Algoritmos y Estructuras de Datos II

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

Trabajo Práctico I

Grupo: 12

Integrante	LU	Correo electrónico
Demartino, Francisco	348/14	demartino.francisco@gmail.com
Paz, Maximiliano León	251/14	m4xileon@gmail.com
Mena, Manuel	313/14	manuelmena1993@gmail.com
Pondal, Iván	078/14	ivan.pondal@gmail.com

Reservado para la cátedra

Instancia	Docente	Nota
Primera entrega		
Segunda entrega		

TAD DCNet

géneros dcnet

igualdad observacional

$$(\forall d, d' : \text{dcnet}) \left(d =_{\text{obs}} d' \iff \left(\begin{array}{l} (topo(d) =_{\text{obs}} topo(d')) \land ((\forall p : pc)(p \in pcs(topo(d)) \land \\ p \in pcs(topo(d')) \Rightarrow_{\text{L}} (dcNetBuffer(d,p) =_{\text{obs}} \\ dcNetBuffer(d',p) \land paquetesMandados(d,p) =_{\text{obs}} \\ paquetesMandados(d',p)) \land ((\forall p : paquetes)((\exists c : \\ pc)(c \in pcs(topo(d') \land c \in pcs(topo(d')) \land_{\text{L}} (p \in \\ dcNetBuffer(d,c) \land p \in dcNetBuffer(d',c))) \Rightarrow_{\text{L}} \\ (recorridoPaquete(d,p) =_{\text{obs}} recorridoPaquete(d',p))) \end{array} \right) \right)$$

generadores

crearRed	: topo	$\longrightarrow$	dcnet
seg	: dcnet	$\longrightarrow$	dcnet
paquetePendiente	: dcnet dcn × pc $p1$ × pc $p2$ × paquete	$\longrightarrow$	dcnet

$\{(p_1 \in pcs(topo(dcn)) \land p_2 \in pcs(topo(dcn))) \land_{\text{L}} conectadas?(topo(dcn), p_1, p_2)\}$

observadores básicos

recorridoPaquete	: dcnet dcn × paquete p	$\longrightarrow$	secu((ip,interface)))
	$\{(\exists c : pc)(c \in pcs(topo(dcn)) \land_{\text{L}} (p \in dcNetBuffer(dcn, c))\}$		
dcNetBuffer	: dcnet dcn × pc p	$\longrightarrow$	conj(paquete) $\{p \in$ pcs(topo(dcn))$\}$
paquetesMandados	: dcnet dcn × pc p	$\longrightarrow$	nat $\{p \in$ pcs(topo(dcn))$\}$
topo	: dcnet	$\longrightarrow$	topologia

otras operaciones

paqueteEnTransito?	: dcnet × paquete	$\longrightarrow$	bool
perteneceBuffers?	: paquete × buffers	$\longrightarrow$	bool
maxPaquetesMandados	: dcnet	$\longrightarrow$	pc
auxMaxPaquetes	: dcnet × conj(pc)	$\longrightarrow$	pc)
pasoSeg	: topo × buffers × buffers	$\longrightarrow$	buffers
regresion	: topo × buffers × secu(buffers)	$\longrightarrow$	buffers
cronoPaquetes	: dcnet × diccionario(pc × conj(paquete))	$\longrightarrow$	secu(buffers)
auxDefinir	: buffers × pc × conj(paquete) × conj(paquete)	$\longrightarrow$	buffers
auxBorrar	: buffers × pc × conj(paquete) × conj(paquete)	$\longrightarrow$	buffers
envioYReciboPaquetes	: topo × buffers × conj(pc)	$\longrightarrow$	buffers
envio	: topo × buffers × buffer	$\longrightarrow$	buffers
nuevosPaquetes	: buffers × buffers	$\longrightarrow$	buffers
damePaquete	: buffer	$\longrightarrow$	paquete
pasarA	: topologia × pc × pc	$\longrightarrow$	pc

axiomas $\forall$ p, p': paquete,$\forall$ c, c': pc,$\forall$ dcn: dcnet,$\forall$ t: topologia

topo(crearRed(t))	$\equiv$	t
topo(seg(dcn))	$\equiv$	topo(dcn)

topo(paquetePendiente(dcn,c,c',p)) ≡ topo(dcn)

paquetesMandados(crearRed(t),c) ≡ 0

paquetesMandados(seg(dcn),c) ≡ paquetesMandados(dcn)

paquetesMandados(paquetePendiente(dcn,o,d,p),c) ≡ **if** $c = o$ **then**
  $paquetesMandados(dcn, c) + 1$
else
  $paquetesMandados(dcn, c)$
fi

dcNetBuffer(dcn,c) ≡ obtener(c,regresion(topo(dcn),vacio,cronoPaquetes(dcn,vacio)))

maxPaquetesMandados(dcn) ≡ auxMaxPaquetes(dcn,pcs(topo(dcn)))

auxMaxPaquetes(dcn,cs) ≡ **if** $\emptyset?(sinUno(cs))$ **then**
  $dameUno(cs)$
else
  if $paquetesMandados(dcn, dameUno(cs) < paquetesMandados(dcn, auxMaxPaquetes(dcn, sinUno(cs)))$ **then**
    $auxMaxPaquetes(dcn, sinUno(cs))$
  else
    $dameUno(cs)$
  fi
fi

paqueteEnTransito?(dcn,p) ≡ perteneceBuffers?(p,regresion(topo(dcn),vacio, cronoPaquetes(dcn,vacio)))

perteneceBuffers?(p,bs) ≡ **if** $\emptyset?(claves(bs))$ **then**
  $false$
else
  if $p \in obtener(dameUno(claves(bs)), bs)$ **then**
    $true$
  else
    $perteneceBuffers?(p, borrar(dameUno(claves(bs)), bs))$
  fi
fi

cronoPaquetes(crearRed(t),bs) ≡ <bs>

cronoPaquetes(seg(dcn),bs) ≡ bs • cronoPaquetes(dcn,∅)

cronoPaquetes(paquetePendiente(dcn,o,d,p),bs) ≡ $auxDefinir(bs, o, Ag(p, \emptyset), obtener(o, bs))$ $cronoPaquetes(dcn, bs)$

auxDefinir(bs,c,n,v) ≡ **if** $def?(c, bs)$ **then**
  $borrar(c, bs)\ \ definir(c, n \cup v, bs)$
else
  $definir(c, n)$
fi

auxBorrar(bs,c,b,p) ≡ **if** $\emptyset?(p - \{b\})$ **then**
  $borrar(c, n)$
else
  $borrar(c, bs)\ \ definir(c, p - \{b\}, bs)$
fi

regresion(t,bs,cbs) ≡ **if** $vacia?(fin(cbs))$ **then**
  $pasoSeg(bs, t, prim(cbs))$
else
  $regresion(t, pasoSeg(bs, t, prim(cbs)), fin(cbs))$
fi

pasoSeg(t,bs,nbs) ≡ envioYReciboPaquetes(t,bs,claves(bs)) nuevosPaquetes(bs,nbs)

envioYReciboPaquetes(t,bs,cp) $\equiv$ **if** $\emptyset?(cp)$ **then**
bs
else
$envioYReciboPaquetes(t, envio(t, bs, dameUno(cp)),$
$sinUno(cp))$
fi

pasarA(t,o,d) $\equiv$ $prim(caminoMin(t, o, d))$

envio(t,bs,b) $\equiv$ $auxDefinir(bs, pasarA(t, \Pi_1(b), dest(\Pi_2(b))),$
$Ag(damePaquete(b), \emptyset),$ $obtener$
$(pasarA(t, \Pi_1(b), dest(\Pi_2(b))), bs)$
$auxBorrar(bs, \Pi_1(b), damePaquete(b),$
$obtener(bs, \Pi_1(b))$

nuevosPaquetes(bs,nbs) $\equiv$ **if** $\emptyset?(claves(nbs))$ **then**
bs
else
$auxDefinir(bs, dameUno(claves(nbs), obtener$
$(dameUno(claves(nbs), nbs), obtener(dameUno$
$(claves(nbs), bs)))$
$nuevosPaquetes(bs, sinUno(nbs))$
fi

TAD buffers es diccionario(pc,conj(paquete))
TAD buffer es tupla(pc,conj(paquete))

Fin TAD

Este TAD modela cómo se conectan las computadoras. Las IP son únicas entre compus de la topología. Las compus tienen interfaces numeradas con los naturales de manera consecutiva (todas funcionan perfecto y todo eso, el DC las cuida y mantiene como corresponde).

TAD TOPOLOGÍA

géneros topologia

generadores

NuevaTopo	:		$\longrightarrow$ topologia	
Compu	:	topologia × nat ip × nat	$\longrightarrow$ topologia	$\{\neg(ip \in compus(t))\}$
Cable	:	topologia × nat ipA × nat ifA × nat ipB × nat ipB	$\longrightarrow$ topologia	

$$\left\{\begin{array}{l}\neg(ipA \in compus(t) \lor ipB \in compus(t)) \land_{\mathrm{L}} \\ (ifA \leq numInterfaces(t, ipA)) \land \\ (ifB \leq numInterfaces(t, ipB)) \land \\ \neg(ifA \in interfacesOcupadasDe(t, ipA)) \land \\ \neg(ifB \in interfacesOcupadasDe(t, ipB)) \land \\ \neg(ipA \in vecinasDe(t, ipB))\end{array}\right\}$$

observadores básicos

compus	:	topologia	$\longrightarrow$ conj(nat)	
cablesEn	:	topologia t × nat ip	$\longrightarrow$ conj(tupla(nat, nat))	$\{ip \in compus(t)\}$
numInterfaces	:	topologia t × nat ip	$\longrightarrow$ nat	$\{ip \in compus(t)\}$

otras operaciones

vecinasDe	:	topologia t × nat ip	$\longrightarrow$ conj(nat)	$\{ip \in compus(t)\}$
interfacesOcupadasDe	:	topologia t × nat ip	$\longrightarrow$ conj(nat)	$\{ip \in compus(t)\}$
conectadas?	:	topologia t × nat ipA × nat ipB	$\longrightarrow$ bool	$\{ipA \in compus(t) \land ipB \in compus(t)\}$
expandirVecinas	:	topologia t × conj(nat) cs	$\longrightarrow$ conj(nat)	$\{cs \subseteq compus(t)\}$
expandirUnPaso	:	topologia t × conj(nat) cs	$\longrightarrow$ conj(nat)	$\{cs \subseteq compus(t)\}$
mapΠ_1	:	conj(tupla(nat × nat)	$\longrightarrow$ conj(nat)	
mapΠ_2	:	conj(tupla(nat × nat)	$\longrightarrow$ conj(nat)	

axiomas $\forall$ t: topologia, $\forall$ $ip, ipBus, ipA, ipB, ifA, ifB, numIfaces$: nat, $\forall$ $ctnn$: conj(tupla(nat, nat)), $\forall$ cs: conj(nat)

compus(NuevaTopo)	$\equiv$	$\emptyset$
compus(Compu(t, ip, $numIfaces$))	$\equiv$	Ag(ip, compus(t))
compus(Cable(t, ipA, ifA, ipB, ifB))	$\equiv$	compus(t)
cablesEn(NuevaTopo, $ipBus$)	$\equiv$	$\emptyset$
cablesEn(Compu(t, ip, $numIfaces$), $ipBus$)	$\equiv$	cablesEn(t, $ipBus$)

cablesEn(Cable(t, ipA, ifA, ipB, ifB), $ipBus$) $\equiv$ **if** $ipBus = ipA$ **then**
Ag($\langle\ ifA,\ ipB\ \rangle$, $\emptyset$)
else
$\emptyset$
fi $\cup$
if $ipBus = ipB$ **then**
Ag($\langle\ ifB,\ ipA\ \rangle$, $\emptyset$)
else
$\emptyset$
fi $\cup$
cablesEn(t, $ipBus$)

numInterfaces(NuevaTopo, $ipBus$) $\equiv$ 0

numInterfaces(Compu(t, ip, $numIfaces$), $ipBus$) $\equiv$ **if** $ipBus = ip$ **then** $numIfaces$ **else** 0 **fi**

numInterfaces(Cable(t, ipA, ifA, ipB, ifB), $ipBus$) $\equiv$ numInterfaces(t)

interfacesOcupadasDe(t, $ipBus$) $\equiv$ mapΠ_1(cablesEn(t, $ipBus$))

vecinasDe(t, $ipBus$) $\equiv$ mapΠ_2(cablesEn(t, $ipBus$))

conectadas?(t, ipA, ipB) $\equiv$ $ipA \in$ expandirVecinas(t, Ag(ipB, $\emptyset$))

expandirVecinas(t, cs) $\equiv$ **if** expandirUnPaso(t, cs) $= cs$ **then**
cs
else
expandirVecinas(t, expandirUnPaso(t, cs))
fi

expandirUnPaso(t, cs) $\equiv$ **if** $\emptyset$?(cs) **then**
$\emptyset$
else
Ag(dameUno(cs), vecinasDe(t, dameUno(cs))) $\cup$ expandirUnPaso(t, sinUno(cs))
fi

mapΠ_1($ctnn$) $\equiv$ **if** $\emptyset$?($ctnn$) **then**
$\emptyset$
else
Ag(Π_1(dameUno($ctnn$)), mapΠ_1(sinUno($ctnn$)))
fi

mapΠ_2($ctnn$) $\equiv$ **if** $\emptyset$?($ctnn$) **then**
$\emptyset$
else
Ag(Π_2(dameUno($ctnn$)), mapΠ_2(sinUno($ctnn$)))
fi

Fin TAD