

Algoritmos y Estructuras de Datos II
Recuperatorio del Trabajo Práctico 1 (Especificación)
Grupo 6

Bayardo, Julián
julian@bayardo.com.ar
850/13

Cuneo, Christian
chriscuneo93@gmail.com
755/13

Gambaccini, Ezequiel
ezequiel.gambaccini@gmail.com
715/13

Lebrero Rial, Ignacio Manuel
ignacialebrero@gmail.com
751/13

2 de Octubre del 2014

1. Aclaraciones

Asumimos que las estaciones, junto con las sendas y las restricciones entre ellas no son mutables: una instancia del TAD Ciudad comienza con un mapa definido y no puede ser cambiado. Suponemos también que las estaciones pueden tener sendas que las conecten a sí mismas, mas sólo una senda que la conecte a otra estación definida. Aparte, consideramos que pueden haber estaciones no conexas, y en cuyo caso no son consideradas bloqueantes (aunque los robots en ellas no podrían moverse hacia ningún lado).

Los RUR se suponen asignados al momento de agregar un robot, la única restricción al respecto es que el RUR efectivamente sea único. No puede existir un robot que no tenga ninguna característica.

Dos restricciones son equivalentes cuando su tabla de verdad es la misma para todo posible conjunto de características.

Cabe destacar que no modelamos el paso del tiempo, por lo que las inspecciones no son automáticas, sino que deben ser causadas deliberadamente por quien maneje el sistema.

2. Sobre el formato

Inicialmente escribimos la especificación utilizando UTF-8 en lugar de LaTeX, pensando que podríamos fácilmente incluir el archivo en nuestro informe, mas esto resultó ser problemático por varias cuestiones. Finalmente, optamos por incluirlo por separado como un archivo en otro pdf. Debido a problemas con esta solución, tuvimos que ajustar el texto para que no pase de 90 columnas; adoptamos como convención que si las precondiciones de una función superan tal tamaño, la guarda se pasa a otra línea y se indenta.

TAD característica ES string

TAD Restriccion

generos restriccion

exporta AND, OR, NOT, VAR, cumple, FALSE, TRUE

igualdad observacional

$(\forall r_1, r_2 : \text{restriccion}) (r_1 =_{\text{obs}} r_2 \iff$
 $(\forall c : \text{conj(caracteristica)}) (\text{cumple}(c, r_1) =_{\text{obs}} \text{cumple}(c, r_2)))$

observadores basicos

$\text{cumple} : \text{conj(caracteristica)} \times \text{restriccion} \rightarrow \text{bool}$

generadores

AND: $\text{restriccion} \times \text{restriccion} \rightarrow \text{restriccion}$

OR: $\text{restriccion} \times \text{restriccion} \rightarrow \text{restriccion}$

NOT: $\text{restriccion} \rightarrow \text{restriccion}$

VAR: $\text{caracteristica} \rightarrow \text{restriccion}$

otras operaciones

TRUE: $\rightarrow \text{restriccion}$

FALSE: $\rightarrow \text{restriccion}$

axiomas $(\forall cs : \text{conj(caracteristica)}) (\forall c_1, c_2 : \text{restriccion}) (\forall c : \text{caracteristica})$

$\text{cumple}(cs, \text{VAR}(c)) \equiv c \in cs$

$\text{cumple}(cs, \text{NOT}(c_1)) \equiv \neg(\text{cumple}(cs, c_1))$

$\text{cumple}(cs, \text{AND}(c_1, c_2)) \equiv \text{cumple}(cs, c_1) \wedge \text{cumple}(cs, c_2)$

$\text{cumple}(cs, \text{OR}(c_1, c_2)) \equiv \text{cumple}(cs, c_1) \vee \text{cumple}(cs, c_2)$

$\text{TRUE} \equiv \text{OR}(\text{VAR}(\text{"dummy"}), \text{NOT}(\text{VAR}(\text{"dummy"})))$

$\text{FALSE} \equiv \text{AND}(\text{VAR}(\text{"dummy"}), \text{NOT}(\text{VAR}(\text{"dummy"})))$

Fin TAD

TAD estacion ES string

TAD conexion ES (estacion, restriccion)

TAD conexiones ES conj(conexion)

TAD Mapa

generos mapa

exporta estaciones, conexiones, nuevo, crearEst, conectar,
esBloqueante, conectadas, camino

igualdad observacional

$(\forall m_1, m_2 : \text{mapa}) (m_1 =_{\text{obs}} m_2 \iff$
 $\text{estaciones}(m_1) =_{\text{obs}} \text{estaciones}(m_2) \wedge$
 $(\forall e \in \text{estaciones}(m_1)) (\text{conexiones}(m_1, e) =_{\text{obs}} \text{conexiones}(m_2, e)))$

observadores basicos

$\text{estaciones} : \text{mapa} \rightarrow \text{conj(estacion)}$

$\text{conexiones} : \text{mapa } m \times \text{estacion } e \rightarrow \text{conj(conexion)} \{e \in \text{estaciones}(m)\}$

generadores

nuevo : $\rightarrow \text{mapa}$

crearEst : $\text{mapa } m \times \text{estacion } a \rightarrow \text{mapa } \{\neg(a \in \text{estaciones}(m))\}$

conectar : $\text{mapa } m \times \text{estacion } a \times \text{estacion } b \times \text{restriccion} \rightarrow \text{mapa}$
 $\{a, b \in \text{estaciones}(m) \wedge \neg \text{conectadas}(m, a, b)\}$

otras operaciones

$\text{esBloqueante} : \text{mapa } m \times \text{conj(caracteristica)} \times \text{estacion } e \rightarrow \text{bool } \{e \in \text{estaciones}(m)\}$

$\text{esBloqueante}' : \text{conj(conexion)} \times \text{conj(caracteristica)} \rightarrow \text{bool}$

$\text{conectadas} : \text{mapa } m \times \text{estacion } a \times \text{estacion } b \rightarrow \text{bool } \{a, b \in \text{estaciones}(m)\}$

$\text{conectadas}' : \text{estacion } \times \text{conj(conexion)} \rightarrow \text{bool}$

$\text{camino} : \text{mapa } m \times \text{estacion } a \times \text{estacion } b \rightarrow \text{restriccion}$

$\{a, b \in \text{estaciones}(m) \wedge \text{conectadas}(m, a, b)\}$

$\text{camino}' : \text{estacion } e \times \text{conj(conexion)} \text{ cc} \rightarrow \text{restriccion}$

$\{(\exists x \in \text{cc}) (\pi_1(x) =_{\text{obs}} e)\}$

axiomas $(\forall m : \text{mapa}) (\forall e, a, b : \text{estacion}) (\forall r : \text{restriccion}) (\forall cs : \text{conj(caracteristica)})$
 $(\forall cc : \text{conj(conexion)})$

```

conexiones(crearEst(m, e), k)  $\equiv \emptyset$ 
conexiones(conectar(m, a, b, r), e)  $\equiv$ 
  (if e =obs a then {b, r}
   else if e =obs b then {a, r} else  $\emptyset$  fi
  fi)  $\cup$  conexiones(m, e)

estaciones(nuevo)  $\equiv \emptyset$ 
estaciones(crearEst(m, e))  $\equiv \{ e \} \cup$  estaciones(m)
estaciones(conectar(m, a, b))  $\equiv$  estaciones(m)

esBloqueante(m, cs, e)  $\equiv$ 
  if  $\emptyset?$ (conexiones(m, e)) then False
  else esBloqueante'(conexiones(m, e), cs) fi

esBloqueante'(cc, cs)  $\equiv$ 
  if  $\emptyset?$ (cc) then True
  else  $\neg$ (cumple(r,  $\pi_2$ (dameUno(cc))))  $\wedge$  esBloqueante'(sinUno(cc), cs) fi

conectadas(m, a, b)  $\equiv$  conectadas'(b, conexiones(m, a))

conectadas'(a, cc)  $\equiv$ 
  if  $\emptyset?$ (cc) then False
  else ( $\pi_1$ (dameUno(cc)) =obs a)  $\vee$  conectadas'(a, sinUno(cc)) fi

camino(m, a, b)  $\equiv$  camino'(b, conexiones(m, a))

camino'(a, cc)  $\equiv$ 
  if  $\pi_1$ (dameUno(cc)) =obs a then  $\pi_2$ (dameUno(cc))
  else caminos'(a, sinUno(cc)) fi

```

Fin TAD

TAD rur ES nat

TAD robot ES {rur, conj(caracteristica)}

TAD Ciudad

generos ciudad

exporta mapeo, robots, posicion, #infracciones, #inspecciones, nueva, agregar,
mover, inspeccion, caracteristicaMasInfractora

igualdad observacional

($\forall c_1, c_2 : \text{ciudad}$) ($c_1 =_{\text{obs}} c_2 \iff$
 mapeo(c_1) =obs mapeo(c_2) \wedge robots(c_1) =obs robots(c_2) \wedge
 (($\forall e \in \text{estaciones}(\text{mapeo}(c))$) (#inspecciones(c_1 , e) =obs #inspecciones(c_2 , e))) \wedge
 (($\forall r \in \text{robots}(c_1)$) (
 #infracciones(c_1 , $\pi_1(r)$) =obs #infracciones(c_2 , $\pi_1(r)$) \wedge
 posicion(c_1 , $\pi_1(r)$) =obs posicion(c_2 , $\pi_1(r)$)))

observadores basicos

mapeo: ciudad \rightarrow mapa

robots: ciudad \rightarrow conj(robot)

posicion: ciudad $c \times \text{rur } i \rightarrow \text{estacion } \{(\exists r \in \text{robots}(c)) \pi_1(r) =_{\text{obs}} i\}$

#infracciones: ciudad $c \times \text{rur } i \rightarrow \text{nat } \{(\exists r \in \text{robots}(c)) \pi_1(r) =_{\text{obs}} i\}$

#inspecciones: ciudad $c \times \text{estacion } e \rightarrow \text{nat } \{e \in \text{estaciones}(\text{mapeo}(c))\}$

generadores

nueva: mapa \rightarrow ciudad

agregar: ciudad $c \times \text{rur } i \times \text{conj}(\text{caracteristica}) \text{ cs} \times \text{estacion } e \rightarrow \text{ciudad}$
 $\{-\emptyset?(cs) \wedge \neg((\exists r \in \text{robots}(c)) \pi_1(r) =_{\text{obs}} i) \wedge e \in \text{estaciones}(\text{mapeo}(c)) \wedge$
 $\neg \text{esBloqueante}(\text{mapeo}(c), cs, e)\}$

mover: ciudad $c \times \text{rur } i \times \text{estacion } e \rightarrow \text{ciudad}$

```

{e ∈ estaciones(mapeo(c)) ∧ ((∃r ∈ robots(c)) π1(r) = obs i) ∧1
conectadas(mapeo(c), posicion(c, i), e)}

inspeccion: ciudad c × estacion e → ciudad {e ∈ estaciones(mapeo(c))}
otras operaciones

masInfracciones: ciudad c × estacion e → conj(robot) {e ∈ estaciones(mapeo(c))}
robotsEnEstacion: ciudad c × conj(robot) cr × estacion → conj(robot) {cr ⊆ robots(c)}
maxInfractor: ciudad c × conj(robot) cr × robot r → robot
  {r ∈ robots(c) ∧ cr ⊆ robots(c) ∧ ¬φ?(cr)}

caracteristicaMasInfractora: ciudad c → caracteristica
  {(∃r ∈ robots(c)) (#infracciones(π1(r)) > 0)}
infractoras: ciudad c × conj(robot) cr → multiconj(caracteristica) {cr ⊆ robots(c)}
aMulti: conj(caracteristica) → multiconj(caracteristica)
maximizar: multiconj(caracteristica) mc → caracteristica {¬φ?(mc)}
agregarN: nat × conj(caracteristica) → multiconj(caracteristica)
axiomas (∀m : mapa) (∀c : ciudad) (∀i : rur) (∀cc : conj(caracteristicas))
  (∀e, e' : estacion) (∀cr : conj(robot)) (∀mc : multiconj(caracteristica))
mapeo(nueva(m)) ≡ m
mapeo(agregar(c, i, cc, e)) ≡ mapeo(c)
mapeo(mover(c, i, e)) ≡ mapeo(c)
mapeo(inspeccion(c, e)) ≡ mapeo(c)

robots(nueva) ≡ φ
robots(agregar(c, i, cc, e)) ≡ Ag({i, cc}, robots(c))
robots(mover(c, i, e)) ≡ robots(c)
robots(inspeccion(c, e)) ≡ robots(c) - masInfracciones(c, e)

posicion(agregar(c, i, cc, e), r) ≡
  if i = obs r then e
  else posicion(c, r) fi
posicion(mover(c, i, e), r) ≡
  if i = obs r then e
  else posicion(c, r) fi
posicion(inspeccion(c, e), r) ≡ posicion(c, r)

#infracciones(agregar(c, i, cc, e), r) ≡
  if i = obs r then 0
  else #infracciones(c) fi

#infracciones(inspeccion(c, e), r) ≡ #infracciones(c, r)

#infracciones(mover(c, i, e), r) ≡
  if r = obs i then
    #infracciones(c, r) + β(¬cumple(camino(mapeo(c), posicion(c, r), e)))
  else #infracciones(c, r) fi

#inspecciones(nueva, e) ≡ 0
#inspecciones(agregar(c, i, cc, e'), e) ≡ #inspecciones(c)
#inspecciones(mover(c, i, e'), e) ≡ #inspecciones(c)
#inspecciones(inspeccion(c, e'), e) ≡ β(e' = obs e) + #inspecciones(c, e)

masInfracciones(c, e) ≡
  if ¬φ?(robotsEnEstacion(c, robots(c), e)) ∧1
    #infracciones(c, π1(maxInfractor(c, robotsEnEstacion(c, robots(c), e)))) > 0
  then Ag(maxInfractor(c, robotsEnEstacion(c, robots(c), e)), φ)
  else φ fi

maxInfractor(c, cr) ≡
  if φ?(sinUno(cr)) v1 #infracciones(c, π1(dameUno(cr))) >

```

```

                                #infracciones(c,  $\pi_1$ (maxInfractor(c, sinUno(cr))))
then dameUno(cr)
else maxInfractor(c, sinUno(cr)) fi

robotsEnEstacion(c, cr, e)  $\equiv$ 
  if  $\phi?$ (cr) then  $\phi$ 
  else
    if posicion(c,  $\pi_1$ (dameUno(cr))) =obs e then
      Ag(dameUno(cr), robotsEnEstacion(c, sinUno(cr), e))
    else robotsEnEstacion(c, sinUno(cr), e) fi
  fi

caracteristicaMasInfractora  $\equiv$  maximizar(infractoras(c, robots(c)))

infractoras(c, cr)  $\equiv$ 
  if  $\phi?$ (cr) then  $\phi$ 
  else
    infractoras(c, sinUno(cr)) u
    if #infracciones(c, dameUno(cr)) > 0 then
      agregarN(#infracciones(c, dameUno(cr)),  $\pi_2$ (dameUno(cr)))
    else  $\phi$  fi
  fi

aMulti(cs)  $\equiv$  if  $\phi?$ (cs) then  $\phi$  else Ag(dameUno(cs), aMulti(sinUno(cs))) fi

agregarN(i, cs)  $\equiv$  if i =obs 0 then  $\phi$  else aMulti(cs) u agregarN(i-1, cs) fi

maximizar(mc)  $\equiv$ 
  if  $\phi?$ (sinUno(mc)) v1 #(dameUno(mc), mc) > #(maximizar(sinUno(mc)), mc) then
    dameUno(mc)
  else maximizar(sinUno(mc)) fi
Fin TAD

```