

TAD IP es STRING**TAD INTERFAZ es NAT****TAD PRIORIDAD es NAT****TAD ID es NAT****TAD PAQUETE es TUPLA**(ID:ID, PRIORIDAD:PRIORIDAD, ORIGEN:COMPU, DESTINO:COMPU)**TAD COMPU es TUPLA**(IP:IP, INTERFACES:CONJ(INTERFAZ))**TAD RED****géneros** red**exporta** red, igualdad observacional, observadores basicos, generadores, vecinos, usaInterfaz?, caminosMinimos, hayCamino?**igualdad observacional**

$$(\forall r1, r2 : \text{red}) \left(\begin{array}{l} r1 =_{\text{obs}} r2 \iff \\ \text{computadoras}(r1) =_{\text{obs}} \text{computadoras}(r2) \wedge_L \\ (\forall c_1, c_2 : \text{compu}) (c_1 \in \text{computadoras}(r1) \wedge c_2 \in \text{computadoras}(r1)) \Rightarrow_L \\ (\text{conectadas?}(r1, c_1, c_2) =_{\text{obs}} \text{conectadas?}(r2, c_1, c_2)) \wedge_L \\ \text{conectadas?}(r1, c_1, c_2) \Rightarrow_L \\ (\text{interfazUsada}(r1, c_1, c_2) =_{\text{obs}} \text{interfazUsada}(r2, c_1, c_2)) \end{array} \right)$$

observadores básicoscomputadoras : red \longrightarrow conj(compu)conectadas? : red $r \times \text{compu } c_1 \times \text{compu } c_2 \longrightarrow \text{bool}$
 $\{c_1 \in \text{computadoras}(r) \wedge c_2 \in \text{computadoras}(r)\}$ interfazUsada : red $r \times \text{compu } c_1 \times \text{compu } c_2 \longrightarrow \text{interfaz}$ $\{\text{conectadas?}(r, c_1, c_2)\}$ **generadores**iniciarRed : $\longrightarrow \text{red}$ agregarComputadora : red $r \times \text{compu } c \longrightarrow \text{red}$
 $\{(\forall c' : \text{compu}) (c' \in \text{computadoras}(r) \Rightarrow \text{ip}(c) \neq \text{ip}(c'))\}$ conectar : red $r \times \text{compu } c_1 \times \text{interfaz } i_1 \times \text{compu } c_2 \times \text{interfaz } i_2 \longrightarrow \text{red}$
 $\left\{ \begin{array}{l} c_1 \in \text{computadoras}(r) \wedge c_2 \in \text{computadoras}(r) \wedge \\ \text{ip}(c_1) \neq \text{ip}(c_2) \wedge \neg \text{conectadas?}(r, c_1, c_2) \wedge \\ \neg \text{usaInterfaz?}(r, c_1, i_1) \wedge \neg \text{usaInterfaz?}(r, c_2, i_2) \end{array} \right\}$ **otras operaciones**vecinos : red $r \times \text{compu } c \longrightarrow \text{conj}(\text{compu})$ $\{c \in \text{computadoras}(r)\}$ auxVecinos : red $r \times \text{compu } c \times \text{conj}(\text{compu}) \longrightarrow \text{conj}(\text{compu})$
 cc $\{c \in \text{computadoras}(r) \wedge cc \in \text{computadoras}(r)\}$ usaInterfaz? : red $r \times \text{compu } c \times \text{interfaz} \longrightarrow \text{bool}$ $\{c \in \text{computadoras}(r)\}$ interfacesUsadas : red $r \times \text{compu } c \times \text{conj}(\text{compu}) \longrightarrow \text{conj}(\text{interfaz})$
 cc $\{c \in \text{computadoras}(r) \wedge cc \in \text{computadoras}(r)\}$ caminos : red $r \times \text{compu } c_1 \times \text{compu } c_1 \longrightarrow \text{conj}(\text{secu}(\text{compu}))$
 $\{c_1 \in \text{computadoras}(r) \wedge c_2 \in \text{computadoras}(r)\}$ auxCaminos : red $r \times \text{compu } c_1 \times \text{compu } c_1 \times \text{secu}(\text{compu}) \times \text{conj}(\text{compu}) \longrightarrow \text{conj}(\text{secu}(\text{compu}))$
 cc $\{c_1 \in \text{computadoras}(r) \wedge c_2 \in \text{computadoras}(r) \wedge cc \in \text{computadoras}(r)\}$ hayCamino? : red $r \times \text{compu } c_1 \times \text{compu } c_2 \longrightarrow \text{bool}$
 $\{c_1 \in \text{computadoras}(r) \wedge c_2 \in \text{computadoras}(r)\}$ caminosMinimos : red $r \times \text{compu } c_1 \times \text{compu } c_2 \longrightarrow \text{conj}(\text{secu}(\text{compu}))$
 $\{c_1 \in \text{computadoras}(r) \wedge c_2 \in \text{computadoras}(r)\}$ auxMinimos : conj(secu(compu)) $\longrightarrow \text{conj}(\text{secu}(\text{compu}))$ **axiomas**computadoras(iniciarRed) $\equiv \emptyset$ computadoras(agregarComputadora(r, c)) $\equiv \text{Ag}(c, \text{computadoras}(r))$ computadoras(conectar(r, c₁, i₁, c₂, i₂)) $\equiv \text{computadoras}(r)$

```

conectadas?(agregarComputadora(r, c), c1, c2)    ≡ ¬(c = c1 ∨ c = c2) ∨ conectadas?(r, c1, c2)
conectadas?(conectar(r, c'1, i1, c'2, i2), c1, c2)    ≡ (c1 = c'1 ∧ c2 = c'2) ∨ (c1 = c'2 ∧ c2 = c'1) ∨
    conectadas?(r, c1, c2)

interfazUsada(agregarComputadora(r, c), c1, c2)    ≡ interfazUsada(r, c1, c2)
interfazUsada(conectar(r, c'1, i1, c'2, i2), c1, c2)    ≡ if (c1 = c'1 ∧ c2 = c'2) then
    i1
else
    if (c1 = c'2 ∧ c2 = c'1) then
    i2
    else
        interfazUsada(r, c1, c2)
    fi
fi

vecinos(r, c)    ≡ auxVecinos(r, c, computadoras(r))

auxVecinos(r, c, cs)    ≡ if ∅?(cs) then
    ∅
else
    if conectados?(r, c, dameUno(cs)) then
        Ag(dameUno(cs), auxVecinos(r, c, sinUno(cs)))
    else
        auxVecinos(r, c, sinUno(cs))
    fi
fi

usaInterfaz?(r, c, i)    ≡ i ∈ interfacesUsadas(r, c, computadoras(r))

interfacesUsadas(r, c, cs)    ≡ if ∅?(cs) then
    ∅
else
    if conectados?(r, c, dameUno(cs)) then
        Ag(interfazUsada(r, c, dameUno(cs)),
            interfacesUsadas(r, c, sinUno(cs)))
    else
        FiltrarInterfacesUsadas(SinUno(ci), c)
    fi
fi

caminos(r, c1, c2)    ≡ auxCaminos(r, c1, c2, c1 • <>, vecinos(r, c1))

auxCaminos(r, c1, c2, recorrido, candidatos)    ≡ if ∅?(candidatos) then
    ∅
else
    if ult(recorrido) = c2 then
        ag(recorrido, ∅)
    else
        if ¬esta?(dameUno(candidatos), recorrido)
        then
            auxCaminos(r, c1, c2,
                recorrido o dameUno(candidatos),
                vecinos(r, dameUno(candidatos)))
            ∪ auxCaminos(r, c1, c2, recorrido,
                sinUno(candidatos))
        else
            auxCaminos(r, c1, c2, recorrido,
                sinUno(candidatos))
        fi
    fi
fi

hayCamino?(r, c1, c2)    ≡ #(caminos(r, c1, c2)) > 0

```

```

caminosMinimos(r, c1, c2)                ≡ auxMinimos(camino(r, c1, c2))

auxMinimos(cc) ≡ if ∅?(cc) then
  ∅
else
  if #(cc) = 1 then
    ag(dameUno(c), ∅)
  else
    if long(dameUno(cc)) < long(dameUno(auxMinimos(sinUno(cc)))) then
      ag(dameUno(cc), ∅)
    else
      if long(dameUno(cc)) = long(dameUno(auxMinimos(sinUno(cc)))) then
        ag(dameUno(c), auxMinimos(sinUno(cc)))
      else
        auxMinimos(sinUno(cc))
    fi
  fi
fi

```

Fin TAD

TAD DCNET

exporta dcnet, igualdad observacioanl, observadores basicos, generadores, paqueteEnTransito?, laQueMasEnvio

géneros dcnet

igualdad observacional

$$(\forall s1, s2 : \text{dcnet}) \left(\begin{array}{l} s1 =_{\text{obs}} s2 \iff \\ \text{red}(s1) =_{\text{obs}} \text{red}(s2) \wedge_L \\ (\forall c : \text{ip})(c \in \text{computadoras}(\text{red}(s1)) \Rightarrow_L \text{EnEspera}(s1, c) =_{\text{obs}} \text{EnEspera}(s2, c)) \wedge_L \\ (\forall p : \text{paquete})((\text{paqueteEnTransito?}(s1, p)) \Rightarrow_L \\ \quad \text{caminoRecorrido}(s1, p) =_{\text{obs}} \text{caminoRecorrido}(s2, p)) \wedge_L \\ (\forall c : \text{ip})((c \in \text{computadoras}(\text{red}(s1))) \Rightarrow_L \\ \quad \text{cantidadEnviados}(s1, c) =_{\text{obs}} \text{cantidadEnviados}(s2, c)) \end{array} \right)$$

observadores básicos

red	: dcnet	→ red	
caminoRecorrido	: dcnet $s \times \text{paquete } p$	→ secu(compu)	$\{\text{paqueteEnTransito?}(s, p)\}$
cantidadEnviados	: dcnet $s \times \text{compu } c$	→ nat	$\{c \in \text{computadoras}(\text{red}(s))\}$
enEspera	: dcnet $s \times \text{compu } c$	→ conj(paquete)	$\{c \in \text{Computadoras}(\text{Red}(s))\}$

generadores

iniciarDCNet	: red	→ dcnet	
crearPaquete	: dcnet $s \times \text{paquete } p$	→ dcnet	
			$\left\{ \begin{array}{l} \neg(\exists p' : \text{paquete}) (\text{paqueteEnTransito}(s, p') \wedge \text{id}(p') = \text{id}(p)) \\ \wedge \text{origen}(p) \in \text{computadoras}(\text{red}(s)) \\ \wedge_L \text{destino}(p) \in \text{computadoras}(\text{red}(s)) \\ \wedge_L \text{hayCamino?}(\text{red}(s), \text{origen}(p), \text{destino}(p)) \end{array} \right\}$
avanzarSegundo	: dcnet	→ dcnet	

otras operaciones

proximoPaquete	: dcnet $s \times \text{compu } c$	→ paquete	$\{c \in \text{computadoras}(\text{Red}(s)) \wedge_L \#(\text{enEspera}(s, c)) > 0\}$
prioritarios	: conj(paquete)	→ conj(paquete)	
computadoraActual	: dcnet $s \times \text{paquete } p$	→ compu	$\{\text{paqueteEnTransito?}(s, p)\}$
proximaComputadora	: dcnet $s \times \text{paquete } p$	→ compu	$\{\text{paqueteEnTransito?}(s, p)\}$
paquetesConOtroDestino	: conj(paquete) \times compu	→ conj(paquete)	

paquetesEntrantes	: $\text{dcnet } s \times \text{compu } c \times \text{conj}(\text{compu}) \text{ } cc \longrightarrow \text{conj}(\text{paquete})$ $\{c \in \text{computadoras}(\text{red}(s)) \wedge cc \subseteq \text{computadoras}(\text{red}(s))\}$
paqueteEnTransito?	: $\text{dcnet} \times \text{paquete} \longrightarrow \text{bool}$
laQueMasEnvio	: $\text{dcnet} \longrightarrow \text{compu}$
lasQueMasEnviaron	: $\text{dcnet } s \times \text{conj}(\text{compu}) \text{ } cc \longrightarrow \text{conj}(\text{compu})$ $\{cc \subseteq \text{computadoras}(\text{red}(s))\}$

axiomas

```

red(iniciarDCNet( $r, n$ ))  $\equiv r$ 
red(crearPaquete( $s, p$ ))  $\equiv \text{red}(s)$ 
red(avanzarSegundo( $s$ ))  $\equiv \text{red}(s)$ 

caminoRecorrido(crearPaquete( $s, p'$ ),  $p$ )  $\equiv$  if  $p = p'$  then
    origen( $p$ ) • <>
else
    caminoRecorrido( $s, p$ )
fi
caminoRecorrido(avanzarSegundo( $s$ ),  $p$ )  $\equiv$  if proximoPaquete( $s, \text{computadoraActual}(s, p)$ )  $\neq p$  then
    caminoRecorrido( $s, p$ )
else
    caminoRecorrido( $s, p$ ) o proximaComputadora( $s, p$ )
fi

cantidadEnviados(iniciarDCNet( $r, n$ ),  $c$ )  $\equiv 0$ 
cantidadEnviados(crearPaquete( $s, p$ ),  $c$ )  $\equiv \text{cantidadEnviados}(s, c)$ 
cantidadEnviados(avanzarSegundo( $s$ ),  $c$ )  $\equiv$  (if  $\#(\text{enEspera}(s, c)) > 0$  then 1 else 0 fi) +
    cantidadEnviados( $s, c$ )

enEspera(iniciarDCNet( $r, n$ ),  $c$ )  $\equiv \emptyset$ 
enEspera(crearPaquete( $s, p$ ),  $c$ )  $\equiv$  if origen( $p$ ) =  $c$  then
    ag( $p$ , enEspera( $s, c$ ))
else
    enEspera( $s, c$ )
fi
enEspera(avanzarSegundo( $s$ ),  $c$ )  $\equiv$  (enEspera( $s, c$ ) - (if  $\#(\text{enEspera}(s, c)) > 0$ 
then proximoPaquete( $s, c$ ) else  $\emptyset$  fi))
     $\cup$  paquetesConOtroDestino(paquetesEntrantes( $s, c$ ,
    computadoras( $\text{red}(s)$ )),  $c$ )

proximoPaquete( $s, c$ )  $\equiv \text{dameUno}(\text{prioritarios}(\text{enEspera}(s, c)))$ 

prioritarios( $cp$ )  $\equiv$  if  $\emptyset?(cp)$  then
     $\emptyset$ 
else
    if  $\#(cp) = 1$  then
         $cp$ 
    else
        if prioridad(dameUno( $cp$ )) < prioridad(dameUno(prioritarios(sinUno( $cp$ ))))
        then
            ag(dameUno( $cp$ ),  $\emptyset$ )
        else
            if prioridad(DameUno( $cp$ )) = prioridad(dameUno(prioritarios(sinUno( $cp$ ))))
            then
                ag(dameUno( $cp$ ), prioridades(sinUno( $cp$ )))
            else
                prioridades(sinUno( $cp$ ))
            fi
        fi
    fi
fi

```

computadoraActual(s, p) \equiv ult(caminoRecorrido(s, p))

proximaComputadora(s, p) \equiv prim(fin(dameUno(caminosMinimos(red(s), computadoraActual(s, p), destino(p))))))

paquetesConOtroDestino(cp, c) \equiv **if** $\emptyset?(cc)$ **then**
 \emptyset
else
if destino(dameUno(cp)) = c **then**
paquetesConOtroDestino(sinUno(cp), c)
else
ag(dameUno(cp), paquetesConOtroDestino(sinUno(cp), c))
fi
fi

paquetesEntrantes(s, c, cc) \equiv **if** $\emptyset?(cc)$ **then**
 \emptyset
else
(if #(enEspera(s , dameUno(cc))) > 0 **then**
if proximaComputadora(s , proximoPaquete(s , dameUno(cc))) = c
then
ag(proximoPaquete(s , dameUno(cc)), \emptyset)
else
 \emptyset
fi
else
 \emptyset
fi) \cup paquetesEntrantes(s, c , sinUno(cc))
fi

paqueteEnTransito?(iniciarDCNet(r, n), p) \equiv false

paqueteEnTransito?(crearPaquete(s, p'), p) \equiv $p = p' \vee$ paqueteEnTransito?(s, p)

paqueteEnTransito?(avanzarSegundo(s)) \equiv paqueteEnTransito?(s, p) \wedge_L
(proximoPaquete(s , computadoraActual(s, p))) $\neq p \vee$
proximaComputadora(s, p) \neq destino(p)

laQueMasEnvio(s) \equiv dameUno(lasQueMasEnviaron(s , computadoras(Red(s))))

lasQueMasEnviaron(s, cc) \equiv **if** $\emptyset?(cc)$ **then**
 \emptyset
else
if #(cc) = 1 **then**
 cc
else
if cantidadEnviados(s , dameUno(cc)) > cantidadEnviados(s ,
dameUno(lasQueMasEnviaron(sinUno(cc)))) **then**
ag(dameUno(cc), \emptyset)
else
if cantidadEnviados(s , dameUno(cc)) = cantidadEnviados(s ,
dameUno(lasQueMasEnviaron(sinUno(cc)))) **then**
ag(dameUno(cc), lasQueMasEnviaron(sinUno(cc)))
else
lasQueMasEnviaron(sinUno(cc))
fi
fi
fi
fi

Fin TAD