Algoritmos y Estructuras de Datos II

Primer Cuatrimestre de 2015

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Trabajo Prï¿½ctico 1

Especificaciï¿½n

Integrante	LU	Correo electrónico
BENITEZ, Nelson	945/13	nelson.benitez92@gmail.com
ROIZMAN, Violeta	273/11	violeroizman@gmail.com
Vï¿½ZQUEZ, Jï¿½sica	318/13	jesis_93@hotmail.com
ZAVALLA, Agustï¿½n	670/13	nkm747@gmail.com

Reservado para la cátedra

Instancia	Docente	Nota
Primera entrega		
Segunda entrega		

Índice

1. TAD AS

TAD AS

géneros as

igualdad observacional

$(\forall dc, dc' : \text{dcnet})\ \ (dc =_{\text{obs}} dc' \iff ())$

usa CAMPUS

exporta

observadores básicos

campus : as $\longrightarrow$ campus

seguridad : as $\longrightarrow$ conj(seguridad)

hayEst? : as a × pos p $\longrightarrow$ bool $\{posValida(campus(a), p)\}$

hayHippie? : as a × pos p $\longrightarrow$ bool $\{posValida(campus(a), p)\}$

#capturas : as a × seg s $\longrightarrow$ nat $\{s \in seguridad(a)\}$

#sanciones : as a × seg s $\longrightarrow$ nat $\{s \in seguridad(a)\}$

generadores

nueva : campus × conj(seguridad) $\longrightarrow$ as
{(∀ *segs*:e) posValida(c,pos(e)) ∧ (∀ *segs*:s,s1) id(s)!=id(s1) ⇒ pos(s)!=pos(s1)}

moverEst : as a × pos pe × pos pd $\longrightarrow$ as
$\left\{\begin{array}{l} posValida(campus(a), pe) \quad \wedge_{\text{L}} \quad hayEst?(a, pe) \quad \wedge \quad adyacente(campus(a), pe, pd) \quad \wedge \\ posValidaPersona(as, pd) \end{array}\right\}$

nuevoHippie : as a × pos p $\longrightarrow$ as $\{posIngreso(campus(a), p) \wedge posValidaPersona(a, p)\}$

nuevoEst : as a × pos p $\longrightarrow$ as $\{posIngreso(campus(a), p) \wedge posValidaPersona(a, p)\}$

sacarEst : as a × pos p $\longrightarrow$ as $\{posValida(campus(a), p) \wedge_{\text{L}} hayEst?(a, p) \wedge posIngreso(a, p)\}$

otras operaciones

haySeg? : as a × pos p $\longrightarrow$ bool

posValidaPersona : as a × pos p $\longrightarrow$ bool

posIngreso : as a × pos p $\longrightarrow$ bool

moverTodos : as a × conj(seguridad) $segs$ $\longrightarrow$ conj(seguridad)

moverSeg : as a × seguridad seg × pos $posSig$ $\longrightarrow$ seguridad

proximasPosiciones : as a × conj(pos) $minPos$ × pos $posAct$ $\longrightarrow$ conj(pos)
$\{\neg(emptyset?(minPos)) \wedge_{\text{L}} posValida(campus(a), posAct) \wedge posicionesValidas(campus(a), minPos\}$

hippiesMasCerca : as a × seguridad seg $\longrightarrow$ conj(pos) $\{seg \in seguridad(a) \wedge hayHippies(a)\}$

encerrado : as a × pos p $\longrightarrow$ bool {hayEst?(p)}

#hippies : as a $\longrightarrow$ nat

#estudiantes : as a $\longrightarrow$ nat

#masVigilante : as a $\longrightarrow$ nat

contarHippies : as a × conj(pos) $poss$ $\longrightarrow$ nat

contarEstudiantes : as a × conj(pos) $poss$ $\longrightarrow$ nat

#masCapturas : as a × conj(seg) $segs$ $\longrightarrow$ conj(seg) {(∀ *segs*:s) s ∈ *seguridad*(*a*)}

#maxCapturas : as a × conj(seg) $segs$ $\longrightarrow$ nat {(∀ *segs*:s) ∈ *seguridad*(*a*)} captura? : as a × pos p $\longrightarrow$ bool

axiomas

campus(nueva($c, segs$)) $\equiv$ c

campus(moverEst(a, p_1, p_2)) $\equiv$ $campus(a)$

campus(nuevoEst(a, p_1)) $\equiv$ $campus(a)$

campus(nuevoHippie(a, p_1)) $\equiv$ $campus(a)$

campus(sacarEst(a, p_1)) $\equiv$ $campus(a)$

seguridad(nueva($c, segs$)) $\equiv$ $segs$

seguridad(moverEst(a, p_1, p_2)) $\equiv$ $moverTodos(a, seguridad(a))$

seguridad(nuevoEst(a, p_1)) $\equiv$ $moverTodos(a, seguridad(a))$

seguridad(nuevoHippie(a, p_1)) $\equiv$ $seguridad(a)$

seguridad(sacarEst(a, p_1)) $\equiv$ $seguridad(a)$

hayEst?(nueva($c, segs$),p) $\equiv$ $False$

hayEst?(nuevoEst(a, p_1),p) $\equiv$ **if** $p_1 = p$ **then** $True$ **else** $hayEst?(a, p)$ **fi**

hayEst?(moverEst(a, p_1, p_2),p) $\equiv$ **if** $p_1 = p$ **then**
  $False$
else
  if $p_2 = p$ **then** $True$ **else** $hayEst?(a, p)$ **fi**
fi

hayEst?(nuevoHippie(a, p_1),p) $\equiv$ $hayEst?(a, p)$

hayEst?(sacarEst(a, p_1),p) $\equiv$ **if** $p_1 = p$ **then** $False$ **else** $hayEst?(a, p)$ **fi**

hayHippie?(nueva($c, segs$),p) $\equiv$ $False$

hayHippie?((nuevoHippie(a, p_1),p) $\equiv$ **if** $p_1 = p$ **then** $True$ **else** $hayHippie?(a, p)$ **fi**

hayHippie?(nuevoEst(a, p_1),p) $\equiv$ $hayHippie?(a, p)$

hayHippie?(sacarEst(a, p_1),p) $\equiv$ $hayHippie?(a, p)$

#capturas(nueva($a, segs$),s) $\equiv$ 0

#capturas(moverEst(a, p_1, p_2),s) $\equiv$ $\#capturas(a, s)$

#capturas(nuevoHippie(a, p_1),s) $\equiv$ **if** $(adyacente(a, p_1, posSeg(a, s)) \land encerrado(a, p_1))$ **then**
  $1 + \#capturas(a, s)$
else
  $\#capturas(a, s)$
fi

#capturas(nuevoEst(a, p_1),s) $\equiv$ $\#capturas(a, s)$

#capturas(sacarEst(a, p_1),s) $\equiv$ $\#capturas(a, s)$

#capturas(a,moverSeg(a, s, p_1)) $\equiv$ $\beta(posValida(campus(a), < \pi_1(p_1) + 1, \pi_2(p_1) >) \land_L (hayHippie?(a, < \pi_1(p_1) + 1, \pi_2(p_1) >) \land_L encerrado(a, < \pi_1(p_1) + 1, \pi_2(p_1) >))) +$
$\beta(posValida(campus(a), < \pi_1(p_1) - 1, \pi_2(p_1) >) \land_L (hayHippie?(a, < \pi_1(p_1) - 1, \pi_2(p_1) >) \land_L encerrado(a, < \pi_1(p_1) - 1, \pi_2(p_1) >))) +$
$\beta(posValida(campus(a), < \pi_1(p_1), \pi_2(p_1) + 1 >) \land_L (hayHippie?(a, < \pi_1(p_1), \pi_2(p_1) + 1 >) \land_L encerrado(a, < \pi_1(p_1), \pi_2(p_1) + 1 >))) +$
$\beta(posValida(campus(a), < \pi_1(p_1), \pi_2(p_1) - 1 >) \land_L (hayHippie?(a, < \pi_1(p_1), \pi_2(p_1) - 1 >) \land_L encerrado(a, < \pi_1(p_1), \pi_2(p_1) - 1 >))) + \#capturas(a, s)$

```
#capturas(moverEst($a, p_1, p_2$),$s$)    ≡ if ($PosValida(campus(a), < \pi_1(posSeg)+1, \pi_2(posSeg) >)$) then
                                                if ($hayHippie(a, < \pi_1(posSeg) + 1, \pi_2(posSeg) >)$) then
                                                    if ($captura?(a, < \pi_1(posSeg) + 1, \pi_2(posSeg) >)$) then
                                                        1
                                                    else
                                                        0
                                                    fi
                                                else
                                                    0
                                                fi
                                            else
                                                0
                                            fi

                                            +

                                            if ($PosValida(campus(a), < \pi_1(posSeg)-1, \pi_2(posSeg) >)$) then
                                                if ($hayHippie(a, < \pi_1(posSeg) - 1, \pi_2(posSeg) >)$) then
                                                    if ($captura?(a, < \pi_1(posSeg) - 1, \pi_2(posSeg) >)$) then
                                                        1
                                                    else
                                                        0
                                                    fi
                                                else
                                                    0
                                                fi
                                            else
                                                0
                                            fi
                                            +

                                            if ($PosValida(campus(a), < \pi_1(posSeg), \pi_2(posSeg)+1 >)$) then
                                                if ($hayHippie(a, < \pi_1(posSeg), \pi_2(posSeg) + 1 >)$) then
                                                    if ($captura?(a, < \pi_1(posSeg), \pi_2(posSeg) + 1 >)$) then
                                                        1
                                                    else
                                                        0
                                                    fi
                                                else
                                                    0
                                                fi
                                            else
                                                0
                                            fi
                                            +

                                            if ($PosValida(campus(a), < \pi_1(posSeg), \pi_2(posSeg)-1 >)$) then
                                                if ($hayHippie(a, < \pi_1(posSeg), \pi_2(posSeg) - 1 >)$) then
                                                    if ($captura?(a, < \pi_1(posSeg), \pi_2(posSeg) - 1 >)$) then
                                                        1
                                                    else
                                                        0
```

```
#sanciones(nueva($a, segs$),$s$)          ≡ $0$
#sanciones(moverEst($a, p_1, p_2$),$s$)   ≡ $\#sanciones(a, s)$
#sanciones(nuevoHippie($a, p_1$),$s$)     ≡ if $(cercanos?(a, p_1, posSeg(a, s)) \wedge_L (hayEst?(casilleroEnComun(a, p_1, posSeg(a, s))) \wedge encerrado(casilleroEnComun(a, p_1, posSeg(a, s)))))$
                                            then
                                                $1 + \#sanciones(a, s)$
                                            else
                                                $\#sanciones(a, s)$
                                            fi
#sanciones(nuevoEst($a, p_1$),$s$)        ≡ $\#sanciones(a, s)$
#sanciones(sacarEst($a, p_1$),$s$)        ≡ $\#sanciones(a, s)$
#sanciones($a$,moverSeg($a, s, p_1$))     ≡ $\beta(posValida(campus(a), < \pi_1(p_1) + 1, \pi_2(p_1) >) \wedge_L (hayEst?(a, < \pi_1(p_1) + 1, \pi_2(p_1) >) \wedge_L encerrado(a, < \pi_1(p_1) + 1, \pi_2(p_1) >))) +$
                                            $\beta(posValida(campus(a), < \pi_1(p_1) - 1, \pi_2(p_1) >) \wedge_L (hayEst?(a, < \pi_1(p_1) - 1, \pi_2(p_1) >) \wedge_L encerrado(a, < \pi_1(p_1) - 1, \pi_2(p_1) >))) +$
                                            $\beta(posValida(campus(a), < \pi_1(p_1), \pi_2(p_1) + 1 >) \wedge_L (hayEst?(a, < \pi_1(p_1), \pi_2(p_1) + 1 >) \wedge_L encerrado(a, < \pi_1(p_1), \pi_2(p_1) + 1 >))) +$
                                            $\beta(posValida(campus(a), < \pi_1(p_1), \pi_2(p_1) - 1 >) \wedge_L (hayEst?(a, < \pi_1(p_1), \pi_2(p_1) - 1 >) \wedge_L encerrado(a, < \pi_1(p_1), \pi_2(p_1) - 1 >))) + \#sanciones(a, s)$
moverTodos(a,segs)                        ≡ if $(\emptyset?(segs))$ then
                                                $\emptyset$
                                            else
                                                if $(hayHippies?(a))$ then
                                                    $Ag(moverTodos(a, sinUno(segs)),$
                                                    $moverSeg(a, dameUno(segs),$
                                                    $dameUno(proxPosiciones$
                                                    $(hippiesMasCerca(a, dameUno(segs))))))$
                                                else
                                                    $moverIngreso(a, segs)$
                                                fi
                                            fi
moverIngreso(a,segs)                      ≡ if $\emptyset?(segs)$ then
                                                $\emptyset$
                                            else
                                                if $(alto(campus(a)) - 1) - \pi_2(dameUno(segs)) > \pi_2(dameUno(segs))$ then
                                                    $ag(moverIngreso(a, sinUno(segs)), mover(dameUno(segs), <$
                                                    $(\pi_1(dameUno(segs)), \pi_2(segs) - 1) >))$
                                                else
                                                    if $(alto(campus(a)) - 1) - \pi_2(dameUno(segs)) < \pi_2(dameUno(segs))$ then
                                                        $ag(moverIngreso(a, sinUno(segs)), mover(dameUno(seg$
                                                        $(\pi_1(dameUno(segs)), \pi_2(segs) + 1) >))$
                                                    else
                                                        $ag(moverIngreso(a, sinUno(segs)), mover(dameUno(seg$
                                                        $dameUno(\{< (\pi_1(dameUno(segs)), \pi_2(segs) - 1) >, < (\pi_1(dameUno(segs)), \pi_2(segs)+1) >\}))$
                                                    fi
                                                fi
                                            fi
```

```
moverSeg(a,seg,nPos)    ≡ if $(distMan(campus(a), \pi_2(seg), nPos) \geq 2$
                          $\vee \neg(posValida(campus(a), nPos)))$ then
                              $seg$
                          else
                              if $\#sanciones(a, seg) < 3$ then
                                  $< \pi_1(seg), nPos >$
                              else
                                  $seg$
                              fi
                          fi
proximasPosiciones(hscerca, posSeg)    ≡ if $\emptyset?(hscerca)$ then
                              $\emptyset$
                          else
                              if $\pi_1(dameUno(hscerca)) > \pi_1(posSeg)$ then
                                  if $\pi_2(dameUno(hscerca)) > \pi_2(posSeg)$ then
                                      $\{< \pi_1(posSeg) + 1, \pi_2(posSeg) >,$
                                      $< \pi_1(posSeg), \pi_2(posSeg) + 1 >\}$
                                      $\cup\, proxPosiciones(sinUno(minPos), posSeg)$
                                  else
                                      if $\pi_2(dameUno(hscerca)) < \pi_2(posSeg)$ then
                                          $\{< \pi_1(posSeg) + 1, \pi_2(posSeg) >,$
                                          $< \pi_1(posSeg), \pi_2(posSeg) - 1 >\}$
                                          $\cup\, proxPosiciones(sinUno(minPos), posSeg)$
                                      else
                                          $\{< \pi_1(posSeg) + 1, \pi_2(posSeg) >\}$
                                          $\cup\, proxPosiciones(sinUno(minPos), posSeg)$
                                      fi
                                  fi
                              else
                                  if $\pi_1(dameUno(hscerca)) < \pi_1(posSeg)$ then
                                      if $\pi_2(dameUno(hscerca)) > \pi_2(posSeg)$ then
                                          $\{< \pi_1(posSeg) - 1, \pi_2(posSeg) >,$
                                          $< \pi_1(posSeg), \pi_2(posSeg) + 1 >\}$
                                          $\cup\, proxPosiciones(sinUno(minPos), posSeg)$
                                      else
                                          if $\pi_2(dameUno(hscerca)) < \pi_2(posSeg)$
                                          then
                                              $\{< \pi_1(posSeg) - 1, \pi_2(posSeg) >,$
                                              $< \pi_1(posSeg), \pi_2(posSeg) - 1 >\}$
                                              $\cup\, proxPosiciones(sinUno(minPos), posSeg)$
                                          else
                                              $\{< \pi_1(posSeg) - 1, \pi_2(posSeg) >\}$
                                              $\cup\, proxPosiciones(sinUno(minPos), posSeg)$
                                          fi
                                      fi
                                  else
                                      if $\pi_2(dameUno(hscerca)) > \pi_2(posSeg)$ then
                                          $\{< \pi_1(posSeg), \pi_2(posSeg) + 1 >\}$
                                          $\cup\, proxPosiciones(sinUno(minPos), posSeg)$
                                      else
                                          $\{< \pi_1(posSeg), \pi_2(posSeg) - 1 >\}$
                                          $\cup\, proxPosiciones(sinUno(minPos), posSeg)$
                                      fi
                                  fi
                              fi
                          fi
hippiesMasCerca(a,seg)  ≡ $minDistsPos(campus(a), \pi_2(seg), posHippies(a))$
#hippies(a)             ≡ $contarHippies(a, conjPos(campus(a), 0, 0))$
#estudiantes(a)         ≡ $contarEstudiantes(a, conjPos(campus(a), 0, 0))$
```

```
contarHippies(a,poss)        ≡ if $\neg(\emptyset?(poss))$ then
                                   if $posValida(campus(a), dameUno(poss))$ then
                                       if $hayHippie(a, dameUno(poss))$ then
                                           $1 + contarHippies(a, sinUno(poss))$
                                       else
                                           $contarHippies(a, sinUno(poss))$
                                       fi
                                   else
                                       $contarHippies(a, sinUno(poss))$
                                   fi
                               else
                                   $0$
                               fi
contarEstudiantes(a,poss)    ≡ if $\neg(\emptyset?(poss))$ then
                                   if $posValida(campus(a), dameUno(poss))$ then
                                       if $hayEst?(a, dameUno(poss))$ then
                                           $1 + contarEstudiantes(a, sinUno(poss))$
                                       else
                                           $contarEstudiantes(a, sinUno(poss))$
                                       fi
                                   else
                                       $contarEstudiantes(a, sinUno(poss))$
                                   fi
                               else
                                   $0$
                               fi
masVigilante(a)              ≡ $dameUno(masCapturas(a, seguridad(a)))$
masCapturas(a,segs)          ≡ if $\neg(\emptyset?(segs))$ then
                                   if $\#capturas(a, dameUno(segs)) \geq maxCapturas(a, segs)$
                                   then
                                       $ag(masCapturas(a, sinUno(segs)), dameUno(segs))$
                                   else
                                       $masCapturas(a, sinUno(segs))$
                                   fi
                               else
                                   $\emptyset$
                               fi
maxCapturas(a,segs)          ≡ if $\emptyset?(segs)$ then
                                   $0$
                               else
                                   if $\#capturas(a, dameUno(segs)) \geq$
                                   $maxCapturas(a, sinUno(segs))$
                                   then
                                       $\#capturas(a, dameUno(segs))$
                                   else
                                       $maxCapturas(a, sinUno(segs))$
                                   fi
                               fi
```

captura?(a, p) $\equiv$ **if** $(posValida(campus(a), <\pi_1(p)+1, \pi_2(p)>)$ **then**
  $(hayObstaculo?(campus(a), <\pi_1(p)+1, \pi_2(p)>) \vee haySeg?(a, <\pi_1(p)+1, \pi_2(p)>))$
else
  $\neg(hayEst?(a, <\pi_1(p), \pi_2(p)>))$
fi
$\wedge$
if $(posValida(campus(a), <\pi_1(p)-1, \pi_2(p)>)$ **then**
  $(hayObstaculo?(campus(a), <\pi_1(p)-1, \pi_2(p)>) \vee haySeg?(a, <\pi_1(p)-1, \pi_2(p)>))$
else
  $\neg(hayEst?(a, <\pi_1(p), \pi_2(p)>))$
fi
$\wedge$
if $(posValida(campus(a), <\pi_1(p), \pi_2(p)+1>)$ **then**
  $(hayObstaculo?(campus(a), <\pi_1(p), \pi_2(p)+1>) \vee haySeg?(a, <\pi_1(p), \pi_2(p)+1>))$
else
  $True$
fi
$\wedge$
if $(posValida(campus(a), <\pi_1(p), \pi_2(p)-1>)$ **then**
  $(hayObstaculo?(campus(a), <\pi_1(p), \pi_2(p)-1>) \vee haySeg?(a, <\pi_1(p), \pi_2(p)-1>))$
else
  $True$
fi

Fin TAD

2. TAD campus

TAD campus

géneros campus

usa campus

exporta

observadores básicos

alto : campus ⟶ nat

ancho : campus ⟶ nat

obstaculos : campus ⟶ conj(pos)

generadores

nuevo : nat *ancho* × nat *alto* × conj(pos) *obst* ⟶ campus
$\{1 \leq ancho \wedge 1 \leq alto \wedge (\forall\, p{:}\text{pos})\ p \in obst \Rightarrow_{\text{L}} posValida(c, p)\}$

otras operaciones

adyacente : as *a* × pos *pe* × pos *pd* ⟶ bool $\{posValida(c, pe) \wedge posValida(c, pd)\}$

posValida : as *a* × pos *p* ⟶ bool

posIngreso : as *a* × pos *p* ⟶ bool

minDistsPos : campus *c* × pos *p* × conj(pos) *posiciones* ⟶ conj(pos) $\{\neg(\emptyset?(posiciones))\}$

minDist : campus *c* × pos *p* × conj(posiciones) *posiciones* ⟶ nat $\{\neg(\emptyset?(posiciones))\}$

distMan : campus c × pos $p1$ × pos $p2$ $\longrightarrow$ nat

restaAbs : nat × nat $\longrightarrow$ nat

conjPos : campus × nat × nat $\longrightarrow$ conj(pos)

axiomas $\forall$ *alto*:nat, $\forall$ *ancho*:nat, $\forall$ *obst*:conj (pos)
$\forall$ p_1:pos $\forall$ p_2:pos

```
alto(nuevo(ancho,alto,obst))                 ≡ $alto$
ancho(nuevo(ancho,alto,obst))                ≡ $ancho$
obstaculos(nuevo(ancho,alto,obst))           ≡ $obst$
posValida(nuevo(ancho,alto,obst),$p_1$)        ≡ $\pi_1(p_1) < ancho \wedge \pi_2(p_1) < alto$
adyacente(nuevo(ancho,alto,obst),$p_1$,$p_2$)    ≡ $(\pi_1(p_1) = \pi_1(p_2) - 1 \vee \pi_1(p_1) = \pi_1(p_2) + 1) \wedge$
                                               $(\pi_2(p_1) = \pi_2(p_2) - 1 \vee \pi_2(p_1) = \pi_2(p_2) + 1)$
posValida(nuevo(ancho,alto,obst),$p_1$)        ≡ $\pi_2(p_1) = alto - 1 \vee \pi_2(p_1) = 0$
minDistsPos(c,p,posiciones)                  ≡ if $\emptyset?(sinUno(posiciones))$ then
                                                 $dameUno(posiciones)$
                                               else
                                                 if $distMan(c, p, dameUno(posiciones)) \leq$
                                                 $minDist(c, p, posiciones)$ then
                                                   $Ag(minDistsPos(c, sinUno(posiciones)),$
                                                   $dameUno(posiciones))$
                                                 else
                                                   $minDistsPos(c, seg, sinUno(posiciones))$
                                                 fi
                                               fi
minDist(c,p,posiciones)                      ≡ if $\emptyset?(sinUno(posiciones))$ then
                                                 $distMan(c, p, dameUno(posiciones))$
                                               else
                                                 if $distMan(c, p, dameUno(posiciones)) \leq$
                                                 $minDist(c, pos/p, sinUno(posiciones))$
                                                 then
                                                   $distMan(c, p, dameUno(posiciones))$
                                                 else
                                                   $minDist(c, p, sinUno(posiciones))$
                                                 fi
                                               fi
distMan(c,$p_1$,$p_2$)                           ≡ $restaAbs(\pi_2(p_1), \pi_2(p_2)) + restaAbs(\pi_1(p_1), \pi_1(p_2))$
restaAbs(n1,n2)                              ≡ if $n2 > n1$ then $n2 - n1$ else $n1 - n2$ fi
conjPos(c,x,y)                               ≡ if $x \geq ancho(c)$ then
                                                 $\emptyset$
                                               else
                                                 if $y \geq alto(c)$ then
                                                   $conjPos(c, x + 1, 0)$
                                                 else
                                                   $ag(conjPos(c, x, y + 1), < x, y >)$
                                                 fi
                                               fi
```

Fin TAD