UBA – Facultad de Ciencias Exactas y Naturales – Departamento de Computación

Algoritmos y Estructura de Datos I

Segundo cuatrimestre de 2013

10 de noviembre de 2013

TPI Cine v 1.0

1. Función cambiarSalaT

```
problema cambiarSalaT (ts:[Ticket], vieja: Sala, nueva: Sala) = result : [Ticket]  {
  requiere mismaCantidad : |ts| == |result| ;
  asegura noCambianLosDeSalaNoVieja : (∀i ← [0..|ts|), sala(ts[i])! = vieja) ts[i] == result[i] ;
  asegura losDeSalaViejaCambianSoloLaSala); (∀i ← [0..|ts|), sala(ts[i]) == vieja)
      pelicula(result[i]) == pelicula(ts[i])
      ∧ usado(ts[i]) == usado(result[i])
      ∧ sala(result[i]) == nueva ;
}
```

```
   Lista<Ticket>Ticket::cambiarSalaT(const Lista<Ticket> &ts , Sala vieja, Sala nueva) {
\\estado E1 ;
\\Vale vieja == pre(vieja) ∧ nueva == pre(nueva) ;
int  i=0;
Lista<Ticket> y;
 \\estado E2 ;
\\Vale Pc ;
while( i < ts.longitud() ) {
 \\estado W1 ;
\\Vale Bc ∧ I ;
if( ts.iesimo(i).sala_ == vieja) {
 \\estado I1 ;
\\Vale Bc ∧ Bif ∧ i == i@W1 ∧ |y| == i@W1 ∧ y == y@W1 ∧ vieja == vieja@W1 ∧ nueva == nueva@W1 ;
y.agregarAtras(Ticket(ts.iesimo(i).pelicula_, nueva, ts.iesimo(i).usado_));
 \\estado I2 ;
\\Vale i == i@I1 ∧ |y| == i@I1 + 1
∧ y == agregarAtras(y@I1, nuevoT(pelicula(iesimo(ts, i@I1)), nueva, usado(iesimo(ts, i@I1))))
∧ vieja == vieja@I1 ∧ nueva == nueva@I1 ;
}else {
 \\estado I3 ;
\\Vale Bc ∧ ¬Bif ∧ i == i@W1 ∧ |y| == i@W1 ∧ y == y@W1 ∧ vieja == vieja@W1 ∧ nueva == nueva@W1 ;
y.agregarAtras(ts.iesimo(i));
 \\estado I4 ;
\\Vale ¬Bif ∧ i == i@I3 ∧ |y| == i@I3 + 1 ∧ y == agregarAtras(y@I3, iesimo(ts, i@I3)) ∧ vieja == vieja@I3 ∧ nueva ==
nueva@I3 ;
}
 \\estado W2 ;
\\Vale (Bif ∧ (i == i@I1 ∧ |y| == i@I1 + 1
∧ y == agregarAtras(y@I1, nuevoT(pelicula(iesimo(ts, i@I1)), nueva, usado(iesimo(ts, i@I1))))
∧ vieja == vieja@I1 ∧ nueva == nueva@I1)) ∨ (¬Bif ∧ (i == i@I3 ∧ |y| == i@I3 + 1
∧ y == agregarAtras(y@I3, iesimo(ts, i@I3)) ∧ vieja == vieja@I3 ∧ nueva == nueva@I3)) ;
i++;
 \\estado W3 ;
\\Vale i == i@W2 + 1 ∧ |y| == |y@W2| ∧ y == y@W2 ∧ vieja == vieja@W2 ∧ nueva == nueva@W2 ;
}
 \\estado E3 ;
\\Vale Qc ;
return y;
 \\estado E4 ;
\\Vale result == y@E3 ∧ Qc ;
}
```

$P_c : i == 0 \land y = [\ ] \land |y| == i \land vieja == vieja@E_1 \land nueva == nueva@E_1$;

$B_c : i < |ts|$;

$\neg B_c : i \geq |ts|$;

$I : 0 \leq i \leq |ts| \land |y| = i \land y == [ticketAgregado(ts, j)|j \leftarrow [0..i)]$
$\land vieja == pre(vieja) \land nueva == pre(nueva)$;

$B_{if} : sala(iesimo(ts, i)) == vieja$;

$\neg B_{if} : sala(iesimo(ts, i)) \neq vieja$;

$Q_c : i == |ts| \land |y| == |ts| \land y == [ticketAgregado(ts, j)|j \leftarrow [0..ts)]$
$\land vieja == vieja@W_3 \land nueva == nueva@W_3$;

$v(i) == |ts| - i$;

$c = 0$;

`aux ticketAgregado (ts:[Ticket],i:Int) : Ticket =` $if(salaT(iesimo(ts, j) == vieja)$
$then(nuevoT(pelicula(iesimo(ts, j)), nueva, usado(iesimo(ts, j))))else(iesimo(ts, j))$;

2. Demostración del Ciclo

I) $P_c \rightarrow I$
II) $(I \land \neg B_c) \rightarrow Q_c$
III) El cuerpo preserva I
IV) $v(i@W_1) > v(i@W_3)$(v es estríctamente decreciente)
V) $(I \land v \leq c) \rightarrow \neg B_c$

i) $P_c \rightarrow I$:

$\backslash\backslash Vale\ P_c$
$\backslash\backslash Implica\ 0 \leq i \leq |ts|;\ (porque\ i == 0)$
$\backslash\backslash Implica\ |y| = i;\ (porque\ i == 0 \land |y| = 0)$
$\backslash\backslash Implica\ y == [ticketAgregado(ts, j)|j \leftarrow [0..i)];\quad (porque\ y == [\ ] \land [ticketAgregado(ts, j)|j \leftarrow [0.,0)] == [\ ])$

ii) $(I \land \neg B_c) \rightarrow Q_c$:

$\backslash\backslash Vale\ (I \land \neg B_c)$
$\backslash\backslash Implica\ i == |ts|;\ (porque\ (\neg B_c \Rightarrow i \geq |ts|) \land (I \Rightarrow i \leq |ts|))$
$\backslash\backslash Implica\ |y| == |ts|;\ (porque\ (I \Rightarrow |y| = i) \land (por\ el\ paso\ anterior,\ i == |ts|))$
$\backslash\backslash Implica\ y == [ticketAgregado(ts, j)|j \leftarrow [0..ts)];\ (porque:\quad (I \Rightarrow y == [ticketAgregado(ts, j)|j \leftarrow [0..i)])$
$\land (por\ un\ paso\ anterior)\ i == |ts|)$
iii) El cuerpo preserva I:

qvq' Vale I en el estado W_3
ie:

- $0 \leq i@W_3 \leq |ts|$:

  $\backslash\backslash$estado W_3;
  $\backslash\backslash$Vale $i_@W_3 == i@W_2 + 1$;
  $\backslash\backslash$Implica $i_@W_3 == i@W_1 + 1$; $(porque: en\ el\ estado\ W_2,$
  $(B_{if} \Rightarrow i@W_2 == i@I_1 \land i@I_1 == i@W_1) \land (\neg B_{if} \Rightarrow i@W_2 == i@I_3 \land i@I_3 == i@W_1) \Rightarrow i@W_2 == i@W_1)$
  $\backslash\backslash$Implica $0 \leq i@W_3 \leq |ts|$; $(porque\ en\ el\ estado\ I_1\ se\ cumple\ I \land B_c \Rightarrow (0 \leq i@W_1 \leq |ts|) \land (i@W_1 < |ts|)$
  $\Rightarrow 0 \leq i@W_1 < |ts| \Rightarrow 1 \leq i@W_1 + 1 < |ts| + 1 \Rightarrow 1 \leq i@W_1 + 1 \leq |ts| \Rightarrow 0 \leq i@W_1 + 1 \leq |ts|$;
  Luego cómo $(i@W_3 == i@W_1 + 1)\ \land (0 \leq i@W_1 + 1 \leq |ts|) \Rightarrow 0 \leq i@W_3 \leq |ts|)$.

- $|y@W_3| == i@W_3$:

$\backslash\backslash estado\ W_3$;
$\backslash\backslash Vale\ |y@W_3| == |y@W_2|$;
$\backslash\backslash Implica\ |y@W_3| == i@W_3$; *(porque por el estado W_2:*
$(B_{if} \Rightarrow |y@W_2| == i@I_1 + 1 \Rightarrow |y@W_2| == i@W_1 + 1\ pues\ i@I_1 == i@W_1) \wedge$
$(\neg B_{if} \Rightarrow |y@W_2| == i@I_3 + 1 \Rightarrow |y@W_2| == i@W_1 + 1\ pues\ i@I_3 == i@W_1) \Rightarrow$
$|y@W_3| == i@W_3$ *pues como vimos anteriormente $i@W_3 == i@W_1 + 1$)*

- $vieja@W_3 == pre(vieja)$:

$\backslash\backslash estado\ W_3$;
$\backslash\backslash Vale\ vieja@W_3 == vieja@W_2$;
$\backslash\backslash Implica\ vieja@W_3 == pre(vieja)$; *(porque por el estado W_2 :*
$(B_{if} \Rightarrow vieja@W_2 == vieja@I_1 \wedge vieja@I_1 == vieja@W_1 \wedge vieja@W_1 == pre(vieja) \Rightarrow vieja@W_3 == pre(vieja)) \wedge$
$(\neg B_{if} \Rightarrow vieja@W_2 == vieja@I_3 \wedge vieja@I_3 == vieja@W_1 \wedge vieja@W_1 == pre(vieja) \Rightarrow vieja@W_3 == pre(vieja)) \Rightarrow$
$vieja@W_3 == pre(vieja))$

- $nueva@W_3 == pre(nueva)$:

$\backslash\backslash estado\ W_3$;
$\backslash\backslash Vale\ nueva@W_3 == nueva@W_2$;
$\backslash\backslash Implica\ nueva@W_3 == pre(nueva)$; *(porque por el estado W_2:*
$(B_{if} \Rightarrow nueva@W_2 == nueva@I_1 \wedge nueva@I_1 == nueva@W_1 \wedge nueva@W_1 == pre(nueva) \Rightarrow nueva@W_3 == pre(nueva)) \wedge (\neg B_{if} \Rightarrow nueva@W_2 == nueva@I_3 \wedge nueva@I_3 == nueva@W_1 \wedge nueva@W_1 == pre(nueva) \Rightarrow nueva@W_3 == pre(nueva)) \Rightarrow nueva@W_3 == pre(nueva))$

- $y@W_3 == [ticketAgregado(ts, j)|j \leftarrow [0..i@W_3)]$:

$\backslash\backslash estado\ W_3$;
$\backslash\backslash Vale\ y@W_3 == y@W_2$;
$\backslash\backslash Implica\ y@W_3 == [ticketAgregado(ts, j)|j \leftarrow [0..i@W_3)]$; *(porque por el estado W_2:*
Vale $B_{if} \Rightarrow y@W_2 == agregarAtras(y@I_1, nuevoT(pelicula(iesimo(ts, j)), nueva, usado(iesimo(ts, j))))$.
Por I_1 *vale* : $y@I_1 == y@W_1$.
Además, como en W_1 vale I $\Rightarrow y@W_1 == [ticketAgregado(ts, j)|j \leftarrow [0..i@W_1)]$;
$\Rightarrow y@W_3 == y@W_2 ==$
$agregarAtras([ticketAgregado(ts, j)|j \leftarrow [0..i@W_1)], nuevoT(pelicula(iesimo(ts, I_1)), nueva, usado(iesimo(ts, I_1))))$,
además, cómo vale B_{if},
$\Rightarrow y@W_3 == [ticketAgregado(ts, j)|j \leftarrow [0..i@W_1]] \wedge i@W_3 == i@W(1) + 1$
$\Rightarrow y@W_3 == [ticketAgregado(ts, j)|j \leftarrow [0..i@W_3 - 1]]$
$\Rightarrow y@W_3 == [ticketAgregado(ts, j)|j \leftarrow [0..i@W_3)]$.
Vale $\neg B_{if} \Rightarrow y@W_2 == agregarAtras(y@I_3, nuevoT(pelicula(iesimo(ts, j)), nueva, usado(iesimo(ts, j))))$.
Por I_3 *vale* : $y@I_3 == y@W_1$.*Además, como en W_1 vale I*
$\Rightarrow y@W_1 == [ticketAgregado(ts, j)|j \leftarrow [0..i@W_1)]$;
$\Rightarrow y@W_3 == y@W_2 ==$
$agregarAtras([ticketAgregado(ts, j)|j \leftarrow [0..i@W_1)], nuevoT(pelicula(iesimo(ts, i@I_3)), nueva, usado(iesimo(ts, i@I_3))))$.
Además, como vale $\neg B_{if}$,
$\Rightarrow y@W_3 == [ticketAgregado(ts, j)|j \leftarrow [0..i@W_1]] \wedge i@W_3 == i@W(1) + 1$
$\Rightarrow y@W_3 == [ticketAgregado(ts, j)|j \leftarrow [0..i@W_3 - 1]]$
$\Rightarrow y@W_3 == [ticketAgregado(ts, j)|j \leftarrow [0..i@W_3)]$.
De dónde queda:
$\backslash\backslash estado\ W_3$:
$\Rightarrow (B_{if} \wedge y@W_3 == [ticketAgregado(ts, j)|j \leftarrow [0..i@W_3)]) \vee (\neg B_{if} \wedge y@W_3 == [ticketAgregado(ts, j)|j \leftarrow [0..i@W_3)])$
$\Rightarrow y@W_3 == [ticketAgregado(ts, j)|j \leftarrow [0..i@W_3)])$

Entonces, probamos que el cuerpo del ciclo preserva el invariante (pues, vale I en el estado W_3)

iv) v@W_1 >v@W_3 (v es estríctamente decreciente):

Cómo $qvq'\ (v(i@W_1) > v(i@W_3)) \wedge (v(i) = |ts| - i) \Rightarrow\ qvq'(|ts| - i@W_1) > (ts| - i@W_3)$
$\backslash\backslash estado\ W_3$
$\backslash\backslash Vale\ i@W_3 == i@W_2 + 1$;
$\backslash\backslash Implica\ i@W_3 == i@W_1 + 1$; $(porque\ (i@W_2 == i@I_2 \wedge i@I_2 == i@I_1 \wedge i@I_1 == i@W_1 \Rightarrow i@W_2 == i@W_1)$
$\vee (i@W_2 == i@I_4 \wedge i@I_4 == i@I_3 \wedge i@I_3 == i@W_1 \Rightarrow i@W_2 == i@W_1))$
$\backslash\backslash Implica\ i@W_3 > i@W_1\ (porque\ i@W_1 + 1 > i@W_1)$
$\backslash\backslash Implica\ (|ts| - i@W_1) > (ts| - i@W_3)$ Qué es lo que queríamos ver; $(porque\ -(i@W_1 + 1) < -(i@W_1) \Rightarrow$

$|ts| - (i@W_1 + 1) < |ts| - i@W_1)$
Luego v es estrictamente decreciente.
$v)(I \wedge v \leq c) \rightarrow \neg B_c$:
$\backslash\backslash Vale\ \ (I \wedge v \leq c);$
$\backslash\backslash Implica\ \ \neg B_c;\ \ (porque\ \ (v \leq c) \Rightarrow (|ts| - i) \leq 0 \Rightarrow (|ts| \leq i) == \neg B_c)$

Luego, como valen (i), (ii), (iii), (iv), (v), *por el Teorema del invariante el ciclo es correcto.*

3. El algorítmo es correcto:

Sólo falta ahora ver que el estado $E_4 \Rightarrow$ los *asegura* de la especificación, es decir, quiero ver que:
$(result == y@E_3 \wedge Q_c) \Rightarrow\ \ (noCambianLosDeSalaNoVieja : (\forall i \leftarrow [0..|ts|), sala(ts[i]) \neq vieja)\ \ ts[i] == result[i])$
$\wedge\ (losDeSalaViejaCambianSoloLaSala : (\forall i \leftarrow [0..|ts|), sala(ts[i]) == vieja)\ \ pelicula(result[i]) == pelicula(ts[i])$
$\wedge\ usado(ts[i]) == usado(result[i])\ \ \wedge sala(result[i]) == nueva).$

$\backslash\backslash$*estado* E_4;
$\backslash\backslash Vale\ \ (result == y@E_3 \wedge Q_c);$
$\backslash\backslash Implica\ \ ((\forall j \leftarrow [0..|ts|), B_{if})iesimo(result, j) == nuevoT(pelicula(iesimo(ts, i)), nueva, usado(iesimo(ts, i))))$
$\wedge\ (\forall j \leftarrow [0..|ts|), \neg B_{if})iesimo(result, j) == iesimo(result, j));$ *(porque:*
$result == y@E_3 \Rightarrow [ticketAgregado(ts, j)|j \leftarrow [0..|ts|)]$
$\Rightarrow (\forall j \leftarrow [0..|ts|))(B_{if} \wedge iesimo(result, j) == nuevoT(pelicula(iesimo(ts, j)), nueva, usado(iesimo(ts, j))))$
$\vee\ (\neg B_{if} \wedge iesimo(result, j) == iesimo(result, j))$
$\Rightarrow ((\forall j \leftarrow [0..|ts|), B_{if})iesimo(result, j) == nuevoT(pelicula(iesimo(ts, j)), nueva, usado(iesimo(ts, j))))$
$\wedge\ (\forall j \leftarrow [0..|ts|), \neg B_{if})iesimo(result, j) == iesimo(result, j))$*.*
$\backslash\backslash Implica\ \ ((\forall j \leftarrow [0..|ts|), sala(iesimo(ts, j)) == vieja)pelicula(iesimo(result, j)) == pelicula(iesimo(ts, j))$
$\wedge\ sala(iesimo(result, j)) == nueva \wedge usado(iesimo(result, j)) == usado(iesimo(ts, j)))$
$\wedge((\forall j \leftarrow [0..|ts|), sala(iesimo(ts, j)) \neq vieja)iesimo(result, j) == iesimo(ts, j));$ *(porque, por definición de nuevoT,*
$iesimo(result, j) == nuevoT(pelicula(iesimo(ts, j)), nueva, usado(iesimo(ts, j)))$
$\Rightarrow pelicula(iesimo(result, j)) == pelicula(iesimo(ts, j)) \wedge sala(iesimo(result, j)) == nueva \wedge usado(iesimo(result, j)) == usado(iesimo(ts, j))$*.*
Además $B_{if} == (sala(iesimo(ts, i)) == vieja)\ \ y\ \ \neg B_{if} == (sala(iesimo(ts, i)) \neq vieja).$
$(\forall j \leftarrow [0..|ts|), B_{if})iesimo(result, j) == nuevoT(pelicula(iesimo(ts, i)), nueva, usado(iesimo(ts, i))))$
$\wedge\ (\forall j \leftarrow [0..|ts|), \neg B_{if})iesimo(result, j) == iesimo(result, j))$
$\Rightarrow ((\forall j \leftarrow [0..|ts|), sala(iesimo(ts, j)) == vieja)pelicula(iesimo(result, j)) == pelicula(iesimo(ts, j))$
$\wedge\ sala(iesimo(result, j)) == nueva \wedge usado(iesimo(result, j)) == usado(iesimo(ts, j)))$
$\wedge\ ((\forall j \leftarrow [0..|ts|), sala(iesimo(ts, j)) \neq vieja)iesimo(result, j) == iesimo(ts, j)))$*.*

$\backslash\backslash Implica\ \ (noCambianLosDeSalaNoVieja : (\forall i \leftarrow [0..|ts|), sala(ts[i]) \neq vieja)\ \ ts[i] == result[i])$
$\wedge\ (losDeSalaViejaCambianSoloLaSala : (\forall i \leftarrow [0..|ts|), sala(ts[i]) == vieja)\ \ pelicula(result[i]) == pelicula(ts[i])$
$\wedge\ usado(ts[i]) == usado(result[i])\ \ \wedge sala(result[i]) == nueva).$ *(porque, por especificación de iesimo,*
iesimo(l,i)==l[i]).