DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Especificación y wp

En búsqueda del camino

9 de septiembre de 2024 Algoritmos y estructuras de datos

Grupo datapath

Integrante	LU	Correo electrónico
Andreone, Joaquin	122/24	`jandreone06@gmail.com`
Comerci, Lucas	818/24	`lukicomerci@gmail.com`
Luis, Theo	130/23	`theoluis44@gmail.com`
Zea, Marcos	405/09	`zea.marcos@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. Especificaciones

`proc grandesCiudades` (in ciudades : $seq\langle Ciudad\rangle$) : $seq\langle Ciudad\rangle$
    `requiere` $\{sinRepetidos(ciudades)\}$
    `asegura` $\{|res| \leq |ciudades| \land sinRepetidos(res)\}$
    `asegura` $\{(\forall i:\mathbb{Z})\ (0 \leq i < |ciudades| \land_L ciudades[i]_1 \geq 50000 \longrightarrow_L ciudades[i] \in res)\}$

`proc sumaDeHabitantes` (in menoresDeCiudades: $seq\langle Ciudad\rangle$, in mayoresDeCiudades: $seq\langle Ciudad\rangle$) : $seq\langle Ciudad\rangle$
    `requiere` $\{sinRepetidos(menoresDeCiudades), sinRepetidos(mayoresDeCiudades),$
    $mismasCiudades(menoresDeCiudades, mayoresDeCiudades)\}$
    `asegura` $\{|res| = |menoresDeCiudades|\}$
    `asegura` $\{(\forall i,j:\mathbb{Z})\ ((0 \leq i < |menoresDeCiudades| \land 0 \leq j < |mayoresDeCiudades|) \land_L$
    $menoresDeCiudades[i]_0 = mayoresDeCiudades[j]_0 \longrightarrow_L (\exists k:\mathbb{Z})\ (0 \leq k < |res| \land$
    $(res[k]_0 = menoresDeCiudades[k]_0 \land res[k]_1 = menoresDeCiudades[i]_1 + mayoresDeCiudades[j]_1)))\}$

`proc hayCamino` (in distancias : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, in desde: $\mathbb{Z}$, in hasta: $\mathbb{Z}$) : Bool
    `requiere` $\{0 \leq desde, hasta < |distancias| \land esSimetrica(distancias) \land tieneDiagonalNula(distancias)\}$
    `asegura` $\{res = \text{true} \leftrightarrow (\exists camino : seq\langle\mathbb{Z}\rangle)\ (esCamino(distancias, desde, hasta, camino))\}$

`proc cantidadCaminosNSaltos` (inout conexion : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, in n: $\mathbb{Z}$)
    `requiere` $\{esMatrizDeConexion(conexion) \land n \geq 1 \land conexion = Conexion_0\}$
    `asegura` $\{mismasDimensiones(conexion, Conexion_0)\}$
    `asegura` $\{(\exists potenciasDeConexion : seq\langle seq\langle seq\langle\mathbb{Z}\rangle\rangle\rangle)\ (|potenciasDeConexion| = n \land$
    $potenciasDeConexion[0] = Conexion_0 \land todasMismasDimensiones(potenciasDeConexion, Conexion_0) \land$
    $sonTodasPotenciasDe(potenciasDeConexion, Conexion_0) \land conexion = potenciasDeConexion[n-1])\}$

`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` $\{0 \leq origen < |distancias| \land 0 \leq destino < |distancias| \land esSimetrica(distancias)\}$
    `asegura` $\{esCamino(distancias, origen, destino, res) \lor res = seq\langle\rangle\}$
    `asegura` $\{(\forall camino : seq\langle\mathbb{Z}\rangle)\ (esCamino(distancias, origen, destino, camino) \longrightarrow_L$
    $longitudDeCamino(res, distancias) \leq longitudDeCamino(camino, distancias))\}$

1.1. Predicados y funciones auxiliares globales

`pred sinRepetidos` (lista: $seq\langle Ciudad\rangle$) {
    $(\forall i:\mathbb{Z})\ (0 \leq i < |lista| \longrightarrow_L (\forall j:\mathbb{Z})\ (0 \leq j < |lista| \land Ciudad[i] = Ciudad[j] \longrightarrow_L j = i))$
}
`pred esSimetrica` (matriz:$seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
    $(\forall i:\mathbb{Z})\ (0 \leq i < |matriz| \longrightarrow_L |matriz[i]| = |matriz|) \land (\forall i,j:\mathbb{Z})\ (0 \leq i,j < |matriz| \land_L matriz[i][j] = matriz[j][i])$
}
`pred tieneDiagonalNula` (matriz: $seq\langle seq\langle\rangle\rangle$) {
    $(\forall i,j:\mathbb{Z})\ (0 \leq i,j < |matriz| \land i = j \longrightarrow_L matriz[i][j] = 0)$
}
`pred mismasCiudades` (in ciudades: $seq\langle Ciudad\rangle$, in otrasCiudades: $seq\langle Ciudad\rangle$) {
    $(\forall i:\mathbb{Z})\ (0 \leq i < |ciudades| \longrightarrow_L (\exists j:\mathbb{Z})\ (0 \leq j < |otrasCiudades| \land_L ciudades[i]_0 = otrasCiudades[j]_0))$
}
`pred esCamino` (distancias: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, desde: $\mathbb{Z}$, hasta: $\mathbb{Z}$, camino: $seq\langle\mathbb{Z}\rangle$) {
    $desde \in camino \land hasta \in camino \land desde \neq hasta \land (\forall i:\mathbb{Z})\ (0 \leq i < |camino| \longrightarrow_L 0 \leq camino[i] < |distancias|) \land$
    $(\forall i:\mathbb{Z})\ (0 \leq i < |camino| - 1 \longrightarrow_L distancias[camino[i]][camino[i+1]] \neq 0)$
}
`pred esMatrizDeConexion` (matriz: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
    $esSimetrica(matriz) \land tieneDiagonalNula(matriz) \land (\forall i,j:\mathbb{Z})\ (0 \leq i,j < |matriz| \land i \neq j \longrightarrow_L$
    $(matriz[i][j] = 1 \lor matriz[i][j] = 0))$
}
`pred esProductoDe` (matrizA: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, matrizB: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, matriz: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
    $mismasDimensiones(matrizA, matrizB) \land mismasDimensiones(matrizA, matriz) \land$
    $(\forall i,j:\mathbb{Z})\ (0 \leq i,j < |matriz| \longrightarrow_L matriz[i][j] = \sum_{k=0}^{|A|-1} matrizA[i][k] \cdot matrizB[k][j])$
}
`pred mismasDimensiones` (matrizA: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, matrizB: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
    $|matrizA| = |matrizB| \land (\forall i:\mathbb{Z})\ (0 \leq i < |matrizA| \longrightarrow_L |matrizA[i]| = |matrizB[i]|)$
}
`pred todasMismasDimensiones` (matrices: $seq\langle seq\langle seq\langle\mathbb{Z}\rangle\rangle\rangle$, matriz: $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) {
    $(\forall i:\mathbb{Z})\ (0 \leq i < |matrices| \longrightarrow_L mismasDimensiones(matrices[i], matriz))$

}
`pred sonTodasPotenciasDe` (matrices: $seq\langle seq\langle seq\langle \mathbb{Z}\rangle\rangle\rangle$, matriz: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) {
$(\forall i : \mathbb{Z})\ (1 \leq i < |matrices| \longrightarrow esProductoDe(matriz, matrices[i-1], matrices[i]))$
}
`aux longitudDeCamino` (camino: $seq\langle \mathbb{Z}\rangle$, distancias: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) : $\mathbb{Z}$ =
$$\sum_{i=0}^{|camino|-2} distancias[camino[i]][camino[i+1]] \,;$$

2. Demostraciones de correctitud

```
S1: res := 0;
S2: i := 0;
S3: while (i < ciudades.length) do
S4:       res := res + ciudades[i].habitantes
S5:       i = i + 1
    endwhile
```

Como se ve en la implementacion de la especificacion, el programa consta solo de un ciclo while, por lo que para demostrar su correctitud debemos usar el "Teorema del Invariante" y el "Teorema de terminación de un ciclo". Se tienen que cumplir estas condiciones:

1. $P_c \Longrightarrow I$
2. $\{I \wedge B\}S\{I\}$
3. $I \wedge \neg B \Longrightarrow Q_c$
4. $\{I \wedge B \wedge v_0 = fv\}S\{fv < v_0\}$
5. $I \wedge fv \leq 0 \Longrightarrow \neg B$

Basado en la especificacion y la implementacion propongo:

- $P_c \equiv \{res = 0 \wedge i = 0\}$
- $B \equiv \{i < |ciudades|\}$
- $S_1 \equiv \{res := res + ciudades[i]_1\}$
- $S_2 \equiv \{i := i + 1\}$
- $Q_c \equiv \{res = \sum_{j=0}^{|ciudades|-1} ciudades[j]_1 \wedge res > 50000\}$
- $I \equiv \{0 \leq i \leq |ciudades| \wedge res = \sum_{j=0}^{i-1} ciudades[j]_1\}$
- $fv = |ciudades| - i$

2.1. $P_c \Longrightarrow I$

Reemplazo:

$$res = 0 \wedge i = 0 \Longrightarrow 0 \leq i \leq |ciudades| \wedge res = \sum_{j=0}^{i-1} ciudades[j]_1$$

$$0 \leq i \leq |ciudades| \equiv 0 \leq 0 \leq |ciudades| \equiv True$$

$$res = \sum_{j=0}^{i-1} ciudades[j]_1 \equiv 0 = \sum_{j=0}^{0-1} ciudades[j]_1 = 0 \equiv True$$

Por lo tanto, se cumple $P_c \Longrightarrow I$

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

Para que la tripla de Hoare sea valida, la precondición debe implicar la “Weakest precondition”del código y la postcondición. Es decir:

$$\{I \wedge B\}S\{I\} \iff \{I \wedge B\} \Longrightarrow_L wp(S, I)$$

$$\begin{aligned}
wp(S, I) \equiv wp(S_1, wp(S_2, I)) \equiv wp(S_1, wp(i := i+1, 0 \leq i \leq |ciudades| \wedge res = \sum_{j=0}^{i-1} ciudades[j]_1)) \\
\equiv wp(S_1, def(i+1) \wedge_L 0 \leq i+1 \leq |ciudades| \wedge res = \sum_{j=0}^{i+1-1} ciudades[j]_1) \\
\equiv wp(res := res + ciudades[i]_1, 0 \leq i+1 \leq |ciudades| \wedge res = \sum_{j=0}^{i} ciudades[j]_1) \\
\equiv def(res) \wedge_L def(ciudades[i]_1) \wedge 0 \leq i+1 \leq |ciudades| \wedge res + ciudades[i]_1 = \sum_{j=0}^{i} ciudades[j]_1 \\
\equiv 0 \leq i < |ciudades| \wedge_L 0 \leq i+1 \leq |ciudades| \wedge res + ciudades[i]_1 = \sum_{j=0}^{i} ciudades[j]_1
\end{aligned}$$

Combino los intervalos de i y resto en ambas partes de la igualdad $ciudades[i]_1$ para transformar el rango de la sumatoria

$$\equiv 0 \leq i < |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j]_1$$

Como $I \wedge B \equiv wp(S, I)$, por lo tanto $\{I \wedge B\} \Longrightarrow wp(S, I)$ que era lo que necesitaba para probar que la tripla de Hoare sea válida.

2.3. $I \wedge \neg B \Longrightarrow Q_c$

Reemplazo:

$$I \wedge \neg B \equiv 0 \leq i \leq |ciudades| \wedge res = \sum_{j=0}^{i-1} ciudades[j]_1 \wedge i \geq |ciudades|$$

Si $i \geq |ciudades| \wedge i \leq |ciudades|$ entonces $i = |ciudades|$

$$i = |ciudades| \wedge res = \sum_{j=0}^{i-1} ciudades[j]_1 \Longrightarrow res = \sum_{j=0}^{|ciudades|-1} ciudades[j]_1 \wedge res > 50000 \equiv Q_c$$

Esto se cumple, ya que por especificación al menos una de ellas es grande, es decir, supera los 50.000 habitantes.

2.4. $\{I \wedge B \wedge v_0 = fv\}S\{fv < v_0\}$

Para que la tripla de Hoare sea válida, la precondición debe implicar la “Weakest precondition”del programa y la postcondicion. Es decir:

$$\{I \wedge B \wedge v_0 = fv\}S\{fv < v_0\} \iff \{I \wedge B \wedge v_0 = fv\} \Longrightarrow wp(S, fv < v_0)$$

Calculo wp:

$$\begin{aligned}
wp(S, fv < v_0) &\equiv wp(S_1, wp(S_2, fv < v_0)) \equiv wp(S_1, wp(i := i+1, |ciudades| - i < v_0)) \\
&\equiv wp(res := res + ciudades[i]_1, def(i+1) \wedge_L |ciudades| - (i+1) < v_0) \\
&\equiv def(ciudades[i]) \wedge_L |ciudades| - i - 1 < v_0 \\
&\equiv 0 \leq i < |ciudades| \wedge_L |ciudades| - i - 1 < v_0
\end{aligned}$$

Y desarrollo la precondición:

$$\{I \wedge B \wedge v_0 = fv\} \equiv 0 \leq i \leq |ciudades| \wedge res = \sum_{j=0}^{i-1} ciudades[j]_1 \wedge i < |ciudades| \wedge |ciudades| - i = v_0$$

Ignoro lo que corresponde a res porque no lo necesito para la demostracion, y combino los rangos de i:

$$0 \leq i < |ciudades| \wedge |ciudades| - i = v_0$$

Entonces:

$$\{I \wedge B \wedge v_0 = fv\} \Longrightarrow wp(S, fv < v_0) \Longleftrightarrow$$

$$0 \leq i < |ciudades| \wedge |ciudades| - i = v_0 \Longrightarrow 0 \leq i < |ciudades| \wedge_L |ciudades| - i - 1 < v_0 \Longleftrightarrow$$

$$|ciudades| - i - 1 < |ciudades| - i \Longleftrightarrow -1 < 0 \equiv True$$

Por lo tanto, la tripla de Hoare es válida.

2.5. $I \wedge fv \leq 0 \Longrightarrow \neg B$

Desarrollo:

$$I \wedge fv \leq 0 \equiv 0 \leq i \leq |ciudades| \wedge res = \sum_{j=0}^{i-1} ciudades[j]_1 \wedge |ciudades| - i \leq 0 \Longrightarrow \neg B$$

Ignoro lo que corresponde a res porque no lo necesito para la demostración:

$$0 \leq i \leq |ciudades| \wedge |ciudades| - i \leq 0 \Longrightarrow i \geq |ciudades| \equiv$$

$$0 \leq i \leq |ciudades| \wedge |ciudades| \leq i \Longrightarrow i \geq |ciudades| \equiv$$

$$i = |ciudades| \Longrightarrow i \geq |ciudades| \equiv True$$