DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico 1: Especificación y WP

3 de octubre de 2024 — Algoritmos y Estructuras de Datos

Grupo VOID

Integrante	LU	Correo electrónico
Barco Viraca, Nadia Belen	1594/21	barconadia38@gmail.com
Sequera, José	637/23	joseenriquesequera@gmail.com
Villarroel, Victoria	1110/21	vicvillarroel30@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. Predicados y auxiliares

pred **sinRepetidos** (ciudades: $seq\langle \mathsf{String} \times \mathbb{Z}\rangle$) {
$(\forall i : \mathbb{Z})\ (0 \leq i < |ciudad| \longrightarrow_L \neg((\exists j : \mathbb{Z})\ (0 \leq j < |ciudad| \wedge_L ciudad[i]_0 = ciudad[j]_0))$
}
pred **esMatrizCuadrada** (distancias: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) {
$(\forall i : \mathbb{Z})\ (0 \leq i < |distancias| \longrightarrow_L |distancias| = |distancias[i]|)$
}
pred **esMatrizSimetrica** (distancias: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) {
$(\forall i, j : \mathbb{Z})\ (0 \leq i, j < |distancias| \longrightarrow_L distancias[i][j] = distancias[j][i]))$
}
pred **estanDentroDelRango** (distancias: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, desde: $\mathbb{Z}$, hasta: $\mathbb{Z}$) {
$0 \leq desde < |distancias| \wedge 0 \leq hasta < |distancias|$
}
pred **diagonalEnCeros** (distancias: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) {
$(\forall i : \mathbb{Z})\ (0 \leq i < |distancias| \longrightarrow_L diatancias[i][i] = 0)$
}
pred **habitantesValidos** (c:$seq\langle \mathsf{String} \times \mathbb{Z}\rangle$) {
$(\forall i : \mathbb{Z})\ (0 \leq i < |c| \longrightarrow_L c[i] \geq 0)$
}
pred **esCamino** (distancias: $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, camino: $seq\langle \mathbb{Z}\rangle$, desde: $\mathbb{Z}$, hasta: $\mathbb{Z}$) {
$|camino| > 1 \wedge_L camino[0] = desde \wedge camino[|camino| - 1] = hasta \wedge (\forall i : \mathbb{Z})\ (0 \leq i < |camino| - 1 \longrightarrow_L 0 \leq camino[i], camino[i+1] < |distancias|) \wedge \text{hayConexion}(distancias, camino[i], camino[i+1])$
}
pred **hayConexion** (distancias:$seq\langle seq\langle \mathbb{Z}\rangle\rangle$, camino:$seq\langle \mathbb{Z}\rangle$, camino[i],camino[i+1]) {
$(\exists j, k, h : \mathbb{Z})\ (0 \leq j, k, h < |distancias| \wedge (distancias[j][k] = camino[i] \wedge distancia[k][h] = camino[i{+}1]) \vee (distancias[k][j] = camino[i] \wedge distancias[j][h] = camino[i+1]))$
}
pred **esMatrizDeOrden1** (conexion : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) {
$(\forall i, j : \mathbb{Z})\ ((0 \leq i, j < |conexion| \longrightarrow_L conexion[i][j] = conexion[j][i] \wedge (conexion[i][j] = 1 \vee conexion[i][j] = 0)))$
}
pred **esMultiplicacionDeMatrices** (in matrizA : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, in matrizB : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$, out matrizC : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) {
$|matrizA| = |matrizB| \wedge (\forall i, j : \mathbb{Z})\ (0 \leq i \leq j < |matrizA| \longrightarrow_L |matrizA[i]| = |matrizA[0]| \wedge |matrizB[i]| = |matrizB[0]|) \wedge_L matrizC[i][j] = \sum_{k=0}^{|matrizA|-1} matrizA[i][k] \times matrizB[k][j]))$
}
aux **distanciaTotal** (camino : $seq\langle \mathbb{Z}\rangle$, distancias : $seq\langle seq\langle \mathbb{Z}\rangle\rangle$) : $\mathbb{Z} = \sum_{k=0}^{|camino|-2} distancias[camino[k]][camino[k+1]]$;

1.2. Procedimientos

1. proc **grandesCiudades** (in ciudades: $seq\langle Ciudad\rangle$) : $seq\langle Ciudad\rangle$

   requiere $\{sinRepetidos(ciudades) \wedge habitantesValidos(ciudades)\}$

   asegura $\{(\forall i : \mathbb{Z})\ (0 \leq i < |res| \longrightarrow_L esCiudad(res[i]) \wedge res[i] \in ciudades) \wedge (\forall j : \mathbb{Z})\ (0 \leq j < |ciudades| \longrightarrow_L (ciudades[j] \in res \leftrightarrow esCiudad(ciudades[j]))))\}$

   asegura $\{sinRepetidos(res)\}$

2. proc **sumaDeHabitantes** (in menoresDeCiudades: $seq\langle Ciudad\rangle$, in mayoresDeCiudades: $seq\langle Ciudad\rangle$) : $seq\langle Ciudad\rangle$

   requiere $\{|menoresDeCiudades| = |mayoresDeCiudades|\}$

   requiere $\{sinRepetidos(menoresDeCiudades) \wedge sinRepetidos(mayoresDeCiudades)\}$

   requiere $\{habitantesValidos(menoresDeCiudades) \wedge habitantesValidos(mayoresDeCiudades)\}$

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

   asegura $\{(\forall k : \mathbb{Z})\ (0 \leq k < |res| \longrightarrow_L (\exists j, h : \mathbb{Z})\ (0 \leq j, h < |menoresDeCiudades| \wedge menoresDeCiudades[j]_0 = mayoresDeCiudades[h]_0 \wedge res[i]_1 = menoresDeCiudades[j]_1 + mayoresDeCiudades[h]_1 \wedge res[i]_0 = mayoresDeCiudad$

   asegura $\{|res| = |menoresDeCiudades| \wedge sinRepetidos(res)\}$

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

    `requiere` $\{esMatrizCuadrada(distancias) \wedge esMatrizSimetrica(distancia) \wedge diagonalEnCero(distancias)\}$

    `requiere` $\{estanDentroDelRango(distancias, desde, hasta)\}$

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

    `asegura` $\{res = \text{true} \leftrightarrow (\exists camino : seq\langle \mathbb{Z}\rangle)\ (esCamino(distancias, camino, desde, hasta))\}$

    `asegura` $\{distanciasValidas(camino)\}$

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

    `requiere` $\{esMatrizCuadrada(conexion) \wedge esMatrizSimetrica(conexion) \wedge diagonalEnCeros(conexion)\}$

    `requiere` $\{esMatrizConSoloUnosYCeros(conexion)\}$

    `requiere` $\{n > 0 \wedge conexion = C_0\}$

    `asegura` $\{(\exists conexiones : seq\langle seq\langle seq\langle \mathbb{Z}\rangle\rangle\rangle)(\forall i, j : \mathbb{Z})\ ((0 \leq i, j < |conexion|) \longrightarrow_L |conexiones[i]| = |C_[0]| \wedge esMtarizCuadrada(conexiones[i] \wedge esMatrizSimetrica(conexiones[i]) \wedge esMultiplicacionDeMatrices(conexiones[i-1], conexiones[0], conexiones[i])) \wedge conexion = conexiones[n-1]\}$

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` $\{esMatrizCuadrada(distancias)\}$

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

    `requiere` $\{estanDentroDelRango(distancias, origen, destino)\}$

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

    `asegura` $\{(\exists camino : seq\langle \mathbb{Z}\rangle)\ (esCamino(distancias, camino, origen, destino) \wedge (\forall camino' : seq\langle \mathbb{Z}\rangle)\ (esCamino(camin$ $distanciaTotal(camino) \leq distanciaTotal(camino') \wedge res = camino)) \vee \neg(\exists camino : seq\langle \mathbb{Z}\rangle)\ (esCamino(distancias, ca$

2. Demostraciones de correctitud

2.1.

Para probar que el programa cumple con su especificación dada una precondición P y una postcondición Q, si siempre que el programa comienza en un estado que cumple P, el programa termina su ejecución, y en el estado final se cumple Q. Evidenciemos esto con una tripla de Hoare {P} S {Q}.

Queremos ver que

1. $P \Longrightarrow wp(S1, Pc)$
2. $Pc \Longrightarrow wp(whileBdoS, Qc)$
3. $Qc \Longrightarrow wp(S3, Q)$

Por monotonía podemos decir que $\boxed{P \Longrightarrow wp(S1; while...; S3, Q)}$ es verdadera.

Elegimos los predicados necesarios
$P \equiv (\exists i : \mathbb{Z})\ ((0 \leq i < |ciudades|) \wedge_L ciudades[i].habitales > 50{,}000)) \wedge (\forall i : \mathbb{Z})\ (0 \leq i < |ciudades| \longrightarrow_L ciudades[i].habitales \geq 0)$

$S1 \equiv \{res = 0; i = 0\}$

$Pc \equiv res = 0 \wedge i = 0 \wedge P$

Demostración de $P \Longrightarrow wp(S1, Pc)$**.**

$$wp(S1, Pc) \equiv wp(res = 0; i = 0, Pc) \overset{Ax,3}{\equiv} wp(res = 0, wp(i = 0, Pc)) \overset{Ax,1}{\equiv} wp(res = 0, def(i = 0) \wedge_L Pc|_{i=0}^{i})$$

$$\equiv wp(res = 0, 0 = 0 \wedge res = 0 \wedge P) \equiv wp(res = 0, true \wedge res = 0 \wedge P) \equiv wp(res = 0, res = 0 \wedge P)$$

$$\overset{Ax,1}{\equiv} def(res = 0) \wedge_L |res = 0 \wedge P|_{res=0}^{res} \equiv true \wedge_L 0 = 0 \wedge P \equiv P$$

Entonces podemos decir que

$$P \Longrightarrow P$$

□

Demostración de $P_c \Longrightarrow wp(whileBdoS, Q_c)$**.**

Al querer obtener esta wp no tenemos garantía del que el ciclo termine. Por eso utilizaremos el teorema de la terminación dado que contiene una función variante que decrece en cada iteraci/on, que nos garantiza que el ciclo terminara y usaremos el teorema del invariante para probar la correctitud del ciclo

- $P_c \Rightarrow I$
- $\{I \wedge B\}S\{I\}$
- $I \wedge \neg B \Rightarrow Q_c$
- $\{I \wedge B \wedge v_0 = f_v\}\ S\ \{f_v < v_0\}$
- $I \wedge f_v \leq 0 \Rightarrow \neg B$

Con las primeras tres nos aseguramos la corrección parcial del ciclo utilizando el teorema del invariante. Luego con 4-5 podemos probar que el ciclo termina por el teorema de la terminación.
Hallemos un invariante adecuado para que se cumpla antes de cada iteración(1), después de cada iteración(2) y al salir del bucle(3). Para esto necesitamos P_c, B, Q_c, f_v

- $I \equiv 0 \leq i \leq |ciudades| \wedge res = \sum_{j=0}^{i-1} ciudades[j].habitantes$
- $P_c \equiv i = 0 \wedge res = 0 \wedge P$
- $Q_c \equiv i = |ciudades| \wedge res = \sum_{i=0}^{|ciudades|-1} ciudades[i].habitantes$
- $B_c \equiv i < |ciudades|$
- $f_v \equiv |ciudades| - i$

Demostración $P_c \Rightarrow I$**.**

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

$$i = 0 \rightarrow 0 \leq i \leq |ciudades| \equiv 0 \leq 0 \leq |ciudades| \equiv true$$

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

entonces vale que

$$P \Longrightarrow true \equiv true$$

□

Demostración $\{I \wedge B\}S\{I\}$
. *Para probar que esta tripla es cierta* $\{I \wedge B\}S\{I\}$ *es equivalente a* $I \wedge B \rightarrow wp(S, I)$ *entonces*

$$wp(S, I) \equiv wp(S1; S2, I) \overset{Ax,3}{\equiv} wp(S1, wp(S2, I))$$

$$wp(S2, I) \overset{Ax,1}{\equiv} def(i = i + 1) \wedge_L I|_{i=i+1}^{i}$$

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

$$wp(S1, I') \equiv wp(res = res{+}ciudades[i].habitantes, I') \overset{Ax,1}{\equiv} def(res) \wedge def(ciudades) \wedge 0 \leq i < |ciudades| \wedge_L I'|_{res=res+ciuadades[i].h}^{res}$$

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

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

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

Ahora veamos si se cumple la implicación

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

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

$$\equiv True$$

$\square$

Demostración $I \wedge \neg B \Rightarrow Q_c$**.**

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

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

$$\Longrightarrow$$

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

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

Demostración $(I \wedge \neg B \wedge v_0 = f_v)S(f_v < v_0)$
. *Decimos que*

$$(I \wedge B \wedge v_0 = f_v)S(f_v < v_0)$$

usamos su equivalente

$$(I \wedge B \wedge v_0 = f_v) \Longrightarrow wp(S, f_v < v_0) entonces$$

$$wp(S, f_v < v_0) \overset{Ax,3}{\equiv} wp(s1, wp(s2, f_v < v_0)) \equiv wp(res + ciudades[i].habitantes, wp(i := i+1, f_v < v_0))$$

$$\overset{Ax,1}{\equiv} wp(res + ciudades[i].habitantes, def(i+1) \wedge_L |ciudades| - (i+1) < v_0)$$

$$\equiv wp(res + ciudades[i].habitantes, |ciudades| - (i+1) < v_0) \overset{Ax,1}{\equiv} def(res + ciudades[i].habitantes) \wedge_L$$

$$|ciudades| - (i+1) < v_0 \overset{Ax,1}{\equiv} 0 \leq i < |ciudades| \wedge_L |ciudades| - (i+1) < v_0$$

nos queda que

$$0 \leq i < |ciudades| \wedge_L |ciudades| - (i+1) < |ciudades| - i \equiv 0 \leq i < |ciudades| \wedge_L -1 < 0$$

$$\equiv 0 \leq i < |ciudades| \wedge_L True \equiv \boxed{0 \leq i < |ciudades|}$$

Nos queda ver que

$$I \wedge B \wedge V_0 = f_v \Longrightarrow 0 \leq i < |ciudades|$$

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

$$\Longrightarrow 0 \leq i < |ciudades|$$

Nos alcanza con ver

$$0 \leq i \leq |ciudades| \Longrightarrow 0 \leq i < |ciudades|$$

$\square$

Demostración $I \wedge f_v \leq 0 \Rightarrow \neg B$.

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

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

$\square$

Probamos que Pc $\Longrightarrow$ wp(while B do S, Qc) y como Q $\Longrightarrow$ Q_c. Y ya probado que el ciclo termina podemos decir que el programa es correcto respecto a su especificación

2.2.

Para demostrar que $res > 50000$ *decidimos que es necesario agregar una nueva clausula a la post condición del programa, a su vez creemos que es importante que esta afirmación sobre res, sea cierta antes de que se ejecute el programa, mientras se ejecuta y al finalizar el programa*

$$I_N \equiv 0 \leq i \leq |ciudades| \wedge res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge (\exists k : \mathbb{Z})(0 \leq k < |ciudades| \wedge ciudades[k].habitantes > 50000))$$

$$Q_N \equiv i = |ciudades| \wedge res = \sum_{i=0}^{|ciudades|-1} ciudades[i].habitantes \wedge res > 50{,}000$$

Probaremos nuevamente la correctitud del ciclo con estas nuevos cambios

Demostración $P_c \Rightarrow I$. *Esto ya lo hemos probado, retomando nos queda que*

$$P \Longrightarrow P \equiv True$$

Demostración $\{I \wedge B\}S\{I\}$
.

$$wp(S, I) \equiv wp(S1; S2, I) \overset{Ax,3}{\equiv} wp(S1, wp(S2, I))$$
$$wp(S2, I) \overset{Ax,1}{\equiv} def(i := i + 1) \wedge_L I|_{i=i+1}^{i}$$

$$\equiv 0 \leq i{+}1 \leq |ciudades| \wedge res = \sum_{j=0}^{i+1-1} ciudades[j].habitantes \wedge (\exists k : \mathbb{Z})(0 \leq k < |ciudades| \wedge ciudades[k].habitantes > 50000))$$

$$\equiv 0 \leq i{+}1 \leq |ciudades| \wedge res = \sum_{j=0}^{i} ciudades[j].habitantes \wedge (\exists k : \mathbb{Z})(0 \leq k < |ciudades| \wedge ciudades[k].habitantes > 50000)) \equiv I'$$

$$wp(S1, I') \equiv wp(res{+}ciudades[i].habitantes, I') \overset{Ax,1}{\equiv} def(res) \wedge def(ciudades) \wedge 0 \leq i < |ciudades| \wedge_L I'|_{res=res+ciuadades[i].h}^{res}$$

$$\equiv 0 \leq i < |ciudades| \wedge_L 0 \leq i + 1 \leq |ciudades| \wedge res + ciudades[i].habitantes = \sum_{j=0}^{i} ciuades[j].habitantes$$
$$\wedge (\exists k : \mathbb{Z})(0 \leq k < |ciudades| \wedge ciudades[k].habitantes > 50000))$$

$$\equiv 0 \leq i < |ciudades| \wedge res = \sum_{j=0}^{i} ciuades[j].habitantes - ciudades[i].habitantes$$
$$\wedge (\exists k : \mathbb{Z})(0 \leq k < |ciudades| \wedge ciudades[k].habitantes > 50000)$$

$$\boxed{\equiv 0 \leq i < |ciudades| \wedge res = \sum_{j=0}^{i-1} ciuades[j].habitantes \wedge (\exists k : \mathbb{Z})(0 \leq k < |ciudades| \wedge ciudades[k].habitantes > 50000)}$$

Ahora veamos si se cumple la implicación

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

$$(\exists k:\mathbb{Z})(0 \leq k < |ciudades| \land ciudades[k].habitantes > 50000) \land i < |ciudades|$$

$$\equiv 0 \leq i < |ciudades| \land res = \sum_{j=0}^{i-1} ciudades[j].habitantes \land i < |ciudades| \land (\exists k:\mathbb{Z})(0 \leq k < |ciudades| \land$$

$$ciudades[k].habitantes > 50000)$$

$$\Longrightarrow$$

$$0 \leq i < |ciudades| \land res = \sum_{j=0}^{i-1} ciuades[j].habitantes \land (\exists k:\mathbb{Z})(0 \leq k < |ciudades| \land ciudades[k].habitantes > 50000)$$

$$\equiv True$$

□

Demostración $I \land \neg B \Rightarrow Q_c$**.** *Retomando la demo con I y Qc nuevas nos queda que*

$$(\exists k:\mathbb{Z})(0 \leq k < |ciudades| \land ciudades[k].habitantes > 50000) \Longrightarrow res > 50000$$

Esta implicaión es cierta ya que sabemos que la cantidad de habitantes de una ciudad es mayor o igual cero y a la vez contiene al menos una ciudad con mayor de 50000 habitantes.

Demostración $(I \land \neg B \land v_0 = f_v)S(f_v < v_0)$
. *Esta demostración es análoga a la que contiene Qc e I no modificadas.*

Demostración $I \land f_v \leq 0 \Rightarrow \neg B$**.** *Esta demostración es análoga a la que contiene Qc e I no modificadas.*