

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

Reservado para la cátedra

Instancia	Docente	Nota
Primera entrega		
Segunda entrega		

1. TAD DCNET

TAD DCNET

géneros `dcnet`

igualdad observacional

$$(\forall d, d' : \text{dcnet}) \quad d =_{\text{obs}} d' \iff \left(\begin{array}{l} (topo(d) =_{\text{obs}} topo(d')) \wedge \\ ((\forall p : pc)(p \in compus(topo(d)) \wedge p \in compus(topo(d'))) \Rightarrow_L \\ (buffer(d, p) =_{\text{obs}} buffer(d', p) \wedge \\ \#paquetesEnviados(d, p) \\ \#paquetesEnviados(d', p)) \wedge \\ ((\forall p : paquetes)((\exists c : pc)(c \in compus(topo(d')) \wedge \\ c \in compus(topo(d')) \wedge_L (p \in buffer(d, c) \wedge \\ p \in buffer(d', c))) \Rightarrow_L \\ (recorridoPaquete(d, p) =_{\text{obs}} recorridoPaquete(d', p))) \end{array} \right) =_{\text{obs}} \right)$$

generadores

`CrearRed` : `topo` \longrightarrow `dcnet`
`Seg` : `dcnet` \longrightarrow `dcnet`
`CrearPaquete` : `dcnet dcn` \times `pc p1` \times `pc p2` \times `paquete` \longrightarrow `dcnet`
 $\{(p_1 \in compus(topo(dcn)) \wedge p_2 \in compus(topo(dcn))) \wedge_L conectadas?(topo(dcn), p_1, p_2)\}$

observadores básicos

`topo` : `dcnet` \longrightarrow `topologia`
`#paquetesEnviados` : `dcnet dcn` \times `pc p` \longrightarrow `nat` $\{p \in compus(topo(dcn))\}$
`buffer` : `dcnet dcn` \times `pc p` \longrightarrow `conj(paquete)` $\{p \in compus(topo(dcn))\}$
`darPrioridad` : `natid` \longrightarrow `nat` $\{id \in buffers(dcn)\}$

otras operaciones

`recorridoPaquete` : `dcnet dcn` \times `nat id` \longrightarrow `secu(tupla(nat, nat, nat, nat))`
 $\{(paqueteEnTransito?(dcn, id)\}$
`cortarRecHasta` : `sec(tupla(nat \times nat \times nat \times nat))` \times `nat` \longrightarrow `sec(tupla(nat, nat, nat, nat))`
`buscarPaquete` : `dcnet dcn` \times `conj(nat) pcs` \times `nat id` \longrightarrow `nat`
 $\{pcs = compus(topo(dcn)) \wedge (\exists ip : nat)(ip \in pcs \wedge id \in buffer(dcn, ip))\}$
 π_1 `Conj` : `conj(tupla(nat, nat, nat, nat))` \longrightarrow `conj(nat)`
`paqueteEnTransito?` : `dcnet` \times `nat` \longrightarrow `bool`
`existePaqEnBuffers?` : `dcnet dcn` \times `conj(nat) pcs` \times `nat id` \longrightarrow `bool` $\{pcs = compus(topo(dcn))\}$
`perteneceBuffers?` : `paquete` \times `buffers` \longrightarrow `bool`
`compuQueMasEnvio` : `dcnet` \longrightarrow `pc`
`laQueMasEnvio` : `dcnet` \times `conj(pc)` \longrightarrow `pc`
`pasoSeg` : `topo` \times `buffers` \times `buffers` \longrightarrow `buffers`
`regresion` : `topo` \times `buffers` \times `secu(buffers)` \longrightarrow `buffers`
`generarHistoria` : `dcnet` \times `diccionario(pc` \times \longrightarrow `secu(buffers)`
`conj(paquete))`
`auxDefinir` : `buffers` \times `pc` \times `conj(paquete)` \times \longrightarrow `buffers`
`conj(paquete)`
`auxBorrar` : `buffers` \times `pc` \times `conj(paquete)` \times \longrightarrow `buffers`
`conj(paquete)`

transacion	: topo \times buffers \times conj(pc)	\longrightarrow buffers
envio	: topo \times buffers \times ip \times conj(paquete)	\longrightarrow buffers
nuevosPaquetes	: buffers \times buffers	\longrightarrow buffers
darPaqueteEnviado	: conj(paquete)	\longrightarrow paquete
pasarA	: topologia \times pc \times pc	\longrightarrow pc

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

topo(crearRed(t))	$\equiv t$
topo(seg(dcn))	$\equiv \text{topo}(dcn)$
topo(CrearPaquete(dcn, c, c', p))	$\equiv \text{topo}(dcn)$
darPrioridad(seg(dcn, id))	$\equiv \text{darPrioridad}(dcn, id)$
darPrioridad(CrearPaquete(dcn, c, c', p), id)	$\equiv \text{if } id = \Pi_1(p) \text{ then } \Pi_4(p) \text{ else } \text{darPrioridad}(dcn, id)$
#paquetesEnviados(crearRed(t), c)	$\equiv 0$
#paquetesEnviados(seg(dcn), c)	$\equiv \text{\#paquetesEnviados}(dcn)$
#paquetesEnviados(CrearPaquete(dcn, o, d, p), c)	$\equiv \text{if } c = o \text{ then } \text{\#paquetesEnviados}(dcn, c) + 1 \text{ else } \text{\#paquetesEnviados}(dcn, c)$
buffer(dcn, c)	$\equiv \text{obtener}(c, \text{regresion}(\text{topo}(dcn), \text{vacio}, \text{generarHistoria}(dcn, \text{vacio})))$
recorridoPaquete(dcn, p)	$\equiv \text{cortarRecHasta}(\text{darCaminoMasCorto}(\text{topo}(dcn), \text{origen}(p), \text{destino}(p)), \text{buscar}(\text{compus}(\text{topo}(dcn)), p))$
cortarRecHasta(s, ip)	$\equiv \text{if } \text{vacía?}(s) \vee_L ip = ip\text{Origen}(\text{prim}(s)) \text{ then } \langle \rangle \text{ else } \text{prim}(s) \bullet \text{cortarRecHasta}(\text{fin}(s), ip)$
buscarPaquete(dcn, compus, id)	$\equiv \text{if } id \in \pi_1 \text{Conj}(\text{buffer}(dcn, \text{dameUno}(\text{compus}))) \text{ then } \text{dameUno}(\text{compus}) \text{ else } \text{buscarPaquete}(\text{sinUno}(\text{compus}), id)$
$\pi_1 \text{Conj}(\text{conjTuplas})$	$\equiv \text{if } \emptyset?(\text{conjTuplas}) \text{ then } \emptyset \text{ else } \text{Ag}(\pi_1(\text{dameUno}(\text{conjTuplas})), \pi_1 \text{Conj}(\text{sinUno}(\text{conjTuplas})))$
paqueteEnTransito?(dcn, id)	$\equiv \text{existePaqEnBuffers?}(dcn, \text{compus}(\text{topo}(dcn)), id)$
existePaqEnBuffers?(dcn, pcs, id)	$\equiv \text{if } \emptyset?(pcs) \text{ then } \text{false} \text{ else } \text{if } id \in \pi_1 \text{Conj}(\text{buffer}(dcn, \text{dameUno}(pcs))) \text{ then } \text{true} \text{ else } \text{existePaqEnBuffers?}(dcn, \text{sinUno}(pcs), id)$

darPaqueteEnviado(cp)	≡ dameUno(PaquetesConPriopedadK(cp,maxPrioridad(cp)))
maxPrioridad(cp)	≡ max(darPrioridad(dameUno(cp),maxPrioridad(sinUno(cp)))
PaquetesConPriopedadK(cp,k)	≡ if $\emptyset?cp$ then \emptyset else if $darPrioridad(dameUno(cp)) = k$ then $Ag(dameUno(cp), PaquetesConPriopedadK(sinUno(cp), k))$ else $PaquetesConPriopedadK(sinUno(cp), k)$ fi fi
compuQueMasEnvio(dcn)	≡ laQueMasEnvio(dcn,compus(topo(dcn)))
laQueMasEnvio(dcn,cs)	≡ if $\emptyset?(sinUno(cs))$ then $dameUno(cs)$ else if $\#paquetesEnviados(dcn, dameUno(cs)) <$ $\#paquetesEnviados(dcn, laQueMasEnvio$ $(dcn, sinUno(cs)))$ then $laQueMasEnvio(dcn, sinUno(cs))$ else $dameUno(cs)$ fi fi
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
generarHistoria(crearRed(t),bs)	≡ $bs \bullet \langle \rangle$
generarHistoria(seg(dcn),bs)	≡ $bs \bullet generarHistoria(dcn, vaco)$
generarHistoria(CrearPaquete(dcn,o,d,p),bs)	≡ if $def?(c, bs)$ then $generarHistoria(dcn, definir(c, n$ ∪ $obtener(o, bs), bs))$ else $generarHistoria(dcn, definir(c, n))$ 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)	≡ nuevosPaquetes(transacion(t,bs,claves(bs)) ,nbs)
transacion(t,bs,cp)	≡ if $\emptyset?(cp)$ then bs else $transacion(t, envio(t, bs, dameUno(cp)), sinUno(cp))$ fi

pasarA(t,o,d)
 envio(t,bs,ip,cp)

```

≡ prim(caminoMin(t, o, d))
≡ if ∅?(darPaqueteEnviado(cp)) then
    bs
  else
    if pasarA(t, ip, destino(darPaqueteEnviado(cp)) =
      destino(darPaqueteEnviado(cp))) then
      envio(t, quitarPaquete(bs, ip), ip, cp -
        darPaqueteEnviado(cp))
    else
      envio(t, quitarPaquete(pasarPaquete(bs, ip, darPaqueteEnviado(
        bs, ip, cp - darPaqueteEnviado(b)))
    fi
  fi
fi
≡ if ∅?(claves(nbs)) then
    bs
  else
    nuevosPaquetes(auxDefinir(bs, dameUno(claves(nbs), obtener
      (dameUno(claves(nbs), nbs), obtener(dameUno
        (claves(nbs), bs))), sinUno(nbs))
    fi
  fi

```

nuevosPaquetes(bs,nbs)

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

Fin TAD

2. TAD TOPOLOGÍA

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	:		→ topologia
Compu	:	topologia × nat <i>ip</i> × nat	→ topologia {¬(<i>ip</i> ∈ compus(<i>t</i>))}
Cable	:	topologia × nat <i>ipA</i> × nat <i>ifA</i> × nat <i>ipB</i> × nat <i>ifB</i>	→ topologia $\left\{ \begin{array}{l} (ipA \in compus(t) \wedge ipB \in compus(t)) \wedge_L \\ (ifA < \#interfaces(t, ipA)) \wedge \\ (ifB < \#interfaces(t, ipB)) \wedge \\ \neg(ifA \in interfacesOcupadasDe(t, ipA)) \wedge \\ \neg(ifB \in interfacesOcupadasDe(t, ipB)) \wedge \\ \neg(ipA \in vecinas(t, ipB)) \end{array} \right\}$

observadores básicos

compus	:	topologia	→ conj(nat)
cablesEn	:	topologia <i>t</i> × nat <i>ip</i>	→ conj(tupla(nat, nat)) { <i>ip</i> ∈ compus(<i>t</i>)}
#interfaces	:	topologia <i>t</i> × nat <i>ip</i>	→ nat { <i>ip</i> ∈ compus(<i>t</i>)}

otras operaciones

vecinas	:	topologia <i>t</i> × nat <i>ip</i>	→ conj(nat) { <i>ip</i> ∈ compus(<i>t</i>)}
interfacesOcupadasDe	:	topologia <i>t</i> × nat <i>ip</i>	→ conj(nat) { <i>ip</i> ∈ compus(<i>t</i>)}
conectados?	:	topologia <i>t</i> × nat <i>ipA</i> × nat <i>ipB</i>	→ bool { <i>ipA</i> ∈ compus(<i>t</i>) ∧ <i>ipB</i> ∈ compus(<i>t</i>)}
darInterfazConectada	:	conj(tupla(nat, nat)) cablesA × nat <i>ipB</i>	→ nat { <i>ipB</i> ∈ π ₂ Conj(cablesA)}
darSegmento	:	topologia <i>t</i> × nat <i>ipA</i> × nat <i>ipB</i>	→ segmento { <i>ipA</i> ∈ compus(<i>t</i>) ∧ _L <i>ipB</i> ∈ vecinas(<i>t</i> , <i>ipA</i>)}
darCaminoMasCorto	:	topologia <i>t</i> × nat <i>ipA</i> × nat <i>ipB</i>	→ secu(segmento) { <i>ipA</i> ∈ compus(<i>t</i>) ∧ <i>ipB</i> ∈ compus(<i>t</i>) ∧ _L conectados?(<i>t</i> , <i>ipA</i> , <i>ipB</i>)}
darRutas	:	topologia × nat <i>ipA</i> × nat <i>ipB</i> × conj(nat) × se- cu(segmento))	→ conj(secu(segmento)) { <i>ipA</i> ∈ compus(<i>t</i>) ∧ <i>ipB</i> ∈ compus(<i>t</i>)}
darRutasVecinas	:	topologia × conj(nat) × nat <i>ip</i> × conj(nat) × se- cu(segmento))	→ conj(secu(segmento)) { <i>ip</i> ∈ compus(<i>t</i>)}
longMenorSec	:	conj(secu(α))	→ nat
secusDeLongK	:	conj(secu(α)) × nat	→ conj(secu(α))
π ₁ Conj	:	conj(tupla(nat, nat))	→ conj(nat))

$\pi_2 \text{Conj}$	$: \text{conj}(\text{tupla}(\text{nat}, \text{nat})) \longrightarrow \text{conj}(\text{nat})$
axiomas	$\forall t: \text{topologia}, \forall ipNueva, ip, ipA, ipB, ifA, ifB, cantIfaces, k: \text{nat}, \forall conjDuplas: \text{conj}(\text{tupla}(\text{nat}, \text{nat})), \forall conjCablesIpA: \text{conj}(\text{tupla}(\text{nat}, \text{nat})), \forall cs, rec, vecinas: \text{conj}(\text{nat}), \forall secus: \text{conj}(\text{secu}(\alpha)), \forall sc: \text{conj}(\text{secu}(\alpha)), \forall ruta: \text{secu}(\text{segmento})$
$\text{compus}(\text{NuevaTopo})$	$\equiv \emptyset$
$\text{compus}(\text{Compu}(t, ipNueva, cantIfaces))$	$\equiv \text{Ag}(ipNueva, \text{compus}(t))$
$\text{compus}(\text{Cable}(t, ipA, ifA, ipB, ifB))$	$\equiv \text{compus}(t)$
$\text{cablesEn}(\text{NuevaTopo}, ip)$	$\equiv \emptyset$
$\text{cablesEn}(\text{Compu}(t, ipNueva, cantIfaces), ip)$	$\equiv \text{cablesEn}(t, ip)$
$\text{cablesEn}(\text{Cable}(t, ipA, ifA, ipB, ifB), ip)$	$\equiv \text{if } ip = ipA \text{ then } \text{Ag}(\langle ifA, ipB \rangle, \emptyset) \text{ else } \emptyset \text{ fi} \cup \text{if } ip = ipB \text{ then } \text{Ag}(\langle ifB, ipA \rangle, \emptyset) \text{ else } \emptyset \text{ fi} \cup \text{cablesEn}(t, ip)$
$\#interfaces(\text{NuevaTopo}, ip)$	$\equiv 0$
$\#interfaces(\text{Compu}(t, ipNueva, cantIfaces), ip)$	$\equiv \text{if } ip = ipNueva \text{ then } cantIfaces \text{ else } \#interfaces(t) \text{ fi}$
$\#interfaces(\text{Cable}(t, ipA, ifA, ipB, ifB), ip)$	$\equiv \#interfaces(t)$
$\text{interfacesOcupadasDe}(t, ip)$	$\equiv \pi_1 \text{Conj}(\text{cablesEn}(t, ip))$
$\text{vecinas}(t, ip)$	$\equiv \pi_2 \text{Conj}(\text{cablesEn}(t, ip))$
$\text{conectados?}(t, ipA, ipB)$	$\equiv \neg \emptyset?(\text{darRutas}(t, ipA, ipB, \emptyset, <>))$
$\text{darInterfazConectada}(conjCablesIpA, ipB)$	$\equiv \text{if } ipB = \pi_2(\text{dameUno}(conjCablesIpA)) \text{ then } \pi_1(\text{dameUno}(conjCablesIpA)) \text{ else } \text{darInterfazConectada}(\text{sinUno}(conjCablesIpA), ipB) \text{ fi}$
$\text{darSegmento}(t, ipA, ipB)$	$\equiv \langle ipA, \text{darInterfazConectada}(\text{cablesEn}(t, ipA), ipB), ipB, \text{darInterfazConectada}(\text{cablesEn}(t, ipB), ipA) \rangle$
$\text{darCaminoMasCorto}(t, ipA, ipB)$	$\equiv \text{dameUno}(\text{secusDeLongK}(\text{darRutas}(t, ipA, ipB, \emptyset, <>), \text{longMenorSec}(\text{darRutas}(t, ipA, ipB, \emptyset, <>)))$
$\text{darRutas}(t, ipA, ipB, rec, ruta)$	$\equiv \text{if } ipB \in \text{vecinas}(t, ipA) \text{ then } \text{Ag}(ruta \circ \text{darSegmento}(t, ipA, ipB), \emptyset) \text{ else } \text{if } \emptyset?(\text{vecinas}(t, ipA) - rec) \text{ then } \emptyset \text{ else } \text{darRutas}(t, \text{dameUno}(\text{vecinas}(t, ipA) - rec), ipB, \text{Ag}(ipA, rec), ruta \circ \text{darSegmento}(t, ipA, \text{dameUno}(\text{vecinas}(t, ipA) - rec))) \cup \text{darRutasVecinas}(t, \text{sinUno}(\text{vecinas}(t, ipA) - rec), ipB, \text{Ag}(ipA, rec), ruta \circ \text{darSegmento}(t, ipA, \text{dameUno}(\text{vecinas}(t, ipA) - rec))) \text{ fi}$
$\text{darRutasVecinas}(t, vecinas, ipB, rec, ruta)$	$\equiv \text{if } \emptyset?(vecinas) \text{ then } \emptyset \text{ else } \text{darRutas}(t, \text{dameUno}(vecinas), ipB, rec, ruta) \cup \text{darRutasVecinas}(t, \text{sinUno}(vecinas), ipB, rec, ruta) \text{ fi}$

$\text{darCaminoMasCorto}(t, ipA, ipB)$ $\text{secusDeLongK}(\text{secus}, k)$ $\text{longMenorSec}(\text{secus})$ $\pi_1 \text{Conj}(\text{conjDuplas})$ $\pi_2 \text{Conj}(\text{conjDuplas})$	$\equiv \text{dameUno}(\text{secusDeLongK}(\text{darRutas}(t, ipA, ipB, \emptyset, <>),$ $\text{longMenorSec}(\text{darRutas}(t, ipA, ipB, \emptyset, <>)))$ $\equiv \text{if } \emptyset?(\text{secus}) \text{ then}$ $\quad \emptyset$ else $\quad \text{if } \text{long}(\text{dameUno}(\text{secus})) = k \text{ then}$ $\quad \quad \text{dameUno}(\text{secus}) \cup \text{secusDeLongK}(\text{sinUno}(\text{secus}), k)$ $\quad \text{else}$ $\quad \quad \text{secusDeLongK}(\text{sinUno}(\text{secus}), k)$ $\quad \text{fi}$ fi $\equiv \text{if } \emptyset?(\text{secus}) \text{ then}$ $\quad 0$ else $\quad \min(\text{long}(\text{dameUno}(\text{secus})),$ $\quad \quad \text{longMenorSec}(\text{sinUno}(\text{secus})))$ fi $\equiv \text{if } \emptyset?(\text{conjDuplas}) \text{ then}$ $\quad \emptyset$ else $\quad \text{Ag}(\pi_1(\text{dameUno}(\text{conjDuplas})),$ $\quad \quad \pi_1 \text{Conj}(\text{sinUno}(\text{conjDuplas})))$ fi $\equiv \text{if } \emptyset?(\text{conjDuplas}) \text{ then}$ $\quad \emptyset$ else $\quad \text{Ag}(\pi_2(\text{dameUno}(\text{conjDuplas})),$ $\quad \quad \pi_2 \text{Conj}(\text{sinUno}(\text{conjDuplas})))$ fi
--	--

TAD segmento es tupla(nat, nat, nat ,nat)

Fin TAD