DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Trabajo práctico 1

Especificación y WP

4 de octubre de 2024 — Algoritmos y Estructuras de Datos - DC - UBA

Grupo AJMS

Integrante	LU	Correo electrónico
Ferechian, Matías	693/23	matifere@gmail.com
Nestmann, Sofía	366/23	sofianestmann@gmail.com
Mirasson, Javier	594/23	javierestebanmn@gmail.com
Ramirez, Ana	931/23	correodeanar@gmail.com

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2610 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (++54 +11) 4576-3300
http://www.exactas.uba.ar

1. Especificación

1.1. grandesCiudades

proc `grandesCiudades` (in ciudades : $seq\langle Ciudad\rangle$) : $seq\langle Ciudad\rangle${

requiere { $(\forall i, j : \mathbb{Z})$ (
$0 \leq i, j < |ciudades| \longrightarrow_L ciudades[i]_0 \neq ciudades[j]_0$
) }

asegura { $(\forall i : \mathbb{Z})$ (
$(0 \leq i < |ciudades|) \wedge_L ((ciudades[i]_1 > 50000) \longrightarrow_L (ciudades[i] \in res))$
) }

}

1.2. sumaDeHabitantes

proc `sumaDeHabitantes` (in menoresDeCiudades : $seq\langle Ciudad\rangle$, in mayoresDeCiudades : $seq\langle Ciudad\rangle$) : $seq\langle Ciudad\rangle${

requiere { $(|menoresDeCiudades| = |mayoresDeCiudades|) \wedge ((\forall i : \mathbb{Z})$ (
$(\exists! j : \mathbb{Z})$ (
$0 \leq i, j < |menoresDeCiudades| \longrightarrow menoresDeCiudades[i]_0 = mayoresDeCiudades[j]_0$
)
) }

asegura {$(\forall n, m : \mathbb{Z})$ (
$(0 \leq n, m < |menoresDeCiudades|) \wedge_L ((menoresDeCiudades[n]_0 = mayoresDeCiudades[m]_0) \longrightarrow_L$
$((res[n]_1 = menoresDeCiudades[n]_1 + mayoresDeCiudades[m]_1) \wedge$
$(res[n]_0 = menoresDeCiudades[n]_0)))$ }

}

1.3. hayCamino

proc `hayCamino` (in distancias : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, in desde : $\mathbb{Z}$, in hasta : $\mathbb{Z}$) : Bool{

requiere { $(\forall i, j : \mathbb{Z})$ (
$(0 \leq i, j, desde, hasta < |distancias|) \wedge_L ((i = j) \longrightarrow (distancias[i][j] = 0)) \wedge (distancias[i][j] = distancias[j][i])$
) }

asegura { res= true $\leftrightarrow (\exists p : seq\langle\mathbb{Z}\rangle)$ (
$(|p| \geq 1) \wedge_L ((p[0] = desde) \wedge (p[|p| - 1] = hasta) \wedge (\forall k : \mathbb{Z})$ (
$(0 \leq k < |p| - 1) \longrightarrow_L (distancias[p[k]][p[k + 1]] > 0)$
)
)}

}

1.4. cantidadCaminosNSaltos

Para la siguiente especificación tendremos en cuenta que: Dada la matriz de orden 1:

$$M_1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Si queremos encontrar la matriz M_2 de orden 2, nos queda que:

$$M_2 = M_1 \times M_1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 2 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

Luego M_2 contiene la cantidad de 2-saltos asociados a cada par i,j que se encuentre dentro de la matriz

proc `cantidadCaminosNSaltos` (inout conexión : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, in n : $\mathbb{Z}$) : {

requiere { $n > 0 \wedge (\forall i, j : \mathbb{Z})$ (
$(0 \leq i, j < |conexión|) \longrightarrow_L ((i = j) \longrightarrow (conexión[i][j] = 0) \wedge (conexión[i][j] = conexión[j][i]) \wedge (0 \leq conexión[i[j] \leq 1))$
) }

asegura { $(\exists p : seq\langle seq\langle seq\langle\mathbb{Z}\rangle\rangle\rangle)$ (
$(\forall i, j, k : \mathbb{Z})$ (
$((0 \leq k < n - 1) \wedge (0 \leq i, j < |conexión|)) \longrightarrow_L (\text{conexión}[i][j] = multEntreMatrices(p[k], p[k + 1]))$

)
) }
}
`aux multEntreMatrices` (mUno : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, mDos : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) : $\mathbb{Z}$ = $(\forall i, j : \mathbb{Z})$ (
$(0 \leq i, j < |mUno|) \wedge_L \sum_{k=1}^{|mUno|} mUno[i][k] \times mDos[k][j]$
);

1.5. caminoMinimo

`proc caminoMinimo` (in origen : $\mathbb{Z}$, in destino : $\mathbb{Z}$, in distancias : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) : $seq\langle\mathbb{Z}\rangle$ {

`requiere` { $(\forall i, j : \mathbb{Z})$ (
$(0 \leq i, j < |distancias|) \wedge_L ((i = j) \longrightarrow (distancias[i][j] = 0)) \wedge (distancias[i][j] = distancias[j][i])$
)

`asegura` {(hayCamino(distancias, origen, destino) $\wedge((\forall$k: $seq\langle\mathbb{Z}\rangle)(\exists$p: $seq\langle\mathbb{Z}\rangle)$(esCamino(distancias, origen, destino, p) $\wedge$ (sumaDistancias(distancias, origen, destino, p) $\leq sumaDistancias(distancias, origen, destino, k))) \longrightarrow res = p) \wedge$
(($\neg$hayCamino(distancias, origen, destino) $\vee$ (origen = destino)) $\longrightarrow$res = [ ])
)
}

`pred esCamino` (distancias : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, origen : $\mathbb{Z}$, destino : $\mathbb{Z}$, p : $seq\langle\mathbb{Z}\rangle$)
{ $(\forall k : \mathbb{Z})$ (
(0$\leq$ i, j, origen, destino < $|p|$) $\wedge_L distancias[p[k]][p[k+1]]$
)
}

`aux sumaDistancias` (distancias : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, origen : $\mathbb{Z}$, destino : $\mathbb{Z}$, s : $seq\langle\mathbb{Z}\rangle$) : $\mathbb{Z}$ = $\sum_{j=origen}^{destino}$ distancias$[j][j+1]$;

2. Demostraciones de correctitud

2.1. Demostración de implementación

Definimos como precondicion y postcondicion de nuestro programa:
P $\equiv \{(\exists i : \mathbb{Z})(0 \leq i < |ciudades|) \wedge_L ciudades[i].habitantes > 50{,}000 \wedge (\forall j : \mathbb{Z})$ (
$(0 \leq j < |ciudades| \longrightarrow_L ciudades[j].habitantes \geq 0) \wedge (\forall i, j : \mathbb{Z})$ (
$(0 \leq i, j < |ciudades| \longrightarrow_L ciudades[j].nombre \neq ciudades[i].nombre)$
)
)}
$Q \equiv \{\sum_{i=0}^{|ciudades|-1} ciudades[i].habitantes\}$

Para demostrar la correctitud del programa probamos que:
{P}S1;S2;S3{Q} es válida, con:

S1 $\equiv res := 0$
S2 $\equiv i := 0$
S3 $\equiv$
$while(i < ciudades.length)do$
$res = res + ciudades[i].habitantes$
$i = i + 1$
$endwhile$

Se busca por monotonía demostrar que:

1. P $\longrightarrow wp$ {S1;S2,P$_c$}
2.$P_c \longrightarrow wp\{S3, Q_c\}$
3.$Q_c \longrightarrow wp\{S4, Q\}$

Esto permite demostrar que P $\longrightarrow wp\{$S1;S2;S3;S4,Q$\}$
Es verdadera por lo cual la Tripla de Hoare es válida .

Elijo $Q_c \equiv Q$

Demuestro:
$Q_c \longrightarrow wp(S4, Q)$

Por el **Axioma 2**:
$wp(S4,Q_c) \equiv Q_c$

Para demostrar $P_c \longrightarrow wp\{S3, Q_c\}$se prueba la correctitud del ciclo mediante:

Teorema del Invariante y Teorema de Terminación

Teorema del Invariante:
Si existe un predicado I tal que:

1. $P_c \longrightarrow I$
2. $\{I \wedge B\}S\{I\}$
3. $I \wedge \neg B \longrightarrow Q_c$

(1)

Entonces el ciclo es parcialmente correcto respecto de la especificación

- Definimos

$P_c \equiv \{\text{res} = 0 \wedge i = 0\}$
$Q_c \equiv \{\text{res} = \sum_{i=0}^{|ciudades|-1} ciudades[i].habitantes\}$
$B \equiv \{i < |ciudades|\}$
$I \equiv \{0 \leq i \leq |ciudades| \wedge \text{res} = \sum\limits_{j=0}^{i-1} ciudades[j].habitantes\}$

Según el teorema del invariante

1. $P_c \longrightarrow I$

- $0 \leq i \leq |ciudades| \equiv 0 \leq 0 \leq |ciudades|$
- $\text{res} = 0 \wedge i = 0 \longrightarrow \text{res} = \sum\limits_{j=0}^{i-1} ciudades[j].habitantes \equiv \sum\limits_{j=0}^{0-1} ciudades[j].habitantes = 0$

$P_c \longrightarrow$I es válido

Ahora demuestro que:
2. $\{I \wedge B\}S\{I\} \longleftrightarrow I \wedge B \longrightarrow wp(S,\ I)$

- $wp(S,\ I) \equiv wp(\mathbf{S1;\ S2}, I)$
$\equiv wp(\mathbf{S1}, \text{wp}(\mathbf{S2,I}))$
$\equiv wp(\mathbf{S1};\ \text{i:=i+1}\ , I)$
$\equiv wp(\text{res:= res + ciudades[j].habitantes; i:=i+1}, I)$
$\equiv wp\left(res := res + ciudades[j].habitantes, wp\left(i := i, (0 \leq i \leq |ciudades|) \wedge \text{res} = \sum\limits_{j=0}^{i-1} ciudades[j].habitantes\right)\right)$
$\equiv wp\left(res := res + ciudades[j].habitantes, 0 \leq i+1 \leq |ciudades| \wedge \sum\limits_{j=0}^{i+1-1} ciudades[j].habitantes\right)$

$\equiv wp\left(res := res + ciudades[j].habitantes, -1 \leq i \leq |ciudades| - 1 \wedge \sum_{j=0}^{i} ciudades[j].habitantes\right)$

$\equiv wp\left(res := res + ciudades[j].habitantes, i < |ciudades| \wedge \sum_{j=0}^{i-1} ciudades[j].habitantes + ciudades[i].habitantes\right)$

$\equiv i < |ciudades| \wedge res + ciudades[j].habitantes = \sum_{j=0}^{i-1} ciudades[j].habitantes + ciudades[i].habitantes$

$\equiv i < |ciudades| \wedge res = \sum_{j=0}^{i-1} ciudades[j].habitantes$

Ahora pruebo I ∧ B ⟶wp(S, I) I $\wedge B \equiv 0 \leq i \leq |ciudades| \wedge res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge$ i $< |ciudades|$

$\equiv i < |ciudades| \wedge \sum_{j=0}^{i-1} ciudades[j].habitantes$

3. {I $\wedge \neg B\} \longrightarrow$ Qc

I ∧ ¬ B

$\equiv$ 0 ≤ i ≤ —ciudades— ∧ res = $\sum_{j=0}^{i-1} ciudades[j].habitantes \wedge \neg\, (i < |ciudades|)$

$\equiv 0 \leq i \leq |ciudades| \wedge$ res $= \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge i \geq |ciudades|$

$\equiv \sum_{j=0}^{|ciudades|-1} ciudades[j].habitantes$

Por el Teorema del Invariante podemos concluir que el ciclo es parcialmente correcto respecto de la especificación

Teorema de Terminación

$$\begin{aligned} &4.\{I \wedge B \wedge fv = V_o\}\mathbf{S}\{fv < V_o\} \\ &5.(I \wedge fv \leq 0) \longrightarrow \neg B \end{aligned} \tag{2}$$

Elijo como función variante:

$$fv = |ciudades| - i \tag{3}$$

- Verifico 4.

$$\begin{aligned} &\{I \wedge B \wedge fv = V_o\}\mathbf{S}\{fv < V_o\} \longleftrightarrow \{I \wedge B \wedge fv = V_o\} \longrightarrow wp(S, fv < V_o) \\ &wp(S, fv < V_o) \equiv wp(res := res + ciudades[i].habitantes; i := i+1, |ciudades| - i < V_o) \\ &\equiv wp(res := res + ciudades[i].habitantes, wp(i : i+1; |ciudades| - i < V_o)) \\ &\equiv wp(res := res + ciudades[i].habitantes, |ciudades| - 1 - i < V_o)) \\ &\text{Puedo ignorar la res ya que no es relevante en este caso} \\ &\equiv |ciudades| - 1 - i < |ciudades| - i \\ &-1 < 0 \end{aligned} \tag{4}$$

- Verifico 5.

$$\begin{aligned} &(I \wedge fv \leq 0) \longrightarrow \neg B \\ &(I \wedge fv \leq 0) \equiv 0 \leq i \leq |ciudades| \wedge \text{res} = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge |ciudades| - i \leq 0 \\ &\text{Ignoro res} \\ &\equiv i \leq |ciudades| \wedge |ciudades| \leq i \;\longleftrightarrow |ciudades| = i \longrightarrow \neg(i < |ciudades|) \\ &\equiv |ciudades| = i \longrightarrow \neg(i < |ciudades|) \end{aligned} \tag{5}$$

Queda verificado el Teorema de Terminación por lo que podemos concluir que el ciclo termina

Como ambos teoremas se cumplen, **el ciclo es correcto**.

Luego me falta ver que P $\longrightarrow wp\ (S1; S2, P_c)$

wp $(S1; S2, P_c) \equiv wp\,(res := 0; i := 0, P_c) \equiv wp\left(res := 0, def(i) \wedge_L {P_c}_i^0\right) \equiv wp(S_2, res = 0 \wedge 0 = 0))$
$\equiv wp(S_2, \text{true} \wedge_L res = 0)) \equiv def(res) \wedge_L {P_c}_{res}^0 \equiv 0 = 0 \equiv \text{true}$

Luego, como P $\longrightarrow$ true es tautología, siempre se cumple

Queda demostrado que **la tripla {P}S1;S2;S3;S4{Q} es válida**

2.2. Demostracion res > 50000

Se utiliza la misma definición del ejercicio anterior para P, B, fv. Se redefinen:
I $\equiv$ $\{0 \leq i \leq |ciudades| \wedge \text{res} = \sum\limits_{j=0}^{i-1} ciudades[j].habitantes \wedge (\exists i : \mathbb{Z})(0 \leq i < |ciudades|) \wedge_L ciudades[i].habitantes > 50{,}000 \wedge (\forall j : \mathbb{Z})$ (
$(0 \leq j < |ciudades| \longrightarrow_L ciudades[j].habitantes \geq 0))$
)}
$Q \equiv \{\text{res} = \sum_{i=0}^{|ciudades|-1} ciudades[i].habitantes \wedge res > 50000\}$
Se procede de manera similar al ejercicio anterior probando por monotonía la validez del programa y la tripla de Hoare.
Resta chequear la validez del ciclo con el nuevo invariante y postcondicion.
Teorema del invariante
$P_c \longrightarrow I$
Como se probo en el punto anterior, las implicancias para $0 \leq i < |ciudades|$ y res $= \sum\limits_{j=0}^{i-1} ciudades[j].habitantes$
se analiza lo agregado al invariante. Como lo agregado esta presente en P_c queda probado trivialmente que $P_c \longrightarrow I$
{I $\wedge B\}$S$\{I\} \longleftrightarrow I \wedge B \longrightarrow$ *wp(S, I)*
Lo agregado al invariante no afecta a la implicancia. Luego vale que: $I \wedge B \longrightarrow$ *wp(S, I)*

{I $\wedge \neg B\} \longrightarrow$Q$_c$
$I \wedge \neg B \equiv 0 \leq i \leq |ciudades| \wedge res = \sum\limits_{j=0}^{i-1} ciudades[j].habitantes \wedge (\exists i : \mathbb{Z})(0 \leq i < |ciudades|) \wedge_L ciudades[i].habitantes > 50{,}000$
$\wedge (\forall j : \mathbb{Z})\ (0 \leq j < |ciudades| \longrightarrow_L ciudades[j].habitantes \geq 0) \wedge \neg\,(i < |ciudades|)$
$\equiv 0 \leq i \leq |ciudades| \wedge res = \sum\limits_{j=0}^{i-1} ciudades[j].habitantes \wedge (\exists i : \mathbb{Z})(0 \leq i < |ciudades|) \wedge_L ciudades[i].habitantes > 50{,}000$
$\wedge (\forall j : \mathbb{Z})\ (0 \leq j < |ciudades| \longrightarrow_L ciudades[j].habitantes \geq 0) \wedge i \geq |ciudades|$
$\equiv i = |ciudades| \wedge res = \sum\limits_{j=0}^{i-1} ciudades[j].habitantes \wedge (\exists i : \mathbb{Z})(0 \leq i < |ciudades|) \wedge_L ciudades[i].habitantes > 50{,}000$
$\wedge (\forall j : \mathbb{Z})\ (0 \leq j < |ciudades| \longrightarrow_L ciudades[j].habitantes \geq 0)$
$\equiv res = \sum\limits_{j=0}^{|ciudades|-1} ciudades[j].habitantes \wedge (\exists i : \mathbb{Z})(0 \leq i < |ciudades|) \wedge_L ciudades[i].habitantes > 50{,}000$
$\wedge (\forall j : \mathbb{Z})\ (0 \leq j < |ciudades| \longrightarrow_L ciudades[j].habitantes \geq 0)$

Sea k una posición en la sucesión *ciudades* que cumple con la precondición de tener mas de 50000 habitantes
$\sum\limits_{j=0}^{k-2} ciudades[j].habitantes$ Para los elementos a la izquierda de la k-esima posición
n para el numero de habitantes de la k-esima posición
$\sum\limits_{j=k+1}^{i-1} ciudades[j].habitantes$ Para los elementos a la derecha de la k-esima posición
Luego
$res = \sum\limits_{j=0}^{|ciudades|-1} ciudades[j].habitantes \wedge (\exists i : \mathbb{Z})(0 \leq i < |ciudades|) \wedge_L ciudades[i].habitantes > 50{,}000$
$\wedge (\forall j : \mathbb{Z})\ (0 \leq j < |ciudades| \longrightarrow_L ciudades[j].habitantes \geq 0) \equiv res = \sum\limits_{j=0}^{|ciudades|-1} ciudades[j].habitantes \wedge$
$res > 50000$
{I $\wedge \neg B\} \longrightarrow$Q$_c$
Las implicancias del teorema de terminación no se ven afectadas por los cambios en el invariante y la postcondicion. Luego siguen
Habiendo analizado todos los cambios y siguiendo los razonamientos del ejercicio anterior,
queda demostrado que el resultado devuelto es mayor a 50000