Trabajo práctico 1

Especificación y WP

10 de septiembre 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` {true}
`asegura` { $(\forall i : \mathbb{Z})$ (
$(0 \leq i < |ciudades|) \wedge_L ((ciudades[i] \in res) \longrightarrow_L (ciudades[i]_1 > 50000))$
) }
}

1.2. sumaDeHabitantes

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

`requiere` { $(|menoresDeCiudades| = |mayoresDeCiudades|) \wedge_L ((\forall i, j : \mathbb{Z}_{\geq 0})$ (
$0 \leq i, j < |menoresDeCiudades| \wedge menoresDeCiudades[i]_0 = mayoresDeCiudades[j]_0$
) }
`asegura` {$(\forall n, m : \mathbb{Z})$ (
$(0 \leq n, m < |menoresDeCiudades|) \wedge_L ((menoresDeCiudades[n]_0 = mayoresDeCiudades[m]_0) \wedge$
$(ciudades[n] \in res) \longrightarrow ((ciudades[n]_1 = menoresDeCiudades[n]_1 + mayoresDeCiudades[m]_1) \wedge$
$(ciudades[n]_0 = menoresDeCiudades[n]_0)))$ }
}

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

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

) }
}
`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` { $(\forall i, j : \mathbb{Z})$ (
$(0 \leq i, j < |distancias|) \wedge_L ((i = j) \longrightarrow (distancias[i][j] = 0)) \wedge (distancias[i][j] = distancias[j][i])$
)
`asegura` {(hayCamino(distancias, origen, destino) $\wedge$(($\forall$k: $seq\langle\mathbb{Z}\rangle$, $\exists$p: $seq\langle\mathbb{Z}\rangle$)(esCamino(distancias, origen, destino, p) $\wedge$ (sumaDistancias(distancias, origen, destino, p) $\leq sumaDistancias(distancias, origen, destino, k))) \longrightarrow res = p) \wedge$
(($\neg$hayCamino(distancias, origen, destino) $\vee$ (origen = destino)) $\longrightarrow$res = [ ])
)
}

`pred esCamino` (distancias : $seq\langle seq\langle\mathbb{Z}\rangle\rangle$, origen : $\mathbb{Z}$, destino : $\mathbb{Z}$, p : $seq\langle\mathbb{Z}\rangle$)
$((\forall k : \mathbb{Z})$ (
0$\leq$ i, j, origen, destino < $|p|$) $\wedge_L distancias[p[k]][p[k+1]]$)
`aux sumaDistancias` (distancias : $seq\langle seq\langle\mathbb{Z}\rangle, origen : \mathbb{Z}, destino : \mathbb{Z}, s : seq\langle\mathbb{Z}\rangle\rangle$) : $\mathbb{Z}$ =
;
$\sum_{j=origen}^{destino} distancias[j][j+1]$

2. Demostraciones de correctitud