivación: Por inducción en n

* caso base:

$$tpi'.0$$
$$= \{especif\}$$
$$(pi'.0, exp.(-1).0)$$
$$= \{def\ pi', def\ exp.\}$$
$$(0, 1) \quad \blacksquare$$

* Paso inductivo:

$$tpi'.(n+1)$$
$$= \{\text{especificación}\}$$
$$(pi'.(n+1), exp.(-1).(n+1))$$
$$= \{\text{def. } pi', \text{def. } exp\}$$
$$(exp.(-1).n/(2*n+1) + pi'.n, (-1) * exp.(-1).n)$$
$$= \{\text{definición local}\}$$
$$(b/(2*n+1) + a, (-1)*b)$$
$$|[\, a = pi'.n$$
$$b = exp.(-1).n \,]|$$
$$= \{\text{igualdad tuplas}\}$$
$$(b/(2*n+1) + a, (-1)*b)$$
$$|[\, (a,b) = (pi'.n, exp.(-1).n) \,]|$$
$$= \{HI\}$$
$$(b/(2*n+1) + a, (-1)*b)$$
$$|[\, (a,b) = tpi'.n \,]|$$

Resultado:

$$tpi'.0 \doteq (0,1)$$
$$tpi'.(n+1) \doteq (b/(2*n+1) + a, (-1)*b)$$
$$|[\, (a,b) = tpi'.n \,]|$$

Redifinición:

$pi'.n \equiv fst.(tpi'.n)$ → sustituye a la 1ra definición, siendo esta última lineal.

$$tpi'.3 = (b/5 + a, (-1)*b)$$
$$|[\, (a,b) = tpi'.2 \,]|$$
$$tpi'.2 = (b'/3 + a', (-1)*b')$$
$$|[\, (a',b') = tpi'.1 \,]|$$
$$tpi'.1 = (b''/1 + a'', (-1)*b'')$$
$$|[\, (a'',b'') = tpi'.0 \,]| = (0,1)$$

$tpi'.1 = (1/1 + 0, (-1) * 1) = (1, -1)$

$tpi'.2 = (-1/3 + 1, 1)$

$tpi'.3 = (1/5 - \frac{1}{3} + 1, -1)$

$tpi'.4 \to tpi'.3 \to tpi'.2 \to tpi'.1 \to tpi'.0$

$pi'.3 \to pi'.2 \to pi'.1$

$exp(-1).3 \to exp.(-1).2 \to exp(-1).1$

Fibonacci con tuplas:

$tFib.n = (fib.n, fib.(n+1))$