

# 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**

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

**generadores**

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

**observadores básicos**

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

**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  
 $\{pcs = \text{compus}(\text{topo}(\text{dcn})) \wedge (\exists ip : \text{nat})(ip \in pcs \wedge id \in \text{buffer}(\text{dcn}, ip))\}$   
 $\pi_1 \text{Conj}$  : 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  $\{pcs = \text{compus}(\text{topo}(\text{dcn}))\}$   
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)	$\equiv \text{obtener}(c, \text{regresion}(\text{topo(dcn)}, \text{vacio}, \text{generarHistoria(dcn, vacio)}))$
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)	$\equiv \text{if } id \in \pi_1 \text{Conj}(\text{buffer(dcn, dameUno(compus))}) \text{ then } \text{dameUno(compus)} \text{ else } \text{buscarPaquete}(\text{sinUno(compus)}, id)$
$\pi_1 \text{Conj}(\text{conjTuplas})$	$\equiv \text{if } \emptyset?(\text{conjTuplas}) \text{ then } \emptyset \text{ else } \text{Ag}(\pi_1(\text{dameUno}(\text{conjTuplas})), \pi_1 \text{Conj}(\text{sinUno}(\text{conjTuplas})))$
paqueteEnTransito?(dcn, id)	$\equiv \text{existePaqEnBuffers?}(dcn, \text{compus}(\text{topo(dcn)}), id)$
existePaqEnBuffers?(dcn, pcs, id)	$\equiv \text{if } \emptyset?(pcs) \text{ then } \text{false} \text{ else } \text{if } id \in \pi_1 \text{Conj}(\text{buffer(dcn, dameUno(pcs))}) \text{ then } \text{true} \text{ else } \text{existePaqEnBuffers?}(dcn, \text{sinUno(pcs)}, id)$

darPaqueteEnviado(cp)	≡ dameUno(PaquetesConPriopedadK(cp,maxPrioridad(cp)))
maxPrioridad(cp)	≡ max(darPrioridad(dameUno(cp),maxPrioridad(sinUno(cp)))
PaquetesConPriopedadK(cp,k)	≡ <b>if</b> $\emptyset?cp$ <b>then</b> $\emptyset$ <b>else</b> <b>if</b> $darPrioridad(dameUno(cp)) = k$ <b>then</b> $Ag(dameUno(cp), PaquetesConPriopedadK(sinUno(cp), k))$ <b>else</b> $PaquetesConPriopedadK(sinUno(cp), k)$ <b>fi</b> <b>fi</b>
compuQueMasEnvio(dcn)	≡ laQueMasEnvio(dcn,compus(topo(dcn)))
laQueMasEnvio(dcn,cs)	≡ <b>if</b> $\emptyset?(sinUno(cs))$ <b>then</b> $dameUno(cs)$ <b>else</b> <b>if</b> $\#paquetesEnviados(dcn, dameUno(cs)) <$ $\#paquetesEnviados(dcn, laQueMasEnvio$ $(dcn, sinUno(cs)))$ <b>then</b> $laQueMasEnvio(dcn, sinUno(cs))$ <b>else</b> $dameUno(cs)$ <b>fi</b> <b>fi</b>
perteneceBuffers?(p,bs)	≡ <b>if</b> $\emptyset?(claves(bs))$ <b>then</b> $false$ <b>else</b> <b>if</b> $p \in obtener(dameUno(claves(bs)), bs)$ <b>then</b> $true$ <b>else</b> $perteneceBuffers?(p, borrar(dameUno(claves(bs)), bs))$ <b>fi</b> <b>fi</b>
generarHistoria(crearRed(t),bs)	≡ $bs \bullet \langle \rangle$
generarHistoria(seg(dcn),bs)	≡ $bs \bullet generarHistoria(dcn, vaco)$
generarHistoria(CrearPaquete(dcn,o,d,p),bs)	≡ <b>if</b> $def?(c, bs)$ <b>then</b> $generarHistoria(dcn, definir(c, n$ <span style="float: right;">∪</span> $obtener(o, bs), bs))$ <b>else</b> $generarHistoria(dcn, definir(c, n))$ <b>fi</b>
auxBorrar(bs,c,b,p)	≡ <b>if</b> $\emptyset?(p - \{b\})$ <b>then</b> $borrar(c, n)$ <b>else</b> $borrar(c, bs) definir(c, p - \{b\}, bs)$ <b>fi</b>
regresion(t,bs,cbs)	≡ <b>if</b> $vacua?(fin(cbs))$ <b>then</b> $pasoSeg(bs, t, prim(cbs))$ <b>else</b> $regresion(t, pasoSeg(bs, t, prim(cbs)), fin(cbs))$ <b>fi</b>
pasoSeg(t,bs,nbs)	≡ nuevosPaquetes(transacion(t,bs,claves(bs)) ,nbs)
transacion(t,bs,cp)	≡ <b>if</b> $\emptyset?(cp)$ <b>then</b> $bs$ <b>else</b> $transacion(t, envio(t, bs, dameUno(cp)), sinUno(cp))$ <b>fi</b>

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
≡ 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

```

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	:	$\longrightarrow$ topologia
Compu	:	$\longrightarrow$ topologia $\{ \neg(ip \in compus(t)) \}$
Cable	:	$\longrightarrow$ 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	:	$\longrightarrow$ conj(nat)
cablesEn	:	$\longrightarrow$ conj(tupla(nat, nat)) $\{ ip \in compus(t) \}$
#interfaces	:	$\longrightarrow$ nat $\{ ip \in compus(t) \}$

#### otras operaciones

vecinas	:	$\longrightarrow$ conj(nat) $\{ ip \in compus(t) \}$
interfacesOcupadasDe	:	$\longrightarrow$ conj(nat) $\{ ip \in compus(t) \}$
conectados?	:	$\longrightarrow$ bool $\{ ipA \in compus(t) \wedge ipB \in compus(t) \}$
darInterfazConectada	:	$\longrightarrow$ nat $\{ ipB \in \pi_2 Conj(cablesA) \}$
darSegmento	:	$\longrightarrow$ segmento $\{ ipA \in compus(t) \wedge ipB \in vecinas(t, ipA) \}$
darCaminoMasCorto	:	$\longrightarrow$ secu(segmento) $\{ ipA \in compus(t) \wedge ipB \in compus(t) \wedge_L conectados?(t, ipA, ipB) \}$
darRutas	:	$\longrightarrow$ conj(secu(segmento))
darRutasVecinas	:	$\longrightarrow$ conj(secu(segmento))
longMenorSec	:	$\longrightarrow$ nat
secusDeLongK	:	$\longrightarrow$ conj(secu( $\alpha$ ))
$\pi_1$ Conj	:	$\longrightarrow$ conj(nat)
$\pi_2$ Conj	:	$\longrightarrow$ conj(nat)

**axiomas**     $\forall t: \text{topologia}, \forall ipNueva, ip, ipA, ipB, ifA, ifB, cantInterfaces, k: \text{nat}, \forall conjDuplas: \text{conj}(tupla(\text{nat}, \text{nat})), \forall conjCablesIpA: \text{conj}(tupla(\text{nat}, \text{nat})), \forall cs, rec, vecinas: \text{conj}(\text{nat}), \forall secus: \text{conj}(\text{secu}(\alpha)), \forall sc: \text{conj}(\text{secu}(\alpha)), \forall ruta: \text{secu}(\text{segmento})$

```

compus(NuevaTopo) ≡ ∅
compus(Compu(t, ipNueva, cantInterfaces)) ≡ Ag(ipNueva, compus(t))
compus(Cable(t, ipA, ifA, ipB, ifB)) ≡ compus(t)
cablesEn(NuevaTopo, ip) ≡ ∅
cablesEn(Compu(t, ipNueva, cantInterfaces), ip) ≡ cablesEn(t, ip)
cablesEn(Cable(t, ipA, ifA, ipB, ifB), ip) ≡ if ip = ipA then Ag(ifA, ipB), ∅ else ∅ fi ∪
    if ip = ipB then Ag(ifB, ipA), ∅ else ∅ fi ∪
    cablesEn(t, ip)

#interfaces(NuevaTopo, ip) ≡ 0
#interfaces(Compu(t, ipNueva, cantInterfaces), ip) ≡ if ip = ipNueva then
    cantInterfaces
else
    #interfaces(t)
fi

#interfaces(Cable(t, ipA, ifA, ipB, ifB), ip) ≡ #interfaces(t)
interfacesOcupadasDe(t, ip) ≡ π1Conj(cablesEn(t, ip))
vecinas(t, ip) ≡ π2Conj(cablesEn(t, ip))
conectados?(t, ipA, ipB) ≡ ¬ ∅?(darRutas(t, ipA, ipB, ∅, <>))
darInterfazConectada(conjCablesIpA, ipB) ≡ if ipB = π2(dameUno(conjCablesIpA)) then
    π1(dameUno(conjCablesIpA))
else
    darInterfazConectada(sinUno(conjCablesIpA), ipB)
fi

darSegmento(t, ipA, ipB) ≡ <ipA, darInterfazConectada(cablesEn(t, ipA), ipB),
    ipB, darInterfazConectada(cablesEn(t, ipB), ipA)>
darCaminoMasCorto(t, ipA, ipB) ≡ dameUno(secusDeLongK(darRutas(t, ipA, ipB, ∅, <>),
    longMenorSec(darRutas(t, ipA, ipB, ∅, <>)))

darRutas(t, ipA, ipB, rec, ruta) ≡ if ipB ∈ vecinas(t, ipA) then
    Ag(ruta ∘ darSegmento(t, ipA, ipB), ∅)
else
    if ∅?(vecinas(t, ipA) - rec) then
        ∅
    else
        darRutas(t, dameUno(vecinas(t, ipA) - rec),
            ipB, Ag(ipA, rec),
            ruta ∘ darSegmento(t, ipA, dameUno(vecinas(t, ipA) - rec))) ∪
            darRutasVecinas(t, sinUno(vecinas(t, ipA) - rec),
                ipB, Ag(ipA, rec),
                ruta ∘ darSegmento(t, ipA, dameUno(vecinas(t, ipA) - rec)))
        fi
    fi
fi

darRutasVecinas(t, vecinas, ipB, rec, ruta) ≡ if ∅?(vecinas) then
    ∅
else
    darRutas(t, dameUno(vecinas), ipB, rec, ruta) ∪
    darRutasVecinas(t, sinUno(vecinas), ipB, rec, ruta)
fi

darCaminoMasCorto(t, ipA, ipB) ≡ dameUno(secusDeLongK(darRutas(t, ipA, ipB, ∅, <>),
    longMenorSec(darRutas(t, ipA, ipB, ∅, <>)))

```

```

secusDeLongK(secus, k)
≡ if  $\emptyset?(secus)$  then
     $\emptyset$ 
  else
    if long(dameUno(secus)) = k then
      dameUno(secus)  $\cup$  secusDeLongK(sinUno(secus), k)
    else
      secusDeLongK(sinUno(secus), k)
    fi
  fi

longMenorSec(secus)
≡ if  $\emptyset?(secus)$  then
    0
  else
    min(long(dameUno(secus)),
        longMenorSec(sinUno(secus)))
  fi

 $\pi_1$ Conj(conjDuplas)
≡ if  $\emptyset?(conjDuplas)$  then
     $\emptyset$ 
  else
    Ag( $\pi_1$ (dameUno(conjDuplas)),
         $\pi_1$ Conj(sinUno(conjDuplas)))
  fi

 $\pi_2$ Conj(conjDuplas)
≡ if  $\emptyset?(conjDuplas)$  then
     $\emptyset$ 
  else
    Ag( $\pi_2$ (dameUno(conjDuplas)),
         $\pi_2$ Conj(sinUno(conjDuplas)))
  fi

```

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

**Fin TAD**