Trabajo práctico 1

Especificación y WP

12 de septiembre de 2024 Algoritmos y Estructura de Datos

SinGrupo4

Integrante	LU	Correo electrónico
Algañaraz, Franco	001/01	francoarga10@gmail.com
Illescas, Marcos	390/14	marcosillescas90@gmail.com
Bahamonde, Matias	694/21	matubaham@gmail.com
Marión, Ian Pablo	004/01	ianfrodin@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

`pred habitantesPositivos` (in s: $seq\langle Ciudad\rangle$) {
$(\forall j:\mathbb{Z})(0 \leq j < |s| \longrightarrow_L s[j].habitantes \geq 0)$
}
`pred noCiudadesRepetidas` (in s: $seq\langle Ciudad\rangle$) {
$(\forall i:\mathbb{Z})(\forall j:\mathbb{Z})(0 \leq i < j < |s| \longrightarrow_L s[i].nombre \neq s[j].nombre)$
}

1.1. grandesCiudades

`proc grandesCiudades` (in ciudades: $seq\langle Ciudad\rangle$) : $seq\langle Ciudad\rangle$
`requiere` $\{habitantesPositivos(ciudades) \land noCiudadesRepetidas(ciudades)\}$
`asegura` $\{|res| = CantidadCiudadesMayor50000(ciudades)\ \land$
$(\forall i:\mathbb{Z})(0 \leq i < |res| \longrightarrow_L (\exists j:\mathbb{Z})(0 \leq j < |ciudades| \land_L (res[i] = ciudades[j] \land ciudades[j].habitantes > 50000)))\}$

`aux CantidadCiudadesMayor50000` (in s: $seq\langle Ciudad\rangle$) : $\mathbb{Z}$ =

$\sum_{i=0}^{|s|-1}$ if $s[i].habitantes > 50000$ then 1 else 0 fi;

1.2. sumaDeHabitantes

`proc sumaDeHabitantes` (in menoresDeCiudades: $seq\langle Ciudad\rangle$, in mayoresDeCiudades: $seq\langle Ciudad\rangle$) : $seq\langle Ciudad\rangle$
`requiere` $\{|menoresDeCiudades| = |mayoresDeCiudades|\ \land$
$mismosNombres(menoresDeCiudades, mayoresDeCiudades)\ \land$
$habitantesPositivos(menoresDeCiudades)\ \land\ habitantesPositivos(mayoresDeCiudades)\ \land$
$noCiudadesRepetidas(menoresDeCiudades)\ \land\ noCiudadesRepetidas(mayoresDeCiudades)\}$
`asegura` $\{|res| = |menoresDeCiudades|\ \land$
$(\forall i:\mathbb{Z})(0 \leq i < |res| \longrightarrow_L (\exists j:\mathbb{Z})(\exists k:\mathbb{Z})(0 \leq j < k < |menoresDeCiudad|\ \land_L$
$siCoincidenNombresSumoHabitantes(res, menoresDeCiudades, mayoresDeCiudades, i, j, k)))\}$

`pred mismosNombres` (in s: $seq\langle Ciudad\rangle$, in t: $seq\langle Ciudad\rangle$) {
$(\forall i:\mathbb{Z})(0 \leq i < |s| \longrightarrow_L (\exists j:\mathbb{Z})(0 \leq j < |t| \land_L s[i].nombre = t[j].nombre))$
}
`pred siCoincidenNombresSumoHabitantes` (in s: $seq\langle Ciudad\rangle$, in t: $seq\langle Ciudad\rangle$, in r: $seq\langle Ciudad\rangle$, in i: $\mathbb{Z}$, in j: $\mathbb{Z}$, in k: $\mathbb{Z}$) {
$(s[i].nombre = t[j].nombre\ \land\ s[i].nombre = r[k].nombre) \longrightarrow s[i].habitantes = t[j].habitantes + r[k].habitantes$
}

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` $\{0 \leq desde < |distancias| \land 0 \leq hasta < |distancias|\}$
`asegura` $\{res = \text{true} \iff existeCamino(distancia, desde, hasta)\}$

`pred existeCamino` (in distancias : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, in desde : $\mathbb{Z}$, in hasta : $\mathbb{Z}$) {
$(\exists camino : seq\langle\mathbb{Z}\rangle)\ (|camino| \geq 2 \land camino[0] = desde \land camino[|camino|-1] = hasta \land (\forall i:\mathbb{Z})(0 \leq i < |camino|-1 \rightarrow distancias[camino[i]][camino[i+1]] > 0))$
}

1.4. cantidadCaminosNSaltos

`proc cantidadCaminosNSaltos` (in conexion : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, in n : $\mathbb{Z}$) : $seq\langle\mathbb{Z}\rangle$
`requiere` $\{n \geq 1 \land conexion = C_0\}$
`asegura` $\{conexion = {C_0}^n\}$

1.5. caminoMínimo

`proc caminoMinimo` (in origen : $\mathbb{N}$, in destino : $\mathbb{Z}$, in distancias : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$) : $seq\langle\mathbb{Z}\rangle$
`requiere` $\{0 \leq origen < |distancias| \land 0 \leq destino < |distancias|\}$
`asegura` $\{(res = [\ ] \iff existeCamino(distancias, origen, destino) \lor origen = destino) \land (res \neq [\ ] \implies (\forall camino : seq\langle\mathbb{Z}\rangle)(camino[0] = origen \land camino[|camino|-1] = destino \land sumaDistancias(res, distancias) \leq sumaDistancias(camino, distancias)))\}$

aux sumaDistancia (in camino : $seq\langle\mathbb{Z}\rangle$, in 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

2.1. Demostrar que la implementación es correcta con respecto a la especificación.

proc poblacionTotal (in ciudades: $seq\langle Ciudad\rangle) : \mathbb{Z}$

requiere $\{(\exists j : \mathbb{Z})(0 \leq j < |ciudades| \wedge_L ciudades[j].habitantes > 50000) \wedge$
$(\forall k : \mathbb{Z})(0 \leq k < |ciudades| \longrightarrow_L ciudades[i].habitantes \geq 0) \wedge$
$(\forall m : \mathbb{Z})(\forall n : \mathbb{Z})(0 \leq m < n < |ciudades| \longrightarrow_L ciudades[m].nombre \neq ciudades[n].nombre)\}$

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

$S_1 : res = 0$
$S_2 : i = 0$
while (i < ciudades.length) do
$S_3 : res = res + ciudades.[i].habitantes$
$S_4 : i = i + 1$
endwhile

$P \equiv A \wedge B \wedge C$
$A \equiv (\exists j : \mathbb{Z})(0 \leq j < |ciudades| \wedge_L ciudades[j].habitantes > 50000)$
$B \equiv (\forall k : \mathbb{Z})(0 \leq k < |ciudades| \longrightarrow_L ciudades[i].habitantes \geq 0)$
$C \equiv (\forall m : \mathbb{Z})(\forall n : \mathbb{Z})(0 \leq m < n < |ciudades| \longrightarrow_L ciudades[m].nombre \neq ciudades[n].nombre)$

$Pc \equiv res = 0 \ \wedge i = 0 \ \wedge A \wedge B \wedge C$

$B \equiv i < |ciudades|$

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

$f_v \equiv |ciudades| - i$

Como el programa finaliza al terminar el ciclo, podemos asumir que:

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

Para demostrar la correctitud del programa basta ver que son válidas las siguientes triplas de Hoare:

$\{Pc\}\ S_3; S_4\{Qc \equiv Q\}\ y\ \{P\}S_1; S_2\{Pc\}$

2.1.1. $Pc \implies wp(S_3; S_4, Qc)$ (Teorema del Invariante y Terminación)

2.1.1.1 $Pc \implies I$

$$Pc \equiv res = 0 \ \wedge i = 0 \ \wedge A \wedge B \wedge C \implies 0 \leq 0 \leq |ciudades| \wedge_L 0 = \sum_{j=0}^{-1} ciudades[j].habitantes \equiv$$

$$0 \leq |ciudades| \wedge_L 0 = 0 \equiv$$

$$True \ \wedge_L \ True \equiv$$

$$True$$

$$Pc \implies I$$

2.1.1.2 $I \wedge B \implies wp(S_3; S_4, I)$

$wp(S_3; S_4, I) \equiv wp(res = res + ciudades.[i].habitantes,\ wp(i = i + 1, I)) \equiv$

$wp(res = res + ciudades.[i].habitantes,\ def(i + 1) \wedge_L I^i_{i+1}) \equiv$

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

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

$def(res + ciudades.[i].habitantes) \wedge_L (I')^{res}_{res+ciudades.[i].habitantes} \equiv$

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

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

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

$$0 \leq i < |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \implies$$
$$0 \leq i < |ciudades| \wedge_L \sum_{j=0}^{i-1} ciudades[j].habitantes = \sum_{j=0}^{i-1} ciudades[j].habitantes \equiv$$
$$True \wedge_L True \equiv$$
$$True$$
$$I \wedge B \implies wp(S_3; S_4, I)$$

2.1.1.3 $I \wedge \neg B \implies Qc$

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

$$i = |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \equiv$$
$$res = \sum_{j=0}^{|ciudades|-1} ciudades[j].habitantes \implies$$
$$\sum_{j=0}^{|ciudades|-1} ciudades[j].habitantes = \sum_{j=0}^{|ciudades|-1} ciudades[j].habitantes \equiv$$
$$True \wedge_L True \equiv$$
$$True$$
$$I \wedge \neg B \implies Qc$$

2.1.1.4 $I \wedge B \wedge v_0 = f_v \implies wp(S_3; S_4, f_v < v_0)$

$wp(S_3; S_4, f_v < v_0) \equiv wp(res = res + ciudades[i].habitantes, wp(i = i + 1, |ciudades| - i < v_0)) \equiv$

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

$wp(res = res + ciudades[i].habitantes, True \wedge_L |ciudades| - i - 1 < v_0) \equiv$

$def(res + ciudades[i].habitante) \wedge_L (|ciudades| - i - 1 < v_0)^{res}_{res+ciudades[i].habitante} \equiv$

$0 \leq i < |ciudades| \wedge_L |ciudades| - i - 1 < v_0 \equiv$

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

$$0 \leq i < |ciudades| \wedge_L res = \sum_{j=0}^{i-1} ciudades[j].habitantes \wedge v_0 = |ciudades| - i \implies$$
$$0 \leq i < |ciudades| \wedge_L |ciudades| - i - 1 < |ciudades| - i \equiv$$
$$True \wedge_L True \equiv$$
$$True$$
$$I \wedge B \wedge v_0 = f_v \implies wp(S_3; S_4, f_v < v_0)$$

2.1.1.5 $I \wedge f_v \leq 0 \implies \neg B$

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

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

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

$$i \geq |ciudades| \equiv$$

$$True$$

$$I \wedge f_v \leq 0 \implies \neg B$$

2.1.2. $P \implies wp(S_1; S_2, Pc)$

$wp(S_1; S_2, Pc) \equiv wp(res = 0, wp(i = 0, Pc)) \equiv wp(res = 0, def(0) \wedge_L Pc_0^i) \equiv$

$wp(res = 0, True \wedge_L res = 0 \wedge 0 = 0 \wedge A \wedge B \wedge C) \equiv$

$def(0) \wedge_L (res = 0 \wedge True \wedge A \wedge B \wedge C)_0^{res} \equiv$

$True \wedge_L 0 = 0 \wedge A \wedge B \wedge C \equiv True \wedge A \wedge B \wedge C \equiv A \wedge B \wedge C$

$P \equiv A \wedge B \wedge C \implies A \wedge B \wedge C \equiv True \wedge True \wedge True \equiv True$

$$P \implies wp(S_1; S_2, Pc)$$

El programa es correcto con respecto a su especificación.

2.2. Demostrar que el valor devuelto es mayor a 50.000.