DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Practico

21 de abril de 2024 — Algoritmos y Estructuras de Datos

Grupo AlgoTango

Integrante	LU	Correo electrónico
Orsi, Lautaro Manuel	689/23	Lautaorsi@gmail.com
Zerbetto De Palma, Gerardo Gabriel	900/22	g.zerbetto@gmail.com
Simoza Sanchez, Valeria Andreina	1027/22	vsimoza.vs@gmail.com
Prieto, Matias	382/23	matiasprieto2003@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. Redistribucion De Los Frutos

proc redistribucionDeLosFrutos (in recursos : $seq\langle\mathbb{Z}\rangle$, in cooperan: $seq\langle\text{Bool}\rangle$) : $seq\langle\mathbb{Z}\rangle$

requiere $\{|recursos| = |cooperan| \land recursosValidos(recursos)\}$

asegura $\{|res| = |recursos| \land_L (\forall i : \mathbb{Z})(0 \leq i < |recursos| \longrightarrow_L \Big(\Big((cooperan[i] = true) \longrightarrow_L res[i] = \frac{fondoMonetario(recursos, cooperan)}{|recursos|}\Big) \land \Big((cooperan[i] = false) \longrightarrow_L res[i] = recursos[i] + \frac{fondoMonetario(recursos, cooperan)}{|recursos|})\Big)\Big)\}$

aux fondoMonetario (recursos:$seq\langle\mathbb{R}\rangle$,cooperan:$seq\langle\text{Bool}\rangle$) : $\mathbb{R} = \sum_{j=0}^{|recursos|-1} \Big(\text{if } \big(cooperan[j] = true\big) \text{ then } \big(recursos[j]\big) \text{ else } \big(0\big) \text{ fi}\Big)$;

pred recursosValidos (recursos:$seq\langle\mathbb{R}\rangle$) {
$(\forall i : \mathbb{Z})(0 \leq i < |recursos| \longrightarrow_L recursos[i] > 0)$
}

Desarrollo 1.1

Para este ejercicio implementamos un aux *fondoMonetario* que calcula —en base a las listas *cooperan* y *recursos*— una sumatoria de la totalidad de recursos que seran redistribuidos al finalizar el paso temporal, sumando al fondo los recursos de aquellos que cooperen y sin sumar los que no.

Empleamos este aux para luego calcular el recurso de cada individuo, si este decidia cooperar su recurso sera la division equitativa y si no cooperaba su recurso sera la plata obtenida mas la division equitativa.

1.2. Trayectoria De Los Frutos Individuales a Largo Plazo

proc trayectoriaDeLosFrutosIndividualesALargoPlazo (inout trayectorias: $seq\langle seq\langle\mathbb{R}\rangle\rangle$, in cooperan: $seq\langle\text{Bool}\rangle$, in apuestas: $seq\langle seq\langle\mathbb{R}\rangle\rangle$, in pagos: $seq\langle seq\langle\mathbb{R}\rangle\rangle$, in eventos: $seq\langle seq\langle\mathbb{N}\rangle\rangle$)

requiere $\{trayectoriasValidas(old(trayectorias))$
$\land\ mismaLongitud(old(trayectorias), pagos, apuestas, eventos, cooperan)$
$\land\ pagosPositivos(pagos)$
$\land\ apuestasValidas(apuestas)$
$\land\ longitugApuestasPagos(apuestas, pagos)$
$\land\ longitudSublistas(pagos)$
$\land\ longitudSublistas(apuestas)$
$\land\ longitudSublistas(eventos)\}$

asegura $\{(\forall i : \mathbb{Z})(0 \leq i < |old(trayectorias)| \longrightarrow_L (trayectorias[i][0] = (old(trayectorias)[i])[0]) \land_L (\forall j : \mathbb{Z})(0 \leq j < |eventos[i]| \longrightarrow_L trayectorias[i][j{+}1] = calculoDeRecursosSegunCooperacion(trayectorias, pagos, apuestas, eventos, cooperan, i, j)))\}$

aux calculoDeRecursosSegunCooperacion (trayectorias,pagos,apuestas:$seq\langle seq\langle\mathbb{R}\rangle\rangle$,eventos:$seq\langle seq\langle\mathbb{N}\rangle\rangle$,cooperan:$seq\langle\text{Bool}\rangle$,individuo:$\mathbb{Z}$,ronda:$\mathbb{Z}$) : $\mathbb{R}$ = if $cooperan[individuo] = true$ then $fondoMonetarioRepartido(trayectorias, pagos, apuestas, eventos, cooperan, ronda)$ else $calculoRecursos(trayectorias[individuo][ronda], pagos[individuo][eventos[individuo][ronda]], apuestas[individuo[eventos[individuo][ronda]]]) + fondoMonetarioRepartido(trayectorias, pagos, apuestas, eventos, cooperan, ronda)$ fi ;

aux calculoRecursos (recurso,pago,apuesta:$\mathbb{R}$) : $\mathbb{R} = recurso * pago * apuesta$;

aux fondoMonetario (trayectorias,pagos,apuestas:$seq\langle seq\langle\mathbb{R}\rangle, eventos : seq\langle seq\langle\mathbb{N}\rangle, cooperan : seq\langle\text{Bool}\rangle, ronda : \mathbb{Z}\rangle\rangle$) : $\mathbb{R} = \sum_{h=0}^{|cooperan|-1}$ if $cooperan[h] = true$ then $calculoRecursos(trayectorias[h][ronda], pagos[h][eventos[h][ronda]], apuestas[h][eventos[h][ronda]])$ else 0 fi ;

aux fondoMonetarioRepartido (trayectorias,pagos,apuestas:$seq\langle seq\langle\mathbb{R}\rangle\rangle$,eventos:$seq\langle seq\langle\mathbb{N}\rangle\rangle$,cooperan:$seq\langle\text{Bool}\rangle$,ronda:$\mathbb{Z}$) : $\mathbb{R} = \frac{fondoMonetario(trayectorias, pagos, apuestas, eventos, cooperan, ronda)}{|cooperan|}$;

aux sumaApuestas (apuestas:$seq\langle seq\langle\mathbb{R}\rangle\rangle$, individuo: $\mathbb{Z}$) : $\mathbb{R} = \sum_{h=0}^{|apuestas[individuo]|-1} apuestas[individuo][h]$;

pred mismaLongitud (trayectorias,pagos,apuestas:$seq\langle seq\langle\mathbb{R}\rangle\rangle$,eventos:$seq\langle seq\langle\mathbb{N}\rangle\rangle$,cooperan:$seq\langle\text{Bool}\rangle$) {
$|trayectorias| = |cooperan| = |apuestas| = |pagos| = |eventos|$
}

pred trayectoriasValidas (trayectorias:$seq\langle seq\langle\mathbb{R}\rangle\rangle$) {

$|trayectorias| > 0 \wedge (\forall i : \mathbb{Z})(0 \leq i < |trayectorias| \longrightarrow_L |trayectorias[i]| = 1 \wedge_L (\forall x : \mathbb{R})(x \in trayectorias[i] \longrightarrow_L x > 0))$

}

`pred pagosPositivos` (pagos:$seq\langle seq\langle \mathbb{R} \rangle\rangle$) {

$(\forall i, j : \mathbb{Z})(0 \leq i < |pagos| \wedge_L (0 \leq j < |pagos[i]| \longrightarrow_L pagos[i][j] > 0))$

}

`pred apuestasValidas` (apuestas:$seq\langle seq\langle \mathbb{R} \rangle\rangle$) {

$(\forall i, j : \mathbb{Z})(0 \leq i < |apuestas| \longrightarrow_L sumaApuestas(apuestas, i) = 1 \wedge_L (0 \leq j < |apuestas[i]| \longrightarrow_L 0 \leq apuestas[i][j] \leq 1))$

}

`pred longitudApuestasPagos` (apuestas,pagos:$seq\langle seq\langle \mathbb{R} \rangle\rangle$) {

$(\forall i : \mathbb{Z})(0 \leq i < |apuestas| \longrightarrow_L |apuestas[i]| = |pagos[i]|)$

}

`pred longitudSublistas` (lista:$seq\langle seq\langle T \rangle\rangle$) {

$(\forall i : \mathbb{Z})(0 \leq i < |lista| \longrightarrow_L (|lista[0]| > 0 \wedge_L |lista[0]| = |lista[i]|))$

}

1.3. Trayectoria Extraña Escalera

`proc trayectoriaExtrañaEscalera` (in trayectoria: $seq\langle \mathbb{R} \rangle$) : Bool

`requiere` $\{|trayectoria| > 0\}$

`asegura` $\{res = true \iff ((|trayectoria| = 1) \vee (|trayectoria| = 2 \wedge \neg(todosIguales(trayectoria)) \vee (|trayectoria| > 2 \wedge hayUnicoMax(trayectoria))))\}$

`pred hayUnicoMax` (Trayectoria: $seq\langle \mathbb{R} \rangle$) {

$CantidadMaximos(Trayectoria) = 1$

}

`pred todosIguales` (lista:$seq\langle \mathbb{R} \rangle$) {

$(\forall i, j : \mathbb{Z})(0 \leq i < |lista| \wedge 0 \leq j < |lista| \wedge i \neq j \longrightarrow_L lista[i] = lista[j])$

}

`aux CantidadMaximos` (lista: $seq\langle \mathbb{R} \rangle$) : $\mathbb{Z} = \sum_{i=1}^{|lista|-2}$ if $\Big(lista[i-1] < lista[i]\Big) \wedge \Big(lista[i] > lista[i+1]\Big)$ then (1) else (0) fi ;

Desarrollo 1.3:

En este ejercicio, utilizamos una separacion en 3 distintos casos, 2 de ellos unicos y uno generalizado, es importante notar que la trayectoria que se recibe es una lista que representa los recursos a medida que avanzan las rondas (o pasos temporales).
El primero, siendo que $|trayectoria|$ es 1 (se juega una ronda) sabemos que sera maximo local pues no tiene vecinos
El segundo, siendo que $|trayectoria|$ es 2 y que, si son distintos, trivialmente alguno es mayor que el otro siendo entonces el maximo local
El ultimo y mas general, dada una secuencia de mas de 2 elementos se busca si efectivamente hay algun numero mayor que sus numeros vecinos y que ademas sea el **unico** con esa propiedad en la secuencia
(Podriamos ahorrarnos el pred hayUnicoMax pero creemos que queda mas legible de esta forma)

1.4. Individuo Decide Si Cooperar O No

`proc individuoDecideSiCooperarONo` (in individuo: $\mathbb{N}$, in recursos: $seq\langle \mathbb{R} \rangle$, inout cooperan: $seq\langle \text{Bool} \rangle$, in apuestas: $seq\langle seq\langle \mathbb{R} \rangle\rangle$, in pagos: $seq\langle \mathbb{R} \rangle$, in eventos: $seq\langle seq\langle \mathbb{N} \rangle\rangle$)

`requiere` $\{mismaLongitud(recursos, pagos, apuestas, eventos, cooperan)$
$\wedge\ individuo < |eventos|$
$\wedge\ pagosPositivos(pagos)$
$\wedge\ apuestasValidas(apuestas)$
$\wedge\ longitudApuestasPagos(apuestas, pagos)$
$\wedge\ longitudSublistas(apuestas)$
$\wedge\ longitudSublistas(eventos)$
$\wedge\ longitudSublistas(pagos)\}$

`asegura` $\{(\exists Trayectorias, TrayectoriasNegadas : seq\langle seq\langle \mathbb{Z} \rangle\rangle)(\exists CooperanNegadaseq\langle \text{Bool} \rangle)$
$\Big(\Big(Trayectorias = ValidarTrayectoria(Trayectorias, cooperan, apuestas, pagos, eventos, recursos)\Big)$
$\wedge\ \Big(\neg(cooperan[individuo]) = cooperanNegada[Individuo]\Big)$
$\wedge\ (\forall i : \mathbb{Z})\Big(0 \leq i < |cooperan| \wedge i \neq Individuo \wedge cooperan[i] = cooperanNegada[i]\Big)$
$\wedge\Big(TrayectoriasNegada = ValidarTrayectorias(TrayectoriasNegada, cooperanNegada, apuestas, pagos, eventos, recursos$

$\longrightarrow_L \Big(\Big((Trayectorias[Individuo][|eventos[0]| - 1] \geq TrayectoriasNegada[Individuo][|eventos[0]| - 1])$

$\wedge\, cooperan[Individuo] = old(cooperan[Individuo]) \Big)$

$\vee \Big((Trayectorias[Individuo][|eventos[0]| - 1] < TrayectoriasNegada[Individuo][|eventos[0]| - 1])$

$\wedge\, cooperan[Individuo] = cooperanNegada[Individuo] \Big) \Big) \}$

`pred ValidarTrayectoria` (trayectorias: $seq\langle seq\langle \mathbb{Z} \rangle\rangle$,cooperan: $seq\langle \mathsf{Bool} \rangle$, apuestas: $seq\langle seq\langle \mathbb{R} \rangle\rangle$, pagos: $seq\langle \mathbb{R} \rangle$,eventos: $seq\langle seq\langle \mathbb{N} \rangle\rangle$, recursos: $seq\langle \mathbb{R} \rangle$) {

$(\forall i : \mathbb{Z})(0 \leq i < |recursos| \longrightarrow_L (trayectorias[i][0] = recursos[i]) \wedge_L$

$(\forall j : \mathbb{Z})(0 \leq j < |eventos[i]| \longrightarrow_L$

$trayectorias[i][j{+}1] = calculoDeRecursosSegunCooperacion(trayectorias, pagos, apuestas, eventos, cooperan, i, j)))$

}

*preds pagosPositivos, apuestasValidas, longitudApuestasPagos, longitudSublistas y calculoDeRecursosSegunCooperacion (con sus dependencias) declarados en ejercicio 1.2

Desarrollo 1.4:

La dificultad de este ejercicio aparece en no tener la secuencia de trayectorias individuales, viendo que calcularlas llevaria mucha complejidad planteamos un cuantificador y predicamos sobre el, analizando las posibles secuencias *trayectorias y trayectoriasNegada* (siendo esta ultima la que corresponderia al caso de la negacion del booleano de cooperacion del individuo) podemos, utilizando los predicados y auxiliares del ejercicio 1.2 y la lista de recursos verificar que efectivamente estas listas son las que deberiamos obtener si calcularamos las trayectorias al largo plazo.

Para validar esta trayectoria comparamos que las primeras posiciones de la trayectoria correspondan a los recursos (basicamente, que el punto de partida de la trayectoria sea correcta) y luego, verificamos que cada posicion N de la lista, sea correspondiente a calcular los recursos N rondas para cada individuo, partiendo de la base de la ronda 0 (recursos). Es importante notar, que para el caso *trayectoriasNegada* utilizaremos la lista *cooperanNegada*, pues el calculo dependera de si el individuo coopera o no.

Al final, podemos observar que utilizando la comparacion de la ultima posicion de *trayectoria* del individuo (basandonos en la cantidad de eventos) con la ultima posicion de la *trayectoriaNegada* asignamos, segun corresponda el valor original (en el primer caso) y el valor negado en el segundo.-

1.5. Individuo Actualiza Apuesta

`proc individuoActualizaApuesta` (in individuo: $\mathbb{N}$, in recursos: $seq\langle \mathbb{R} \rangle$, in cooperan: $seq\langle \mathsf{Bool} \rangle$, inout apuestas: $seq\langle seq\langle \mathbb{R} \rangle\rangle$, in pagos: $seq\langle seq\langle \mathbb{R} \rangle\rangle$, in eventos: $seq\langle seq\langle \mathbb{N} \rangle\rangle$)

`requiere` $\{pagosPositivos(pagos) \wedge mismaLongitud(recursos, cooperan, apuestas, pagos, eventos)$

$\wedge\, apuestasValidas(apuestas) \wedge longitudApuestasPagos(apuestas, pagos)^*\}$

`asegura` $\{(\forall potencialApuesta : seq\langle \mathbb{R} \rangle)$

$\big(esMaximaGanancia(potencialApuesta, recursos[individuo], pagos[individuo], eventos[individuo]) \wedge_L$

$apuestaValida(potencialApuesta) \wedge_L |potencialApuesta| = |apuestas[individuo]|\big)$

$\longrightarrow_L apuestas[individuo] = potencialApuesta\}$

`pred esMaximaGanancia` (apuesta: $seq\langle \mathbb{R} \rangle$, recurso: $\mathbb{R}$, pago: $seq\langle \mathbb{R} \rangle$, evento: $seq\langle \mathbb{N} \rangle$) {

if$\Big($ $(\exists otraApuesta : seq\langle \mathbb{R} \rangle)$

$(RecursoIndividualFinal(otraApuesta, pago, evento, recurso) > RecursoIndividualFinal(apuesta, pago, evento, recu$

Then$(False)$

Else$(True)$

}

`aux RecursoIndividualFinal` (in apuesta: $seq\langle \mathbb{R} \rangle$, in pago: $seq\langle \mathbb{R} \rangle$, in evento: $seq\langle \mathbb{N} \rangle$, in recurso: $\mathbb{R}$) : $\mathbb{R}$ =

$\Big(recurso. \prod_{i=0}^{|evento|-1} \big(pago[evento[i]].apuesta[evento[i]]\big)\Big)$;

`pred apuestaValida` (in apuesta: $seq\langle \mathbb{R} \rangle$) {

$\big(\sum_{i=0}^{|apuesta|-1} apuesta[i]\big) = 1$

}

*pagosPositivos, mismaLongitud, apuestasValidas y longitudApuestasPagos definidos en el ejercicio 1.2

Desarrollo 1.5:

En esta especificacion elegimos predicar acerca del cuantificador sobre las potenciales apuestas que existen para decidir cuales generan una ganancia maxima. El predicado esMaximaGanancia es el que justamente decide si la potencialApuesta es la de mayor ganancia posible y evaluando otraApuesta para verificar que efectivamente no haya ninguna otra que la supere.

2. Demostracion de correctitud

`proc frutoDelTrabajoPuramenteIndividual` (in recurso : $\mathbb{R}$, in apuesta : $\langle$ s : $\mathbb{R}$, c : $\mathbb{R}\rangle$, in pago : $\langle$ s : $\mathbb{R}$, c : $\mathbb{R}\rangle$, in eventos : $seq\langle \mathsf{Bool}\rangle$, out res : $\mathbb{R}$)

`requiere` $\{apuesta_c + apuesta_s = 1 \land pago_c > 0 \land pago_s > 0 \land apuesta_c > 0 \land apuesta_s > 0 \land recurso > 0\}$

`asegura` $\{res = recurso(apuesta_c pago_c)^{\#apariciones(eventos,True)}(apuesta_s pago_s)^{\#apariciones(eventos,False)}\}$

Donde #apariciones(eventos, True) es el auxiliar utilizado en la teorica, y #(eventos, True) es su abreviacion

```
1  res := recurso;
2  i := 0;
3  while (i < |eventos|) do
4      if eventos[i] then
5          res := res * apuesta.c * pago.c;
6      else
7          res := res * apuesta.s * pago.s;
8      endif
9      i := i + 1
10 endwhile
```

Para demostrar que la especificacion es correcta respecto a la implementacion, hay que demostrar la tripla de Hoare del requiere, la implementacion y el asegura.
Primero demostramos la correctitud del ciclo. Para esto planteamos:
$P_c \equiv i = 0 \land res = recurso$
$Q_c \equiv res = recurso(apuesta_c pago_c)^{\#(eventos,True)}(apuesta_s pago_s)^{\#(eventos,False)}$
$I \equiv 0 \leq i \leq |eventos| \land_L res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,i),False)}$
$B \equiv i < |eventos|$
$Fv = |eventos| - i$

Queremos ver que se cumplan:
1)$P_c \longrightarrow I$
2)$\{I \land B\} while....endwhile \{I\}$
3)$I \land -B \longrightarrow Q_c$
4)$\{I \land B \land v_0 = Fv\} while....endwhile \{Fv < v_0\}$
5)$I \land Fv \leq 0 \longrightarrow -B$

1) $P_c \longrightarrow I$:
$(i = 0 \land res = recurso) \longrightarrow (0 \leq i \leq |eventos| \land_L$
$res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,i),False)})$ Aumo que el antecedente es verdadero. Como i = 0, reemplazo i por 0
$res = recurso \longrightarrow (0 \leq 0 \leq |eventos| \land_L$
$res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,0),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,0),False)})$
$\equiv res = recurso \longrightarrow res = recurso(apuesta_c pago_c)^{\#(<>,True)}(apuesta_s pago_s)^{\#(<>,False)}$
$\equiv res = recurso \longrightarrow res = recurso(apuesta_c pago_c)^0(apuesta_s pago_s)^0$
$\equiv res = recurso \longrightarrow res = recurso$
Como siempre es verdadero, se cumple $P_c \longrightarrow I$

2) $\{I \land B\}$ while....endwhile $\{I\}$: Para demostrar esto hay que demostrar $I \land B \longrightarrow wp(while...endwhile, I)$

wp(while...endwhile, I) $\equiv$ wp(if...endif, wp(i := i+1, I) $\equiv wp(if...endif, def(i) \land_L (0 \leq i+1 \leq |eventos| \land_L$
$res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i+1),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,i+1),False)}))$

$\equiv (def(eventos[i]) \land_L (eventos[i] = \text{true} \land wp(res := res * apuesta_c * pago_c, 0 \leq i+1 \leq |eventos| \land_L$
$res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i+1),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,i+1),False)})) \lor (eventos[i] = \text{false} \land$
$wp(res := res * apuesta_s * pago_s, 0 \leq i+1 \leq |eventos| \land_L$
$res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i+1),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,i+1),False)})))$

$\equiv (eventos[i] = \text{true} \land 0 \leq i+1 \leq |eventos| \land_L$
$res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i+1),True)-1}(apuesta_s pago_s)^{\#(subseq(eventos,0,i+1),False)}) \lor (eventos[i] = \text{false} \land$
$0 \leq i+1 \leq |eventos| \land_L res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i+1),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,i+1),False)-1})$

Utilizando la propiedad $P \longrightarrow (Q \lor R) \leftrightarrow (P \longrightarrow Q) \lor (P \longrightarrow R)$ vemos los casos de la implicacion original por separado
Caso eventos[i]=true:

$(0 \leq i \leq |eventos| \wedge_L res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,i),False)} \wedge$
$i < |eventos|) \longrightarrow (eventos[i] = \text{true} \wedge 0 \leq i+1 \leq |eventos| \wedge_L$
$res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i+1),True)-1}(apuesta_s pago_s)^{\#(subseq(eventos,0,i+1),False)})$

$\equiv (0 \leq i < |eventos| \wedge_L res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,i),False)} \longrightarrow$
$(eventos[i] = \text{true} \wedge 0 \leq i+1 \leq |eventos| \wedge_L$
$res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i+1),True)-1}(apuesta_s pago_s)^{\#(subseq(eventos,0,i+1),False)})$

Asumo verdadero el antecedente y reemplazo res
$\equiv 0 \leq i < |eventos| \longrightarrow (eventos[i] = \text{true} \wedge 0 \leq i+1 \leq |eventos| \wedge_L$
$recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,i),False)} =$
$recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i+1),True)-1}(apuesta_s pago_s)^{\#(subseq(eventos,0,i+1),False)})$

Este predicado significa que si el elemento actual es true, contar las apariciones de trueen eventos hasta el elemento anterior va a dar uno menos, lo cual es verdadero

Caso eventos[i]=false:

$(0 \leq i \leq |eventos| \wedge_L res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,i),False)} \wedge$
$i < |eventos|) \longrightarrow (eventos[i] = \text{false} \wedge 0 \leq i+1 \leq |eventos| \wedge_L$
$res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i+1),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,i+1),False)-1})$

Haciendo lo mismo que en el caso anterior:
$\equiv 0 \leq i < |eventos| \longrightarrow (eventos[i] = \text{false} \wedge 0 \leq i+1 \leq |eventos| \wedge_L$
$recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,i),False)} =$
$recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i+1),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,i+1),False)-1})$

Este predicado significa que si el elemento actual es false, contar las apariciones de falseen eventos hasta el elemento anterior va a dar uno menos, lo cual es verdadero
Como ambos casos de la implicacion son verdaderos, la implicacion es verdadera.

3) $I \wedge -B \longrightarrow Q_c$
$0 \leq i \leq |eventos| \wedge_L res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,i),False)} \wedge$
$i \geq |eventos| \longrightarrow res = recurso(apuesta_c pago_c)^{\#(eventos,True)}(apuesta_s pago_s)^{\#(eventos,False)}$

$\equiv i = |eventos| \wedge_L res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,i),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,i),False)} \longrightarrow$
$res = recurso(apuesta_c pago_c)^{\#(eventos,True)}(apuesta_s pago_s)^{\#(eventos,False)}$

Asumo verdadero el antecedente y reemplazo i por $|eventos|$

$\equiv res = recurso(apuesta_c pago_c)^{\#(subseq(eventos,0,|eventos|),True)}(apuesta_s pago_s)^{\#(subseq(eventos,0,|eventos|),False)} \longrightarrow$
$res = recurso(apuesta_c pago_c)^{\#(eventos,True)}(apuesta_s pago_s)^{\#(eventos,False)}$

Como $subseq(eventos, 0, |eventos|) =$ eventos:

$\equiv res = recurso(apuesta_c pago_c)^{\#(eventos,True)}(apuesta_s pago_s)^{\#(eventos,False)} \longrightarrow$
$res = recurso(apuesta_c pago_c)^{\#(eventos,True)}(apuesta_s pago_s)^{\#(eventos,False)}$

Ambos lados de la implicacion son iguales, entoces la implicacion es verdadera

4) $\{I \wedge B \wedge v_0 = Fv\}while....endwhile\{Fv < v_0\}$:
Para demostrar esto hay que demostrar: $(I \wedge B \wedge v_0 = Fv) \longrightarrow wp(while...endwhile, Fv < v_0)$

$wp(while...endwhile, |eventos| - i < v_0) \equiv wp(if...endif, wp(i := i+1, |eventos| - i < v_0)$
$\equiv wp(if...endif, |eventos| - (i+1) < v_0)$

$\equiv def(eventos[i]) \wedge_L ((eventos[i] = \text{true} \wedge wp(res := res * apuesta_c * pago_c, |eventos| - (i+1) < v_0)) \vee$
$(eventos[i] = \text{false} \wedge wp(res := res * apuesta_s * pago_s, |eventos| - (i+1) < v_0)))$
$\equiv (eventos[i] = \text{true} \wedge |eventos| - (i+1) < v_0) \vee (eventos[i] = \text{false} \wedge |eventos| - (i+1) < v_0)$
$\equiv (eventos[i] = \text{true} \vee eventos[i] = \text{false}) \wedge |eventos| - (i+1) < v_0 \equiv |eventos| - (i+1) < v_0$

Volviendo a la equivalencia original:

$(I \wedge B \wedge v_0 = Fv) \longrightarrow wp(while...endwhile, Fv < v_0) \equiv (I \wedge B \wedge v_0 = |eventos| - i) \longrightarrow |eventos| - (i+1) < v_0$

Asumo verdadero el antecedente y reemplazo v_0 por $|eventos| - i$
$\equiv (I \wedge B) \longrightarrow |eventos| - (i+1) < |eventos| - i \equiv (I \wedge B) \longrightarrow |eventos| - i - 1 < |eventos| - i$

Como el consecuente es siempre verdadero, la implicacion es verdadera

5) $I \wedge Fv \leq 0 \longrightarrow -B$

$I \wedge |eventos| - i \leq 0 \longrightarrow i \geq |eventos| \equiv I \wedge |eventos| \leq i \longrightarrow i \geq |eventos|$

La implicacion es siempre verdadera

Con esto queda demostrado, por Teorema del Invariante y Teorema de Terminacion de Ciclo, que vale la siguiente tripla de Hoare:
$\{i = 0, res = recurso\} while....endwhile \{res = recurso(apuesta_c pago_c)^{\#(eventos,True)}(apuesta_s pago_s)^{\#(eventos,False)}\}$

Solo queda demostrar que P_c cumple:

$(apuesta_c + apuesta_s = 1 \wedge pago_c > 0 \wedge pago_s > 0 \wedge apuesta_c > 0 \wedge apuesta_s > 0 \wedge recurso > 0) \longrightarrow$
$wp(res := recurso; i := 0, P_c)$

$\equiv (apuesta_c + apuesta_s = 1 \wedge pago_c > 0 \wedge pago_s > 0 \wedge apuesta_c > 0 \wedge apuesta_s > 0 \wedge recurso > 0) \longrightarrow$
$wp(res := recurso; wp(i := 0, res = recurso \wedge i = 0))$

$\equiv (apuesta_c + apuesta_s = 1 \wedge pago_c > 0 \wedge pago_s > 0 \wedge apuesta_c > 0 \wedge apuesta_s > 0 \wedge recurso > 0) \longrightarrow$
$wp(res := recurso; res = recurso \wedge 0 = 0)$

$\equiv (apuesta_c + apuesta_s = 1 \wedge pago_c > 0 \wedge pago_s > 0 \wedge apuesta_c > 0 \wedge apuesta_s > 0 \wedge recurso > 0) \longrightarrow$
$recurso = recurso \wedge 0 = 0$

Como el consecuente es siempre verdadero, la implicacion es verdadera

Al demostrar esto queda demostrado que la especificacion es correcta respecto de la implementacion