DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Fondo Monetario Común

Trabajo práctico 1: Especificación y WP

21 de abril de 2024 — Algoritmos y Estructuras de Datos

Grupo somtirogla

Integrante	LU	Correo electrónico
Fainsod, Gastón	4/20	gaston.fainsod@gmail.com
Berkowsky, Sasha Nicolas	1158/23	snberkowsky@gmail.com
Gvirtz, Bruno	1173/23	bgvirtz18@gmail.com
Poutays, Manuel	1256/23	manuelpoutays@gmail.com

Corrector: Santiago
Nota: Insuficiente+

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. Introducción

La teoría de juegos es un área de las matemáticas aplicadas que utiliza modelos para estudiar interacciones entre distintos factores dentro de una estructura, haciendo alusión a participantes en juegos de azar. Esta estructura pueden tener comportamientos estocásticos, con el fin de obtener estrategias óptimas, el estudio de dichos modelos puede ser de interes a la hora de elegir las interacciones a realizar. Esta area de la matemática es de interes para comprender comportaminetos por ejemplo en economía, de manera similar, en este trabajo se busca estudiar los comportamientos óptimos en un juego de apuestas en el que los individiduos se ven obligados a apostar en cada paso temporal todos sus recursos. Para un jugador que comienza con un capital inicial w_0, el cual distribuye dicho capital entre n eventos en distintas proporciones $\overline{b} = (b_1, b_2, ..., b_n)$, con distintos pagos por evento $\overline{Q} = (Q_1, Q_2, ..., Q_n)$, el capital resultante al salir el evento $j \in [1, n]$ sera:

$$w_1 = w_0 b_j Q_j \tag{1}$$

Si se realizan k eventos consecutivos, se puede generalizar el capital final como:

$$w_k = w_0 \prod_{i=1}^{k-1} P_i \tag{2}$$

siendo P_i el producto entre el pago y la proporción destinada en el paso i. Otro tema a considerar es cuando hay más de un participante en el juego, los cuales pueden elegir entre contribuir o no a un fondo común. El hecho de participar de un fondo común se entiende como distribuir las ganancias posteriores al evento entre todas las personas que jugaron (entre los que participan y no lo hacen en el fondo común). Por lo cual si se tiene T participantes del fondo común y N participantes en el juego, la distribución del fondo (G) será

$$G = \frac{1}{N} \sum_{i=1}^{T} {w'}_{k_i} \tag{3}$$

donde ${w'}_{k_i}$ es la ganancia que hubiese tenido en principio el participante i en el paso k. Al distribuir las ganancias, se puede redefinir los recursos resultantes (w_{k_i}) obtenidos en Ecuación 2 de la siguiente manera:

$$w_{k_i} = \begin{cases} G & \text{sii el participante i participa del fondo común} \\ G + {w'}_{k_i} & \text{sii el participante i no participa del fondo común} \end{cases} \tag{4}$$

Con el fin de obtener los parametros óptimos entre los participantes, en este trabajo se especificaran distintas funciones necesarias para llevar a cabo la solución del problema.

2. Especificación

2.1. redistribucionDeLosFrutos

Este procedimiento recibe los recursos resultantes para cada individuo y los redistribuye entre todos los participantes por si participan o no del fondo común según lo planteado en Ecuación 3 y Ecuación 4,

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

requiere $\{|cooperan| > 0\}$

requiere $\{esListaDeRecursos(recursos) \land_L |recursos| > 0\}$ Hay cierta redundancia, ya que |cooperan| >0

requiere $\{|cooperan| = |recursos|\}$

asegura $\{|recuros| = |res|\}$

- Mezclan if con predicados.

asegura $\{(\forall i : \mathbb{Z})\ (0 \leq i < |recursos| \longrightarrow_L \text{if } cooperan[i] \text{ then } res[i] = valorCoopera(recursos, cooperan) \text{ else } res[i] = valorCoopera(recursos, cooperan) + recursos[i] \text{ fi})\}$

aux valorCoopera (recursos : $seq\langle\mathbb{R}\rangle$, cooperan : $seq\langle\mathsf{Bool}\rangle$) : $\mathbb{R} = \frac{1}{|recursos|}\sum_{i=0}^{|recursos|-1} \text{if } cooperan[i] = True \text{ then } recursos[i] \text{ else } 0 \text{ fi}$;

pred esListaDeRecursos (recursos: $seq\langle\mathbb{R}\rangle$) {

$(\forall i : \mathbb{Z})$ (

$0 \leq i < |recursos| \longrightarrow_L recursos[i] > 0$

)

}

2.2. trayectoriaDeLosFrutosIndividualesALargoPlazo

Este procedimiento busca obtener los recursos de cada individuo para cada paso de tiempo. Para esto se utiliza el procedimiento redistribucionDeLosFrutos para obtener los valores paso a paso.

proc trayectoriaDeLosFrutosIndividualesALargoPlazo (inout trayectorias : $seq\langle seq\langle\mathbb{R}\rangle\rangle$, in cooperan : $seq\langle\mathsf{Bool}\rangle$, 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 $\{1 = |trayectorias|\}$ - MAL. Interpretaron trayectorias de manera transpuesta.

trayectorias es un listado de trayectorias. Cada uno de sus elementos, es una trayectoria con un único elemento (los recursos iniciales para ese individuo).

requiere $\{esListaDeRecursos(trayectorias[0])\}$

requiere $\{esMatrizDeApuestas(apuestas)\}$

requiere $\{esMatrizDePagos(pagos)\}$

requiere $\{(\forall i : \mathbb{Z})$ (

$0 \leq i < |trayectoria[0]| \longrightarrow_L todosIgualesA(|trayectoria[0]|, \langle|eventos[i]|, |pagos|, |apuestas|, |cooperan|\rangle)$

)} - También interpretaron eventos de manera transpuesta.

requiere $\{(\forall i : \mathbb{Z})$ (

$0 \leq i < |apuestas[0]| \longrightarrow_L todosIgualesA(|apuestas[0]|, \langle|pagos[i]|, |apuestas[i]|\rangle)$

)} |apuestas| (Mal el rango).

asegura $\{(\forall j : \mathbb{Z})\ (0 \leq j < (|trayectorias| - 1) \longrightarrow_L$ (- Tienen que especificar que la longitud no se modifico
$actualizarRec(trayectorias[j+1], trayectorias[j], cooperan, apuestas, pagos, eventos[j])))\}$)

pred actualizarRec (recursosF: $seq\langle\mathbb{R}\rangle$,recursosI: $seq\langle\mathbb{R}\rangle$, cooperan: $seq\langle\mathsf{Bool}\rangle$, apuestas: $seq\langle seq\langle\rangle\rangle\mathbb{R}$, pagos: $seq\langle seq\langle\rangle\rangle\mathbb{R}$, ganadores: $seq\langle\mathbb{N}\rangle$) {

$(\forall i : \mathbb{Z})\ (0 \leq i < |recursosF| \longrightarrow_L \text{if } cooperan[i] = true \text{ then}$ - Mezclan ifs con predicados.

recursosF[i]=valorCoopera(recursosI,cooperan,apuestas,pagos,ganadores) else

- Tienen que especificar que la longitud de la trayectoria es 1 más que antes
- Tienen que especificar que los elementos iniciales siguen siendo los mismos.

recursosF[i]=valorCoopera(recursosI,cooperan,apuestas,pagos,ganadores)+

recursosObtenidos(recursosI[i],apuestas[i],pagos[i],ganadores[i]) fi) }

`aux valorCoopera` (recursos : $seq\langle\mathbb{R}\rangle$, cooperan : $seq\langle$Bool$\rangle$, apuestas : $seq\langle\mathbb{R}\rangle$, pagos : $seq\langle\mathbb{R}\rangle$, ganadores : $seq\langle\mathbb{N}\rangle$) : $\mathbb{R}$ $= \frac{1}{|recursos|}\sum_{i=0}^{|recursos|-1}$ if $cooperan[i] = True$ then

recursosObtenidos(recursos[i],apuestas[i],pagos[i],ganadores[i])] else 0 fi ;

`aux recursosObtenidos` (recurso : $\mathbb{R}$,apuestas : $seq\langle\mathbb{R}\rangle$, pagos : $seq\langle\mathbb{R}\rangle$, ganador: $\mathbb{N}$) : $\mathbb{R}$ $= recurso * apuesta[ganador] * pago[ganador]$;

`pred todosIgualesA` (elemento: $\mathbb{N}$, valores: $seq\langle\mathbb{N}\rangle$) {

$(\forall i : \mathbb{Z})\ (valores[i] = elemento)$

}

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

$(\forall i : \mathbb{Z})$ (

$0 \leq i < |apuestas| \longrightarrow_L (\sum_{j=0}^{|apuestas[0]|-1} apuestas[i][j] = 1 \wedge_L (\forall k : \mathbb{Z})$ (

$0 \leq k < |apuestas[0]| \longrightarrow_L 0 < apuestas[i][k] < 1)$

) (en realidad, las apuestas tmb podrían ser 0 o 1)

)

}

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

$(\forall i : \mathbb{Z})$ (

$0 \leq i < |pagos| \longrightarrow_L (\forall k : \mathbb{Z})$ (

$0 \leq k < |pagos[0]| \longrightarrow_L 0 < pagos[i][k]$

)

)

}

2.3. trayectoriaExtrañaEscalera

Este procedimiento busca registrar si en la trayectoria de un individuo hay un único máximo local.

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

`requiere` $\{|trayectoria| > 1\}$ (interpretación libre, podría haber una trayectoria de longitud 1)

`requiere` $\{esListaDeRecursos(trayectoria)\}$

`asegura` $\{res \leftrightarrow 1 = \sum_{i=1}^{|trayectoria|-2}$ if $(trayectoria[i] > trayectoria[i-1] \wedge_L trayectoria[i] < trayectoria[i+1])$ then 1 else 0 fi $+ sumaExtremos(trayectoria)\}$

`aux sumaExtremos` (trayectoria : $seq\langle\mathbb{R}\rangle$) : $\mathbb{N}$ = (if $trayectoria[0] > trayectoria[1]$ then 1 else 0 fi) + if $trayectoria[|trayectoria|-1] > trayectoria[|trayectoria|-2]$ then 1 else 0 fi) ;

2.4. individuoDecideSiCooperarONo

Este procedimiento compara las ganancias que obtiene un individuo con su elección de cooperar o no en el fondo común con el caso contrario al elegido. Luego cambia o no de elección quedandose con la que genere mayor ganancia individual.

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

requiere $\{cooperan = cooperan_0\}$ ✓ - Falta eventos[i][j] < |apuestas[0]|

requiere $\{0 \leq individuo < |cooperan|\}$ ✓

requiere $\{\cancel{(\forall i : \mathbb{Z})\ (0 \leq i < |recursos| \longrightarrow_L}\ esListaDeRecursos(recursos[i]))\}$ ✗

requiere $\{esMatrizDeApuestas(apuestas)\}$ ✓

requiere $\{esMatrizDePagos(pagos)\}$ ✓

requiere $\{(\forall i : \mathbb{Z})$ (Para que eventos[i] este definido |eventos| >= |cooperan|

$0 \leq i < |cooperan| \longrightarrow_L todosIgualesA(|cooperan|, \langle |eventos[i]|, |pagos|, |apuestas|, |recursos|\rangle)$

)} Me parece que interpretaron eventos al revés. No especifican longitud de eventos.

requiere $\{(\forall i : \mathbb{Z})$ (

$0 \leq i < |apuestas[0]| \longrightarrow_L todosIgualesA(|apuestas[0]|, \langle |pagos[i]|, |apuestas[i]|\rangle)$

)} Para que apuestas[i] esté definido |apuestas| >= |apuestas[0]|, lo que no tiene por qué ser así.

asegura $\{(\exists cooperanAlternativo : seq\langle \text{Bool}\rangle)$ (

$(\forall i : \mathbb{Z})$ (- No especifican la longitud de cooperanAlternativo.

$0 \leq i < |cooperan| \implies$ if $i \neq individuo$ then $cooperanAlternativo[i] = cooperan_0[i]$ else $cooperanAlternativo[i] = \neg cooperan_0[i]$ fi ✓

)

)}

asegura $\{(\exists trayectoria_0 : seq\langle seq\langle \mathbb{R}\rangle\rangle)$ ((Estan usando trayectoria y recursos de manera transpuesta).

$|trayectoria_0| = |eventos| \land_L$ /
$trayectoria_0[0] = recursos \land_L (\forall j : \mathbb{Z})$ (

$0 \leq j < |trayectoria_0| - 1 \implies$
$actualizarRec(trayectoria_0[j+1], trayectoria[j], cooperan_0, apuestas, pagos, eventos[j])$ ✓

)

)}

asegura $\{(\exists trayectoria_1 : seq\langle seq\langle \mathbb{R}\rangle\rangle)$ (

$|trayectoria_1| = |eventos| \land_L$
$trayectoria_1[0] = recursos \land_L (\forall j : \mathbb{Z})$ (

$0 \leq j < |trayectoria_1| - 1 \implies$
$actualizarRec(trayectoria[j+1], trayectoria[j], cooperanAlternativo, apuestas, pagos, eventos[j])$

) No pueden usar variables de un asegura en el asegura que sigue. Las variables ligadas en un Existe (E) o en un Para Todo (V), se encuentran definidas hasta que cierra el paréntesis.

)}

asegura {if $trayectoria_0[|eventos|-1][individuo] > trayectoria_1[|eventos|-1][individuo]$ then $cooperan = cooperan_0$ else $cooperan = cooperanAlternativo$ fi} ✓

2.5. individuoActualizaApuesta

Este procedimiento actualiza la distribución de apuestas a la que mayor cantidad de ganancias genere.

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

`requiere` $\{esListaDeRecursos(recursos)\}$

`requiere` $\{esMatrizDeApuestas(apuestas)\}$

`requiere` $\{esMatrizDePagos(pagos)\}$

`requiere` $\{apuestas = apuestas_0\}$

`requiere` $\{0 \leq individuo < |cooperan|\}$

`requiere` $\{(\forall i : \mathbb{Z})$ (

$0 \leq i < |cooperan| \longrightarrow_L todosIgualesA(|cooperan|, \langle|eventos[i]|, |pagos|, |apuestas|, |recursos|\rangle)$

)} (ver correcciones ejercicio anterior).

`requiere` $\{(\forall i : \mathbb{Z})$ (

$0 \leq i < |apuestas[0]| \longrightarrow_L todosIgualesA(|apuestas[0]|, \langle|pagos[i]|, |apuestas[i]|\rangle)$

)}

`asegura` $\{(\exists trayectoria_0 : seq\langle seq\langle\mathbb{R}\rangle\rangle)$ (

$|trayectoria_0| = |eventos| \land_L$
$trayectoria_0[0] = recursos \land_L (\forall \jmath : \mathbb{Z})$ (

$0 \leq j < |trayectoria_0| - 1 \implies$
$actualizarRec(trayectoria_0[j+1], trayectoria[j], cooperan, apuestas_0, pagos, eventos[j])$

)

)}

`asegura` $\{(\forall apuestasAlternativa : seq\langle seq\langle\mathbb{R}\rangle\rangle)\ (esMatrizDeApuestas(apuestasAlternativa) \longrightarrow_L$
$(\exists trayectoriaAlternativa : seq\langle seq\langle\mathbb{R}\rangle\rangle)$ (- hay que relacioanar el tamaño de las apuestas alternativas con el tamaÑo de la apuesta que viene por parámetro.

$(\forall j : \mathbb{Z})$ (

$0 \leq j < |trayectoriaAlternativa| - 1 \implies$
$(actualizarRecursos(trayectoriaAlternativa[j+1], trayectoriaAlternativa[j], cooperan$
$, apuestasAlternativas, pagos, eventos[j]$

$) \land trayectoria_0[|eventos| - 1][individuo] > trayectoriaAlternativas[|eventos| - 1][individuo])$

)) Acá están comparando contra la apuesta original y preguntandose si es la óptima, en vez de dar una nueva apuesta que sea óptima. (En definitiva, la trayectoria_0 tiene que ser la de apuestas al salir, no al entrar.)

}

`pred todasIgualesMenos` (nueva: $seq\langle seq\langle\mathbb{R}\rangle\rangle$, vieja: $seq\langle seq\langle\mathbb{R}\rangle\rangle$,in posicion: $\mathbb{N}$) {

$(\forall i : \mathbb{Z})$ (

$0 \leq i < |nueva| \longrightarrow_L$ if $i = posicion$ then $vieja[i] \neq nueva[i]$ else $vieja[i] = nueva[i]$ fi

)

}

- Cuando tienen un inout tienen que especificar lo que cambia y lo que se conserva también. En este caso se conserva el tamaÑo de las apuestas y las apuestas de todo otro individuo.

3. Demostraciones de Correctitud

Se quiere demostrar la correctitud del programa de la Figura 1.

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 $eventos$: $seq\langle \text{Bool}\rangle$, 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 $\{res = recurso(apuesta_c\ pago_c)^{\#\texttt{apariciones}(eventos,T)}\ (apuesta_s\ pago_s)^{\#\texttt{apariciones}(eventos,F)}\}$

Donde $\#\texttt{apariciones}(eventos, T)$ es el auxiliar utilizado en la téorica, y $\#(eventos, T)$ es su abreviación.

```
res = recursos
i = 0
while (i < |eventos|) do
    if eventos[i] then
        res = (res * apuesta.c) * pago.c
    else
        res = (res * apuesta.s) * pago.s
    endif
    i = i + 1
endwhile
```

Figura 1: Recursos resultantes a lo largo de los eventos para 100 participantes, en verde se ve el caso donde todos los participantes aportan al fondo comun, en azul y rojo los casos con 1 y 2 desertores .

La descripción de la figura está mal.

Donde definimos P igual al requiere del procedimiento y Q al asegura del mismo. Para esto se utilizará el teorema del invariante, el cual dado un predicado I, se dice que el programa es correcto si cumple las siguientes condiciones:

teorema de corrección de ciclo.

el ciclo del programa

1. $P_c \implies I$
2. $\{I \land B\}S\{I\}$
3. $I \land \neg B \implies Q_c$
4. $\{I \land B \land v_0 = fv\}S\{fv < v_0\}$
5. $I \land fv \leq 0 \implies \neg B$

Siendo

- $P_c = \{i = 0 \land res = recursos\}$
- $I = \{0 \leq i \leq |eventos| \land_L$ $res = recursos(apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),T)} * (apuestas_s * pago_s)^{apariciones(subseq(eventos,0,i),F)}\}$
- $B = \{i < |eventos|\}$
- $Q_c = \{res = recursos * (apuesta.c * pago.c)^{\#apariciones(eventos,True)} * (apuesta.s * pago.s)^{\#apariciones(eventos,False)}\}$
- $fv = |eventos| - i$
- S = {

```
if eventos[i] then
    res = (res * apuesta.c) * pago.c
else
    res = (res * apuesta.s) * pago.s
endif
i = i + 1
```

}

A partir del invariante propuesto I se quieren probar las condiciones del teorema:

1. $P_c \implies I$

   $\{i = 0 \wedge res = recursos\} \implies \{0 \leq i \leq |eventos| \wedge_L res = recursos(apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),T)} * (apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),F)}\}$

   utilizando que i=0 y que la función subseq(eventos,0,i=0) = $\langle\rangle$

   $\{i = 0 \wedge res = recursos\} \implies \{0 \leq i \leq |eventos| \wedge_L res = recursos(apuestas_c * pago_c)^{apariciones(\langle\rangle,T)} * (apuestas_c * pago_c)^{apariciones(\langle\rangle,F)}\}$

   utilizando que $apariciones(\langle\rangle, T) = apariciones(\langle\rangle, F) = 0$

   $\{i = 0 \wedge res = recursos\} \implies \{0 \leq i \leq |eventos| \wedge_L res = recursos * (apuestas_c * pago_c)^0 * (apuestas_c * pago_c)^0 = recursos\}$

   lo cual es cierto dado que si $i = 0 \implies 0 \leq 0 \leq |eventos|$ y res=recursos $\implies$ res=recursos

2. $\{I \wedge B\}S\{I\}$

   S = {

```
if eventos[i] then
    res = (res * apuesta.c) * pago.c
else
    res = (res * apuesta.s) * pago.s
endif
i = i + 1
```

   }
   $B = \{i < |eventos|\}$
   $I = \{0 \leq i \leq |eventos| \wedge_L res = recursos(apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),T)} * (apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),F)}\}$
   $I \wedge B = \{0 \leq i < |eventos| \wedge_L res = recursos(apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),T)} * (apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),F)}$

   para probar esto, se utiliza el teorema que dice que $\{P\}S\{Q\}$ es valida sii $\{P\} \implies wp(S, P)$, lo cual en este problema sería:

   $\{I \wedge B\} \implies wp(S, I)$ ✓

   $\{i < |eventos| \wedge 0 \leq i \leq |eventos| \wedge_L res = recursos(apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),T)} * (apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),F)}\} \implies wp(S, I)$

   $\{0 \leq i < |eventos| \wedge_L res = recursos(apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),T)} * (apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),F)}\} \implies wp(S, I)$

   algún motivo para cambiarle el nombre?

   Definimos res como oldres, por lo tanto queremos ver que Wp(S,I) cumple :

   $Wp(S, I) = Wp(\text{if } ev[i] = True \text{ then } res = (oldres * A_c)P_c \text{ else } res = (oldres * A_s)P_s \text{ fi}; i := i + 1, I)$ ✓

   $Wp(S, I) = Wp(\text{if } ev[i] = True \text{ then } res = (oldres * A_c)P_c \text{ else } res = (oldres * A_s)P_s \text{ fi}, I^i_{i+1}) =$ ✓

   $def(ev[i]) \wedge_L ((ev[i] = True \wedge wp(res = (oldres * A_c)P_c, I^i_{i+1})) \vee (\neg ev[i] = True \wedge wp(res = (oldres * A_s)P_s, I^i_{i+1})))$

Utilizando $def(ev[i]) = 0 \le i < |eventos|$ se simplifica $(0 \le i < |eventos| \wedge 0 \le i+1 \le |eventos|) = 1 \le i+1 \le |eventos|$ y wp(S,I) queda:

$Wp(S,I) = 0 \le i < |eventos| \wedge ((ev[i] = True \wedge wp(res = (oldres * A_c)P_c, I^i_{i+1})) \vee (\neg ev[i] = True \wedge wp(res = (oldres * A_s)P_s, I^i_{i+1})))$

$Wp(S,I) = 0 \le i < |eventos| \wedge ((ev[i] = True \wedge (I^i_{i+1})^{res}_{(oldres*A_c)*P_c})) \vee (\neg ev[i] = True \wedge I^i_{i+1}{}^{res}_{(oldres*A_s)*P_s}))$

$Wp(S,I) = 0 \le i < |eventos| \wedge ((ev[i] = True \wedge_L 0 \le i+1 \le |eventos| \wedge_L$
$(oldres * A_c)P_c = recursos(apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i+1),T)} *$
$(apuestas_s * pago_s)^{apariciones(subseq(eventos,0,i+1),F)}) \vee (\neg ev[i] = True \wedge_L 0 \le i+1 \le |eventos| \wedge_L$
$(oldres * A_s)P_s = recursos(apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i+1),T)} *$
$(apuestas_s * pago_s)^{apariciones(subseq(eventos,0,i+1),F)}))$

Para simplificar veamos que:

$recursos(apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i+1),T)} * (apuestas_s * pago_s)^{apariciones(subseq(eventos,0,i+1),F)}$
$= recursos(apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),T)} * (apuestas_c * pago_c)^{apariciones(subseq(eventos,i,i+1),T)} *$
$(apuestas_s * pago_s)^{apariciones(subseq(eventos,0,i),F)} * (apuestas_s * pago_s)^{apariciones(subseq(eventos,i,i+1),F)} =$
$oldres * (apuestas_c * pago_c)^{apariciones(subseq(eventos,i,i+1),T)} * (apuestas_s * pago_s)^{apariciones(subseq(eventos,i,i+1),F)}$

por lo tanto:

$Wp(S,I) = 0 \le i < |eventos| \wedge ((ev[i] = True \wedge$
$(oldres * A_c)P_c = oldres * (apuestas_c * pago_c)^{apariciones(subseq(eventos,i,i+1),T)} *$
$(apuestas_s * pago_s)^{apariciones(subseq(eventos,i,i+1),F)}) \vee$
$(\neg ev[i] = True \wedge (oldres * A_s)P_s = oldres * (apuestas_c * pago_c)^{apariciones(subseq(eventos,i,i+1),T)} *$
$(apuestas_s * pago_s)^{apariciones(subseq(eventos,i,i+1),F)}$

Lo cual nos dice que en caso de que ev[i]=True sea cierto entonces apariciones(subseq(eventos,i,i+1),T)=1 y se cumpliría $(oldres * A_c)P_c = oldres * (apuestas_c * pago_c)^1 * (apuestas_s * pago_s)^0$, en caso contrario se cumpliría $(oldres * A_s)P_s = oldres * (apuestas_c * pago_c)^0 * (apuestas_s * pago_s)^1$

3. $I \wedge \neg B \implies Q_c$

   $I = \{0 \le i \le |eventos| \wedge_L$
   $res = recursos(apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),T)} * (apuestas_s * pago_s)^{apariciones(subseq(eventos,0,i),F)}\}$,
   $\neg B = \{i \ge |eventos|\}$
   $Q_c = \{res = recursos * (apuesta.c * pago.c)^{\#apariciones(eventos,True)} * (apuesta.s * pago.s)^{\#apariciones(eventos,False)}\}$

   Utilizando que $I \wedge \neg B \implies 0 \le i \le |eventos| \wedge i \ge |eventos| \implies i = |eventos|$
   $i = |eventos| \implies subseq(eventos, 0, i) = eventos$

   podemos reducir $I \wedge \neg B$ a

   $I \wedge \neg B = \{i = |eventos| \wedge res = recursos(apuestas_c * pago_c)^{apariciones(eventos,T)}$
   $*(apuestas_s * pago_s)^{apariciones(eventos,F)} \equiv Q_c\}$

   lo cual sería equivalente a

   $i = |eventos| \wedge Q_c \implies Q_c$

   que es lo que queriamos probar.

4. $\{I \wedge B \wedge v_0 = f_v\}S\{fv < v_0\}$

$S = \{$

```
if eventos[i] then
    res = (res * apuesta.c) * pago.c
else
    res = (res * apuesta.s) * pago.s
endif
i = i + 1
```

$\}$ $fv = |eventos| - i$
$v_o = f_v$ $I = \{0 \leq i \leq |eventos| \wedge_L$
$res = recursos(apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),T)} * (apuestas_s * pagos_s)^{apariciones(subseq(eventos,0,i),F)}\}$
$B = \{i < |eventos|\}$
$(I \wedge B \wedge v_o = f_v) = \{0 \leq i < |eventos| \wedge_L$
$res = recursos(apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),T)} * (apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),F)} \wedge$
$f_v = |eventos| - i \wedge v_o = f_v$

Esto es equivalente a demostrar que $\{I \wedge B \wedge v_0 = fv\} \implies wp(S, f_v < v_o)$ veamos como queda la wp :

$wp(if....endif; i := i + 1, |eventos| - i < v_o)$
$wp(if....endif, wp(i := i + 1, |eventos| - i < v_o))$
$wp(if...endif, (|eventos| - i)^{i}{}_{i+1} < v_o)$

$(eventos[i] = True \wedge wp(res = (res * apuesta.c) * pago.c, |eventos| - (i + 1) < v_o)) \vee (eventos[i] = False \wedge wp(res = (res * apuesta.s) * pago.s, |eventos| - (i + 1) < v_o))$

$(eventos[i] = True \wedge (|eventos| - (i + 1))^{res}{}_{(res*apuesta.c)*pago.c} < v_o) \vee$
$(eventos[i] = False \wedge (|eventos| - (i + 1))^{res}{}_{(res*apuesta.s)*pago.s} < v_o)$

$(eventos[i] = True \wedge (|eventos| - (i + 1)) < v_o) \vee (eventos[i] = False \wedge (|eventos| - (i + 1)) < v_o)$

$(eventos[i] = True \vee eventos[i] = False) \wedge (|eventos| - (i + 1)) < v_o$

$(|eventos| - (i + 1)) < v_o$. Hasta acá cáculo wp, de ahora en más la implicación.

Como $f_v = v_0$ equivale a —eventos— - i, reemplazamos v_0 con esa expresión:

$(|eventos| - (i + 1)) < |eventos| - i$
$-(i + 1) < -i$
$i + 1 > i$
Lo cual es verdadero. Por lo tanto, demostramos que:
$\{I \wedge B \wedge v_0 = fv \implies wp(S, f_v < v_o)$

5. $I \wedge fv \leq 0 \implies \neg B$

$fv = |eventos| - i \leq 0 \implies |eventos| \leq i$
$I = \{0 \leq i \leq |eventos| \wedge_L$
$res = recursos(apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),T)} * (apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),F)}\}$
$\neg B = \{i \geq |eventos|\}$
Utilizando $|eventos| \leq i \wedge 0 \leq i \leq |eventos| \implies \{i = |eventos|\}$ se obtiene :

$\{i = |eventos| \wedge_L$
$res = recursos(apuestas_c * pago_c)^{apariciones(subseq(eventos,0,i),T)} * (apuestas_s * pagos_s)^{apariciones(subseq(eventos,0,i),F)}\}$
$\implies \{i \geq |eventos|\}$

Lo cual es cierto dado que $\{i = |eventos|\} \implies \{i \geq |eventos|\}$

Por lo tanto programa es correcto

Muy bien la demostración de ciclo. Les falto completar con los pasos de demostración de programa completo. Tienen que usar el resultado de monotonía y probar que {P} i:=0; res := recursos {P_c}. Deberían comentarlo al principio y al final.

4. Anexo

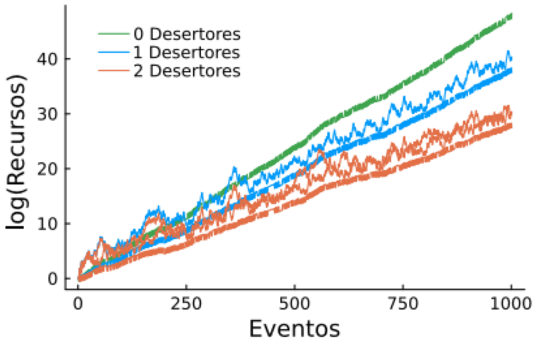

Figura 2: Recursos resultantes a lo largo de los eventos para 100 participantes, en verde se ve el caso donde todos los participantes aportan al fondo comun, en azul y rojo los casos con 1 y 2 desertores .

En la Figura 2 se ven los resultados obtenidos en un trabajo, donde se obtiene que al aumentar la cantidad de desertores, a pesar de que estos obtienen mejores resultados que el grupo participante del fondo común, los recursos para desertores y contribuyentes son inferiores a largo plazo que si todos hubiesen contribuido al fondo común. En caso de implementar un algoritmo como el tratado en este trabajo se buscaría obtener resultados consistentes con estos.