Trabajo práctico 1

Especificación y WP

12 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` $\{(|ciudades| > 0) \wedge_L noHayNombresRepetidos(ciudades)\}$
`asegura` { $(\forall i : \mathbb{Z})$ (
$(0 \leq i < |ciudades|) \longrightarrow_L ((ciudades[i]_1 > 50000) \longrightarrow (ciudades[i] \in res))$
) }
}
`pred` `noHayNombresRepetidos` (lista: $seq\langle Ciudad\rangle$) {
$(\forall i, j : \mathbb{Z})$ (
$(0 \leq i, j < |lista|) \longrightarrow_L (lista[i]_0 \neq lista[j]_0)$
)
}

1.2. sumaDeHabitantes

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

`requiere` $\{(|menoresDeCiudades| > 0) \wedge_L$
$(|menoresDeCiudades| = |mayoresDeCiudades|) \wedge noHayNombresRepetidos(menoresDeCiudades) \wedge$
$noHayNombresRepetidos(mayoresDeCiudades) \wedge ((\forall i : \mathbb{Z})$ (
$(0 \leq i < |menoresDeCiudades|) \longrightarrow_L (\exists! j : \mathbb{Z})$ (
$(0 \leq j < |menoresDeCiudades|) \wedge_L (menoresDeCiudades[i]_0 = mayoresDeCiudades[j]_0)$
)
))}
`asegura` $\{(\forall n, m : \mathbb{Z})$ (
$(0 \leq n, m < |menoresDeCiudades|) \longrightarrow_L ((menoresDeCiudades[n]_0 = mayoresDeCiudades[m]_0) \longrightarrow_L$
$((res[n]_1 = menoresDeCiudades[n]_1 + mayoresDeCiudades[m]_1) \wedge$
$(res[n]_0 = menoresDeCiudades[n]_0)))$
)}
}

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` $\{(0 \leq desde, hasta < |distancias|) \wedge matrizCuadradaSimetrica(distancias)\}$
`asegura` $\{res = \text{true} \leftrightarrow (\exists p : seq\langle\mathbb{Z}\rangle)$ (
$(|p| > 1) \wedge_L ((p[0] = desde) \wedge (p[|p| - 1] = hasta) \wedge (\forall k : \mathbb{Z})$ (
$(0 \leq k < |p| - 1) \longrightarrow_L (distancias[p[k]][p[k+1]] > 0)$
))
)}
}
`pred` `matrizCuadradaSimetrica` (matriz: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
$(\forall i, j, k : \mathbb{Z})$ (
$((0 \leq i, j, k < |matriz|) \wedge_L (0 \leq i, j < |matriz[k]|)) \longrightarrow_L$
$((i = j) \longrightarrow (matriz[i][j] = 0)) \wedge (matriz[i][j] = matriz[j][i])$
)
}

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` $\{(n > 0 \ \wedge matrizCuadradaSimetrica(conexión)) \wedge_L (\forall i, j : \mathbb{Z})$ (
$(0 \leq i, j < |conexión|) \longrightarrow_L (0 \leq conexión[i][j] \leq 1)$
$) \wedge (C_0 = conexión)\}$
`asegura` $\{(\exists p : seq\langle seq\langle seq\langle \mathbb{Z}\rangle\rangle\rangle)$ (
$(p[0] = C_0) \longrightarrow_L (\forall i, j, k : \mathbb{Z})$ (
$(((0 < k < n) \wedge (0 \leq i, j < |conexión|)) \longrightarrow_L (p[k][i][j] = multEntreMatrices(p[k-1], p[k-1])))$ $\wedge_L$
$(conexión[i][j] = multEntreMatrices(p[n-1], p[n-1]))$
)
)}

}

`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` $\{matrizCuadradaSimetrica(distancias) \wedge (0 \leq origen, destino < |distancias|)\}$
`asegura` {
$(existeCamino(distancias, origen, destino) \wedge_L$
$(menorCamino(distancias, origen, destino) = res)) \vee$
$((\neg existeCamino(distancias, origen, destino) \vee (origen = destino)) \longrightarrow_L res = [\ ])\}$
}
`aux todosLosCaminos` (distancias : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, origen : $\mathbb{Z}$, destino : $\mathbb{Z}$) : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$ = $(\exists! m : seq\langle seq\langle \mathbb{Z}\rangle\rangle)$ (
$(|m| > 1) \wedge_L (\exists p : seq\langle \mathbb{Z}\rangle)$ (
$((|p| > 1) \wedge_L ((p[0] = origen) \wedge (p[|p|-1] = destino) \wedge (\forall k : \mathbb{Z})$ (
$(0 \leq k < |p| - 1) \longrightarrow_L (distancias[p[k]][p[k+1]] > 0)$
$))) \longrightarrow_L (p \in m)$
)
) ;
`aux menorCamino` (distancias : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, origen : $\mathbb{Z}$, destino : $\mathbb{Z}$) : $seq\langle \mathbb{Z}\rangle$ = $(\exists p : seq\langle \mathbb{Z}\rangle)$ (
$(\exists i : \mathbb{Z})$ (
$(0 \leq i < |todosLosCaminos(distancias, origen, destino)|) \wedge_L (\forall j : \mathbb{Z})$ (
$((0 \leq j < |todosLosCaminos(distancias, origen, destino)|)) \longrightarrow_L$
$\Big(((\sum_{n=1}^{|todosLosCaminos(distancias,origen,destino)[i]|-1} distancias[n-1][n]) \leq$
$(\sum_{n=1}^{|todosLosCaminos(distancias,origen,destino)[j]|-1} distancias[n-1][n])) \longrightarrow$
$(p = todosLosCaminos(distancias, origen, destino)[i])$
)
)
)
;
`aux existeCamino` (distancias : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, desde : $\mathbb{Z}$, hasta : $\mathbb{Z}$) : Bool = true $\longleftrightarrow (\exists p : seq\langle \mathbb{Z}\rangle)$ (
$(|p| > 1) \wedge_L ((p[0] = desde) \wedge (p[|p|-1] = hasta) \wedge (\forall k : \mathbb{Z})$ (
$(0 \leq k < |p| - 1) \longrightarrow_L (distancias[p[k]][p[k+1]] > 0)$
))
);

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.$P_c \longrightarrow wp\{S3, Q_c\}$

Siendo que $Q_c$coincide con el final del código, $Q_c \equiv Q$
Esto permite demostrar que P $\longrightarrow wp${S1;S2;S3,Q} es verdadera por lo cual la Tripla de Hoare es válida

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. $I \wedge \neg B \longrightarrow Q_c$
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)* $\equiv wp(\textbf{S1; S2}, I)$
  $\equiv wp(\textbf{S1}, \text{wp}(\textbf{S2,I}))$
  $\equiv wp(\textbf{S1};$ i:=i+1 $, I)$
  $\equiv wp($res:= res + ciudades[j].habitantes; i:=i+1$, I)$

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

Ahora pruebo I $\wedge$ B $\longrightarrow$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 $\wedge$ ¬ B

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

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

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

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

Teorema de Terminación

$$\begin{aligned} &4.\{I \wedge B \wedge fv = V_o\}\mathbf{S}\{fv < V_o\} \\ &5.(I \wedge fv \leq 0) \longrightarrow \neg B \end{aligned} \tag{1}$$

Elijo como función variante:

$$fv = |ciudades| - i \tag{2}$$

- Verifico 4.

$$\begin{aligned} &\{I \wedge B \wedge fv = V_o\}\mathbf{S}\{fv < V_o\} \longleftrightarrow \{I \wedge B \wedge fv = V_o\} \longrightarrow wp(S, fv < V_o) \\ &wp(S, fv < V_o) \equiv wp(res := res + ciudades[i].habitantes; i := i+1, |ciudades| - i < V_o) \\ &\equiv wp(res := res + ciudades[i].habitantes, wp(i : i+1; |ciudades| - i < V_o)) \\ &\equiv wp(res := res + ciudades[i].habitantes, |ciudades| - 1 - i < V_o) \\ &\text{Puedo ignorar la res ya que no es relevante en este caso} \\ &\equiv |ciudades| - 1 - i < |ciudades| - i \\ &-1 < 0 \end{aligned} \tag{3}$$

- Verifico 5.

$$\begin{aligned} &(I \wedge fv \leq 0) \longrightarrow \neg B \\ &(I \wedge fv \leq 0) \equiv 0 \leq i \leq |ciudades| \wedge \text{res} = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge |ciudades| - i \leq 0 \\ &\text{Ignoro res} \\ &\equiv i \leq |ciudades| \wedge |ciudades| \leq i \longleftrightarrow |ciudades| = i \longrightarrow \neg(i < |ciudades|) \\ &\equiv |ciudades| = i \longrightarrow \neg(i < |ciudades|) \end{aligned} \tag{4}$$

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 $(S1; S2, P_c) \equiv wp\,(res := 0; i := 0, P_c) \equiv wp\,\big(res := 0, def(i) \wedge_L P_{c\,i}^{\,0}\big) \equiv wp(S_2, res = 0 \wedge 0 = 0))$
$\equiv wp(S_2, \text{true} \wedge_L res = 0)) \equiv def(res) \wedge_L P_{c\,res}^{\,0} \equiv 0 = 0 \equiv$ true

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

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

2.2. Demostracion res > 50000

Se utiliza la misma definición del ejercicio anterior para P, B, fv. Se redefinen:
I $\equiv \{0 \leq i \leq |ciudades| \wedge res = \sum\limits_{j=0}^{i-1} ciudades[j].habitantes \wedge (\exists h : \mathbb{Z})(0 \leq h < |ciudades|) \wedge_L ciudades[h].habitantes > 50{,}000 \wedge (\forall j : \mathbb{Z})$ (
$(0 \leq j < |ciudades|) \longrightarrow_L (ciudades[j].habitantes \geq 0)$
)}
$Q \equiv \{\text{res} = \sum_{i=0}^{|ciudades|-1} ciudades[i].habitantes \wedge res > 50000\}$
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 $P_c \longrightarrow I$
$\{\text{I} \wedge B\}S\{I\} \longleftrightarrow I \wedge B \longrightarrow wp(S, I)$
Lo agregado al invariante no afecta a la implicancia. Luego vale que: $I \wedge B \longrightarrow wp(S,\ I)$

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

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
$res = \sum\limits_{j=0}^{|ciudades|-1} ciudades[j].habitantes \wedge (\exists i : \mathbb{Z})(0 \leq i < |ciudades|) \wedge_L ciudades[i].habitantes > 50{,}000$
$\wedge (\forall j : \mathbb{Z})\ (0 \leq j < |ciudades| \longrightarrow_L ciudades[j].habitantes \geq 0) \equiv res = \sum\limits_{j=0}^{|ciudades|-1} ciudades[j].habitantes \wedge$
$res > 50000$
$\{\text{I} \wedge \neg B\} \longrightarrow \text{Q}_c$

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