Trabajo Práctico 1: Especificación de TADs

Cómo funcionan las Blockchains: $BerretaCoin

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Grupo: Ameri

Integrante	LU	Correo electrónico
Rankov, Jorge	714/23	jrankov@dc.uba.ar
Falbo, Tiziana	nnn/nn	nnn@gmail.com
Facundo	nnn/nn	nnn@gmail.com
Bautista	nnn/nn	nnn@gmail.com

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria – (Pabellon I/Planta Baja)
Intendente Guiraldes 2610 – C1428EGA
Ciudad Autónoma de Buenos Aires – Rep. Argentina
Tel/Fax: (+54) 11 4576–3300
http://www.exactas.uba.ar

TAD $BerretaCoin {

obs blockchain: Seq<bloques>

proc nuevoBerretaCoin(): BerretaCoin {
asegura { res.blockchain = <> }
}

proc agregarBloque (inout B: BerretaCoin, in S: bloque) {
requiere { $B = B_0 \wedge (0 < |\text{S.transacciones}| \leq 50) \wedge$ bloqueValido(B, S) $\wedge$ sonTransaccionesValidas(S, B) }

asegura { B.blockchain = B_0.blockchain ++ {S} }
}

proc maximosTenedores (in B: BerretaCoin): Seq<$\mathbb{Z}$> {
asegura { $(\forall i\text{: } \mathbb{Z})(0 \leq i < |res|) \rightarrow$ esUsuario $(res_{[i]}$, B) }

asegura { (∀id: $\mathbb{Z}$) (id ∈ res) → ¬ (∃ otro: $\mathbb{Z}$) (esUsuario (otro, B)) ∧
(montoDeUsuario (otro, B) ≥ montoDeUsuario (id, B)) }

asegura { $(\forall i\text{: } \mathbb{Z})(\forall j\text{: } \mathbb{Z})$ $(0 \leq i < |res|) \wedge (0 \leq j < |res|) \wedge (i \neq j) \rightarrow res_{[i]} \neq res_{[j]}$ }
}

proc montoMedio (in B: BerretaCoin): Float {
requiere { $|\text{B.blockchain}| \neq 0$ }

asegura { res = promedio (B.blockchain) }
}

proc cotizacionAPesos (in cotizaciones: Seq<$\mathbb{Z}$>, in B: BerretaCoin): Seq<$\mathbb{Z}$> {
requiere { (∀C ∈ cotizaciones) (C > 0) }

requiere { |cotizaciones| = |B.blockchain| }

asegura { |res| = |cotizaciones| }

asegura { $(\forall i : \mathbb{Z})(0 \leq i < |\text{res}|) \rightarrow res_{[i]}$ = cotizarBloque (i, cotizaciones, B.blockchain) }
}

Predicados

pred bloqueValido (B: BerretaCoin, S: bloque) {
$(|\text{B.blockchain}| < 3000) \rightarrow \text{S.transacciones}_{[0]}.\text{idComprador} = 0 \land$
$(|\text{B.blockchain}| \geq 3000) \rightarrow \text{S.transacciones}_{[0]}.\text{idComprador} \neq 0 \land$
$(|S.transacciones| \leq 50)$
$\land$
$[(\forall\ \text{transaccion} \in \text{S.transacciones}) \rightarrow [(\text{transaccion.idComprador} \neq \text{transaccion.idVendedor}) \land$
$(\text{transaccion.id} > 0 \land \text{transaccion.idComprador} \geq 0 \land \text{transaccion.idVendedor} > 0 \land \text{transaccion.monto} > 0)] \land$
estaOrdenada(S) $\land$ esIdDeBloqueConsecutivo(B, S)
}

pred estaOrdenada (S: bloque) {
$(\forall i\text{: } \mathbb{Z})\ (0 \leq \text{i} < |\text{S.transacciones}|) \rightarrow (\text{S.transacciones}_{[i]}.\text{idTransaccion} = i)$
}

pred esIdDeBloqueConsecutivo (B: BerretaCoin, S: bloque) {
$(|\text{B.blockchain}| = 0) \rightarrow (\text{S.idBloque} = 0) \land$
$(|\text{B.blockchain}| \neq 0) \rightarrow (\text{S.idBloque} = \text{B.blockchain}[\ |\text{B.blockchain}|-1\ ].\text{idBloque} + 1)$
}

pred sonTransaccionesValidas (S: bloque, B: blockchain) {
$(\forall id :\mathbb{Z})(\forall j :\mathbb{Z})(0 \leq \text{j} < |\text{S.transacciones}|)$ [esUsuario(id, B) $\lor$ esUsuarioDeBloque(id, S)]
$\rightarrow$ montoDeUsuarioHastaTransaccion(id, S, B, j) ≥ 0
}

pred esUsuario (id: $\mathbb{Z}$, B: BerretaCoin) {
$(\exists \text{i: } \mathbb{Z})(\exists \text{j: } \mathbb{Z})\ (0 \leq \text{i} < |\text{b.blockchain}| \land 0 \leq \text{j} < |\text{b.blockchain}_{[i]}.\text{transacciones}|) \land$
$(\text{id} = \text{b.blockchain}_{[i]}.\text{transacciones}_{[j]}.\text{idVendedor})$
}

Auxiliares

aux MontoDeUsuario (id: $\mathbb{Z}$; B: BerretaCoin): $\mathbb{Z}$=

$$\sum_{j=0}^{|bloques|-1} [\sum_{i=0}^{|bloques_{[j]}.transacciones|-1} (ifThenElse(bloques_{[j]}.transacciones_{[i]}.idComprador = id, bloques_{[j]}.transacciones_{[i]}.monto, 0)) + \sum_{i=0}^{bloques_{[j]}.transacciones|-1} (ifThenElse(bloques_{[j]}.transacciones_{[i]}.idVendedor = id, bloques_{[j]}.transacciones_{[i]}.monto, 0))]$$

aux montoDeUsuarioHastaTransaccion (id: $\mathbb{Z}$, S: bloque, B: BerretaCoin, pos: $\mathbb{Z}$) =

montoDeUsuario(id, B) +

$$[\sum_{i=0}^{pos} (ifThenElse(S.transacciones_{[i]}.idComprador = id, bloques_{[j]}.transacciones_{[i]}.monto, 0))$$

+

$$\sum_{i=0}^{pos} (ifThenElse(S.transacciones_{[i]}.idVendedor = id, bloques_{[j]}.transacciones_{[i]}.monto, 0))]$$

aux promedio (bloques: seq<bloques>): Float =

$$\frac{\sum_{j=0}^{|bloques|-1} (\sum_{i=0}^{|bloques_{[j]}.transacciones|-1} (ifThenElse(bloques_{[j]}.transacciones_{[i]}.idComprador \neq 0), bloques_{[j]}.transacciones_{i}.monto, 0))}{\sum_{j=0}^{|bloques|-1} \left(\sum_{i=0}^{|bloques_{[j]}.transacciones|-1} (ifThenElse(bloques_{[j]}.transacciones_{[i]}.idComprador \neq 0,1,0)) \right)}$$

aux cotizarBloque (posicion: $\mathbb{Z}$, cotizaciones: Seq<$\mathbb{Z}$>, blockchain: Seq<bloques>): $\mathbb{Z}$=

$$\left(\sum_{i=0}^{|blockchain[posicion]|-1} blockchain[posicion].transacciones_{[i]}.montos \right) * \text{cotizaciones}[posicion]$$

}

Anotaciones y Cosas a reubicar después

cadenaDeBloques = Seq<Seq<$\mathbb{Z}\times\mathbb{Z}\times\mathbb{Z}\times\mathbb{Z}$>>
bloque = struct(transacciones: seq<struct (idTransaccion:$\mathbb{Z}$, idComprador:$\mathbb{Z}$, idVendedor:$\mathbb{Z}$, monto:$\mathbb{Z}$)>, idBloque:$\mathbb{Z}$)

Procs

proc montosDeUsuarios {
  asegura { $\forall$id $\in$ sinRepetidos (Usuarios (Cripto.blockchain)) $\rightarrow$ id $\in$ res
    $\longleftrightarrow$ (esMaximo (MontoDeUsuario (Cripto.blockchain, id)); Montos (Usuarios (Cripto.blockchain))) }
}

proc Usuarios (in bc: BerretaCoin): Seq<$\mathbb{Z}$> {
  asegura { ($\forall$i: $\mathbb{Z}$) ($0 \leq i < |bc.blockchain|$) }

  asegura { ($\forall$j: $\mathbb{Z}$) ($0 \leq i < |bc.blockchain_{[i]}|$) }

  asegura {$(bc.blockchain_{[i][j][1]} \wedge bc.blockchain_{[i][j][2]}) \in res \wedge$ ningunOtroElem $\in res$ }
}

Preds

pred esTransaccionCreacion (t: Seq<$\mathbb{Z}\times\mathbb{Z}\times\mathbb{Z}\times\mathbb{Z}$>) {
  $t_{[1]} = 0$
}

pred sinRepetirId (ids: Seq<$\mathbb{Z}\times\mathbb{Z}$>) {
  ($\forall$i,j: $\mathbb{N}$) ((0 $\leq$ i < |ids| $\wedge_L$ (0 $\leq$ j < |ids| $\wedge_L$ (j $\neq$ i)) $\rightarrow_L$ $id_{[i][0]} \neq id_{[j][0]}$))
}

pred esMaximo (Monto: $\mathbb{Z}$, Montos: Seq<$\mathbb{Z}$>) {
  ($\forall$i $\in$ Montos) $\rightarrow_L$ Monto $\geq$ i
}

pred esTransaccionValida (b: Seq<$\mathbb{Z}\times\mathbb{Z}\times\mathbb{Z}\times\mathbb{Z}$>, ids: Seq<$\mathbb{Z}\times\mathbb{Z}$>) {
  ($\forall$i,j: $\mathbb{N}$) (j $\leq$ i <$|b|$ $\wedge_L$ (0 $\leq$ j <$|ids|$) $\wedge_L$ ($b_{[i][2]} = ids_{[j][0]}$)) $\rightarrow_L$ ($b_{[i][3]} \leq ids_{[j][1]}$)
}

Auxs

aux Montos (S: Seq<$\mathbb{Z}$>): Seq<$\mathbb{Z}$>=

$$\sum_{j=0}^{|s|-1} (MontoDeUsuario(S_{[i]}))$$

aux sinRepetidos (S: Seq<$\mathbb{Z}$>): Seq<$\mathbb{Z}$>=

$$[S_{[0]}] + \sum_{i=1}^{|s|-1} ifThenElse(S_{[i]} \in SubSeq(S, 0, i-1); \emptyset; [S_{[i]}])$$

aux Usuarios (S:Seq<Seq<$\mathbb{Z}\times\mathbb{Z}\times\mathbb{Z}\times\mathbb{Z}$>>): Seq<$\mathbb{Z}$>=

$$\sum_{i=0}^{|s|-i|s_{[i]}|-1} \sum_{j=0} (S_{[i][j][1]}, S_{[i][j][2]})$$