Trabajo practico 1

Especificacion y WP

9 de octubre de 2024 — Algoritmos y Estructuras de Datos

X-force

Integrante	LU	Correo electrónico
Krivonosoff, Thiago	310/24	thiagokribas@gmail.com
Pellitero, Agustin	185/24	agustinignaciopelli@gmail.com
Miguel, Facundo	702/24	facumiguel4025@gmail.com
Montenegro, Ulises	477/24	ulinicolasmonte@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. Punto 1

1.1. Ejercicio 1

`proc grandesCiudades` (in ciudades : $seq\langle Ciudad\rangle$) : $seq\langle Ciudad\rangle$
 `requiere` $\{noRepetidos(ciudades) \wedge noHabitantesNegativos(ciudades)\}$
 `asegura` $\{|res| \leq |ciudades|\}$
 `asegura` $\{(\forall elem : Ciudad)\ ((elem.habitantes > 50000 \wedge elem \in ciudades) \leftrightarrow elem \in res)\}$

`pred noRepetidos` (in ciudades: $seq\langle Ciudad\rangle$) {
$(\forall i, j : \mathbb{Z})\ (0 \leq i, j < |ciudades|) \longrightarrow_L (i \neq j \longrightarrow ciudades[i].nombre \neq ciudades[j].nombre)$
}
`pred noHabitantesNegativos` (in ciudades: $seq\langle Ciudad\rangle$) {
$(\forall i : \mathbb{Z})\ (0 \leq i < |ciudades|) \longrightarrow_L ciudades[i].habitantes >= 0$
}

1.2. Ejercicio 2

`proc sumaDeHabitantes` (in menoresDeCiudades: $seq\langle Ciudad\rangle$, in mayoresDeCiudades: $seq\langle Ciudad\rangle$) : $seq\langle Ciudad\rangle$
 `requiere` $\{noRepetidos(menoresDeCiudades) \wedge noRepetidos(mayoresDeCiudades)\}$
 `requiere` $\{noHabitantesNegativos(menoresDeCiudades) \wedge noHabitantesNegativos(mayoresDeCiudades)\}$
 `requiere` $\{|menoresDeCiudades| = |mayoresDeCiudades| \wedge_L$
 $mismosElementos(menoresDeCiudades, mayoresDeCiudades)\}$
 `asegura` $\{|res| = |mayoresDeCiudades|\}$
 `asegura` $\{(\forall elem : Ciudad)\ ((elem \in res) \leftrightarrow$
 $(\forall i : \mathbb{Z})\ (0 \leq i < |res|) \longrightarrow_L (\exists j : \mathbb{Z})\ (0 \leq j < |res|)) \wedge_L$
 $mayoresDeCiudades[i].nombre = menoresDeCiudades[j].nombre \longrightarrow$
 $(res[i].nombre = mayoresDeCiudades[i].nombre \wedge$
 $res[i].habitantes = mayoresDeCiudades[i].habitantes + menoresDeCiudades[j].habitantes)\}$

`pred mismosElementos` (in s1: $seq\langle Ciudad\rangle$, in s2: $seq\langle Ciudad\rangle$) {

$(\forall i : \mathbb{Z})\ (0 \leq i < |s1|) \longrightarrow_L (\exists j : \mathbb{Z})\ (0 \leq j < |s1|) \wedge_L s1[i].nombre = s2[j].nombre$
}

1.3. Ejercicio 3

`proc hayCamino` (in distancia: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, in desde: $\mathbb{Z}$, in hasta: $\mathbb{Z}$) : Bool
 `requiere` $\{esCuadrada(distancia) \wedge_L filaIgualColumna(distancia) \wedge_L diagonalCero(distancia)\}$
 `requiere` $\{0 \leq desde < |distancia|\}$
 `requiere` $\{0 \leq hasta < |distancia|\}$
 `requiere` $\{matrizTodosPositivos(distancia)\}$
 `asegura` $\{res = True \leftrightarrow (\exists sec : seq\langle\mathbb{Z}\rangle)\ (|sec| > 1) \wedge_L secuenciaEsCamino(distancia, sec, desde, hasta)\}$

`pred secuenciaEsCamino` (in distancia: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, in sec: $seq\langle\mathbb{Z}\rangle$, in desde: $\mathbb{Z}$, in hasta: $\mathbb{Z}$) {
$sec[0] = desde \wedge sec[|sec| - 1] = hasta \wedge (\forall i : \mathbb{Z})\ (0 \leq i < |sec|) \longrightarrow_L$
$0 \leq sec[i] < |distancia| \wedge_L todosConexionAnterior(sec, distancia)$
}
`pred diagonalCero` (in mat: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
$(\forall i, j : \mathbb{Z})\ ((0 \leq i, j < |mat| \wedge i = j) \longrightarrow_L mat[i][j] = 0)$
}
`pred todosConexionAnterior` (in sec: $seq\langle\mathbb{Z}\rangle$, in mat: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
$(\forall i : \mathbb{Z})\ (1 \leq i < |sec| \longrightarrow_L mat[sec[i]][sec[i-1]] \neq 0)$
}
`pred esCuadrada` (in mat: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
$(\forall i : \mathbb{Z})\ (0 \leq i < |mat|) \longrightarrow_L |mat| = |mat[i]|$
}
`pred filaIgualColumna` (in mat: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
$(\forall i, j : \mathbb{Z})\ (0 \leq i, j < |mat|) \longrightarrow_L mat[i][j] = mat[j][i]$
}
`pred matrizTodosPositivos` (in mat: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
$(\forall i, j : \mathbb{Z})\ (0 \leq i, j < |mat|) \longrightarrow_L 0 \leq mat[i][j]$
}

1.4. Ejercicio 4

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

`requiere` $\{1 \leq n\}$

`requiere` $\{esCuadrada(conexion) \wedge_L filaIgualColumna(conexion) \wedge_L diagonalCero(conexion)\}$

`requiere` $\{(\forall i, j : \mathbb{Z})\ (0 \leq i, j < |conexion|) \longrightarrow_L conexion[i][j] \in [0, 1]\}$

`requiere` $\{conexion = C_0\}$

`asegura` $\{(\exists sec : seq\langle seq\langle seq\langle \mathbb{Z}\rangle\rangle\rangle)\ (|sec| = n) \wedge sec[0] = C_0 \wedge conexion = sec[|sec| - 1] \leftrightarrow$ $(\forall i : \mathbb{Z})\ (1 \leq i < |sec|) \longrightarrow_L esCuadrada(sec[i]) \wedge |sec[i]| = |conexion| \longrightarrow_L$ $esLaMultiplicacion(sec[i], C_0, sec[i - 1])\}$

`pred esLaMultiplicacion` (in mat: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, in $\text{mat}_0 : seq\langle seq\langle \mathbb{Z}\rangle\rangle$, in $mat_1 : seq\langle seq\langle \mathbb{Z}\rangle\rangle$)\{

$(\forall i, j : \mathbb{Z})\ (0 \leq i, j < |mat|) \longrightarrow_L mat[i][j] = \sum_{k=0}^{|mat|-1} mat_0[i][k] * mat_1[k][j]$

\}

1.5. Ejercicio 5

`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` $\{esCuadrada(distancias) \wedge_L filaIgualColumna(distancias) \wedge_L diagonalCero(distancias)\}$

`requiere` $\{0 \leq destino, origen < |distancias|\}$

`requiere` $\{matrizTodosPositivos(distancias)\}$

`asegura` $\{(\exists s_1 : seq\langle \mathbb{Z}\rangle)\ (|s_1| > 1) \wedge (res = s_1 \leftrightarrow (\forall s_2 : seq\langle \mathbb{Z}\rangle)\ (|s_2| > 1) \longrightarrow_L$ $(secuenciaEsCamino(distancias, s_1, origen, destino) \wedge secuenciaEsCamino(distancias, s_2, origen, destino) \wedge$ $(longitudCamino(s_1, distancias) \leq longitudCamino(s_2, distancias)))\}$

`asegura` $\{res = [] \leftrightarrow (\forall s : seq\langle \mathbb{Z}\rangle)\ (\neg secuenciaEsCamino(distancias, s, origen, destino))\}$

`aux longitudCamino` (in sec: $seq\langle \mathbb{Z}\rangle$, in distancias: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) : $\mathbb{Z} = \sum_{i=1}^{|sec|-1} distancias[sec[i]][sec[i-1]]$;

2. Punto 2

2.1. Ejercicio 1

Demostramos que la implementación es correcta con respecto a la especificación dada mediante teorema de invariante y teorema de terminación.

Por teorema del invariante primero debemos demostrar los siguientes puntos:

- $P_c \longrightarrow I$
- $\{I \wedge B\}\ S\ \{I\}$
- $(I \wedge \neg B) \longrightarrow Q_c$

Definimos $\text{P}_c \equiv (res = 0 \wedge i = 0)$

Definimos $\text{Q}_c \equiv (\sum_{j=0}^{|\text{ciudades}|-1} \text{ciudades}[j].habitantes = res \wedge i = |\text{ciudades}|)$

Elegimos nuestra invariante.

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

Primer paso Probamos primero la implicación de la precondición del ciclo hacia el invariante:

$\text{P}_c \longrightarrow I$

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

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

$$True \wedge True$$

$$True$$

Segundo paso Ahora probamos que vale la siguiente tripla de Hoare:

{I $\wedge B$} S $\{I\}$

Para probar que esto sea verdadero se debe cumplir {I $\wedge B\} \longrightarrow wp(S, I)$

Hacemos uso del axioma para calcular wp(S, I).

$$wp(S, I) \equiv wp(S_1, wp(S_2, I))$$

$$wp(S_2, I) \equiv def(S_2) \wedge_L I^{i}_{i:=i+1}$$

$$wp(S_2, I) \equiv True \wedge 0 \leq i+1 \leq |\text{ciudades}| \wedge_L \sum_{j=0}^{i} \text{ciudades}[j].habitantes = res$$

Terminamos de definir el wp de S2:

$$wp(S_2, I) \equiv 0 \leq i+1 \leq |\text{ciudades}| \wedge_L \sum_{j=0}^{i} \text{ciudades}[j].habitantes = res$$

Ahora definimos el wp de S1:

$$wp(S_1, wp(S_2, I)) \equiv wp(res := res + \text{ciudades}[i].habitantes, 0 \leq i+1 \leq |\text{ciudades}| \wedge_L \sum_{j=0}^{i} \text{ciudades}[j].habitantes = res)$$

$$wp(S_1, wp(S_2, I)) \equiv def(res + \text{ciudades}[i].habitantes) \wedge_L I^{res}_{res+\text{ciudades}[i].habitantes}$$

$$wp(S_1, wp(S_2, I)) \equiv 0 \leq i < |\text{ciudades}| \wedge_L 0 \leq i+1 \leq |\text{ciudades}| \wedge_L \sum_{j=0}^{i} \text{ciudades}[j].habitantes = res + \text{ciudades}[i].habitantes$$

$$wp(S_1, wp(S_2, I)) \equiv 0 \leq i < |\text{ciudades}| \wedge_L \sum_{j=0}^{i} \text{ciudades}[j].habitantes = res + \text{ciudades}[i].habitantes$$

$$wp(S_1, wp(S_2, I)) \equiv 0 \leq i < |\text{ciudades}| \wedge_L \sum_{j=0}^{i} \text{ciudades}[j].habitantes - \text{ciudades}[i].habitantes = res$$

Ahora queda definido el wp de S:

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

Ahora veo la implicacion del invariante y la guarda hacia wp(S,I):

$$\{I \wedge B\} \longrightarrow wp(S, I) \equiv$$

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

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

Se cancelan los terminos y queda:

$$0 \leq i \leq |\text{ciudades}| \longrightarrow True$$

$$True$$

Tercer paso Ahora probamos que vale la siguiente implicación: ((I $\wedge \neg B) \longrightarrow Q_c$)

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

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

i esta entre las longitudes de ciudades y se cancelan las sumatorias

$$i = |\text{ciudades}| \longrightarrow i = |\text{ciudades}|$$

$$True$$

Ahora por teorema de terminación debemos demostrar que la ejecucion del ciclo siempre termina, nuestra función variante: $\mathbf{F}_v = |\textbf{ciudades}| - i - 1$

Para probar que esto sea verdadero se debe cumplir lo siguiente.

- $\{I \wedge B \wedge F_v = v_0\}S\{F_v < v_0\}$
- $(I \wedge F_v \leq 0) \longrightarrow \neg B$

Primer paso probamos la primera implicación:
$\{$ I $\wedge B \wedge F_v = v_0\}S\{F_v < v_0\}$

$$\{0 \leq i \leq |\text{ciudades}| \wedge_L \sum_{j=0}^{i-1} \text{ciudades}[j].habitantes = res \wedge 0 \leq i < |\text{ciudades}| \wedge F_v = |\text{ciudades}| - i - 1\} \longrightarrow$$

$$wp(S, F_v < v_0)$$

Calculamos la wp para probar $\{$ I $\wedge B \wedge F_v = v_0\} \longrightarrow wp(S, \{F_v < v_0\})$

$$wp(S1, wp(S2, F_v < v_0)) \equiv$$

$$wp(S1, wp(i := i + 1, |\text{ciudades}| - i - 1 < v_0)) \equiv$$

$$wp(S1, def(i := i + 1) \wedge |ciudades| - (i + 1) - 1 < v_0) \equiv$$

$$wp(S1, True \wedge |\text{ciudades}| - i - 2 < v_0) \equiv$$

$$wp(res := res + \text{ciudades}[i].habitantes, |\text{ciudades}| - i - 2 < v_0) =$$

$$def(res := res + ciudades[i].habitantes) \wedge_L |ciudades| - i - 2 < v_0 \equiv$$

$$wp(S, F_v < v_0) \equiv 0 \leq i < |ciudades| \wedge_L |ciudades| - i - 2 < v_0$$

Finalmente la implicación nos queda de la forma
$\{$ I $\wedge B \wedge F_v = v_0\} \longrightarrow wp(S, F_v < v_0)$

$$\{0 \leq i \leq |\text{ciudades}| \wedge_L \sum_{j=0}^{i-1} \text{ciudades}[j].habitantes = res \wedge 0 \leq i < |\text{ciudades}| \wedge |\text{ciudades}| - i - 1 \ = v_0 \longrightarrow$$

$$0 \leq i < |ciudades| \wedge_L |ciudades| - i - 2 < v_0 \equiv$$

$$|ciudades| - i - 2 < |ciudades| - i - 1 \equiv True$$

Segundo paso Probamos la segunda implicación:
(I $\wedge F_v \leq 0) \longrightarrow \neg B$

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

$$0 \leq i \leq |\text{ciudades}| \wedge_L |\text{ciudades}| \leq i + 1 \longrightarrow |\text{ciudades}| - 1 \leq i \leq |\text{ciudades}| \equiv$$

$$|\text{ciudades}| - 1 \leq i \leq |\text{ciudades}| \longrightarrow |\text{ciudades}| - 1 \leq i \leq |\text{ciudades}| \equiv$$

$$True$$

Nos queda probar que la precondición inicial del programa implica la wp entre (i = 0, res = 0) y la P_c

$$P_i \longrightarrow wp(i := 0, wp(res := 0, P_c))$$

$$P_i \longrightarrow wp(i := 0, (def(res) \wedge_L i = 0 \wedge 0 = 0))$$

$$P_i \longrightarrow wp(i := 0, i = 0)$$

$$P_i \longrightarrow def(i) \wedge_L 0 = 0$$

$$P_i \longrightarrow True$$

$$True$$

2.2. Ejercicio 2

Teniendo en cuenta la correctitud del programa demostrada en el 2.1, sabemos que el valor devuelto por el programa, siempre que se cumpla la precondición, tendrá la pinta de:

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

En el procedimiento, se especifica que el parámetro de entrada es de tipo IN, por lo tanto en la precondición y la postcondición, el parámetro *ciudades* mantendrá las mismas características, entre ellas que sus elementos son todos mayores o iguales a 0. Por lo tanto, al tomar un elemento de *ciudades*, obligatoriamente será menor o igual a la sumatoria del total de elementos de *ciudades*, es decir:

$$(\forall i : \mathbb{Z})\ (0 \leq i < |ciudades|) \longrightarrow_L$$

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

Ahora bien, según la postcondición sabemos que res es igual la sumatoria de todos sus elementos, entre los cuales sabemos que al menos alguno de ellos es mayor a 50.000, por lo tanto:

$$(\exists k : \mathbb{Z})\ (0 \leq k < |ciudades|) \wedge_L$$

$$50{,}000 < ciudades[k].habitantes$$

Podemos afirmar entonces que:

$$50{,}000 < ciudades[k].habitantes \leq \sum_{j=0}^{|\text{ciudades}|-1} \text{ciudades}[j].habitantes$$

Y por transitividad concluimos que:

$$50{,}000 < \sum_{j=0}^{|\text{ciudades}|-1} \text{ciudades}[j].habitantes$$