



DEPARTAMENTO  
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

## Trabajo Práctico 2: Diseño

Primer cuatrimestre - 2015

Algoritmos y Estructuras de Datos II

### Grupo 2

Integrante	LU	Correo electrónico
Benitez, Nelson	945/13	nelson.benitez92@gmail.com

Instancia	Docente	Nota
Primera entrega		
Segunda entrega		



**Facultad de Ciencias Exactas y Naturales**  
Universidad de Buenos Aires

Ciudad Universitaria – Pabellón I (Planta Baja)

Intendente Güiraldes 2160 – C1428EGA

Ciudad Autónoma de Buenos Aires – Rep. Argentina

Tel/Fax: (+54 +11) 4576-3300

<http://www.exactas.uba.ar>

# Índice

# 1 DCNet

## 1.1 Interfaz

se explica con AS

usa

géneros as

### Operaciones

CREARSISTEMA(**in**  $r : \text{red}$ )  $\longrightarrow res : \text{dcnet}$

**Pre**  $\equiv \{\text{true}\}$

**Post**  $\equiv \{res =_{\text{obs}} \text{iniciarDCNet}(r)\}$

**Descripción:** Crea un sistema DCNet.

**Complejidad:**  $O(L \times n^5)$

**Aliasing:**  $res.red$  es un puntero a la red que recibimos por parámetro

Las complejidades están en función de las siguientes variables:

$n$  : la cantidad total de computadoras que hay en el sistema,

$L$  : el hostname más largo de todas las computadoras,

$k$  : la cola de paquetes más larga de todas las computadoras.

## 1.2 Representación

se representa con sistema

donde sistema es  $\text{tupla}\langle \text{Compus} : \text{arreglo}(\text{arreglo}(\text{tupla}\langle \text{hayHippie} : \text{bool}, , \text{tupla}\langle \text{pl} : \text{nat}, \text{nombre} : \text{string} \rangle),$   
 $\text{hayEst} : \text{bool}, \text{nombre} : \text{string} \rangle,$   
 $\text{hayAgente} : \text{bool} \rangle$   
 $\text{estudiantes} : \text{DiccTrie}(\text{nombre} : \text{string}, \text{pos} : \text{pos}),$   
 $\text{hippies} : \text{DiccTrie}(\text{nombre} : \text{string}, \text{pos} : \text{pos}),$   
 $\text{agentes} : \text{DiccHash}(\text{pl} : \text{nat}, \text{tupla}\langle \text{pos} : \text{pos}, \text{cantSanc} : \text{nat},$   
 $\text{cantCapturas} : \text{nat} \rangle$   
 $\text{masVigilante} : \text{placa} : \text{nat},$   
 $\text{conKSanciones} : \text{tupla}\langle \text{ocurrioSancion} : \text{bool}, \text{porKSanc} : \text{arreglo}(\text{placa}) \rangle$

### Invariante de representación

1. Todos los IP de *compus* pertenecen al conjunto de claves de *CompusPorPref* y la longitud de dicho arreglo es igual al cardinal de las claves del diccionario.
2. Los pN de las tuplas que tiene el arreglo *compus* apuntan al conjunto de paquetes(PorNom) de un significado en *CompusPorPref* cuya clave es igual al IP de esa posición en el arreglo.
3. Los pN' apuntan al conjunto de paquetes(porNom') de un significado en *CompusPorPref* cuya clave es igual al IP de esa posición en el arreglo

4. Los paquetes del significado  $pN'$  son iguales a los paquetes de  $pN$
5. El origen de  $pN'$  es distinto al destino de  $pN'$  y ambos son posiciones válidas del arreglo compus
6. PosActual de  $pN'$  es una posición válida del arreglo compus
7. La  $\#PaquetesEnviados$  de cada compu es mayor o igual a la actual cantidad total de paquetes que pasaron por esa compu
8. Todos los conjuntos de los significados de *CompusPorPref* son disjuntos dos a dos.
9. Los conjuntos de los campos de la tupla PorNom, PorPrior son iguales.
10. La matriz de caminosMinimos es cuadrada de lado  $n$ , con  $n$  igual al tamaño del arreglo de compus.
11. Para cualquier compu en el sistema  $f, d$  caminosMinimos[f][d] se corresponde con caminoMinimo(red, f, d)
12. La longitud de *CaminosMinimos* es igual a la longitud del arreglo que tiene *CaminosMinimos* en cada posición.
13. La longitud del arreglo, que tiene un arreglo de *CaminosMinimos* es menor o igual a la longitud de *CaminosMinimos*.
14. Los elementos del arreglo anteriormente mencionado son IPs del diccionario *CompusPorPref* y no tiene repetidos.
15. La computadora que más paquetes envió es aquella cuyo índice es igual a *LaQMasEnvio*

$\text{Rep} : \widehat{\text{sistema}} \rightarrow \text{boolean}$

$(\forall s : \widehat{\text{sistema}})$

$\text{Rep}(s) \equiv$

1.  $\forall s : \text{String} \text{ def?}(s, s.\text{CompusPorPref}), (\exists c : \text{compu}), \text{esta?}(c, s.\text{Compus}) \wedge \pi_1(c) = s \wedge \text{longitud}(s.\text{Compus}) = \#CLAVES(s.\text{CompusPorPref})$
2.  $\forall c : \text{compu} \text{ esta?}(c, s.\text{Compus}), * \pi_2(c) = \text{obtener}(\pi_1(c), s.\text{CompusPorPref})$
3.  $\forall c : \text{compu} \text{ esta?}(c, s.\text{Compus}), * \pi_3(c) = \text{obtener}(\pi_3(c), s.\text{CompusPorPref})$
- 4, 5, 6.
- $(\forall c : \text{nat}) 0 \leq c < \text{Longitud}(s.\text{compus}) \Rightarrow_L$   
 $\text{Longitud}(s.\text{compus}[c].pN) = \text{Longitud}(s.\text{compus}[c].pN') \wedge$   
 $(\forall p : \text{paquetePos}) \text{esta?}(p, s.\text{compus}[c].pN') \Rightarrow_L$   
 $\text{esta}(\pi_1(p), s.\text{compus}[c].pN) \wedge 0 \leq \text{indiceOrigen}(p) < \text{Longitud}(s.\text{compus})$   
 $\wedge 0 \leq \text{indiceDestino}(p) < \text{Longitud}(s.\text{compus})$   
 $\wedge 0 \leq \text{posActual}(p) < \text{Longitud}(s.\text{compus})$   
 $\wedge \neg(\text{indiceDestino}(p) = \text{indiceOrigen}(p))$
7.  $(\forall c : \text{nat}) 0 \leq c < \text{Longitud}(s.\text{compus}) \Rightarrow_L$   
 $(\forall p : \text{paquetePos}) \text{pertenece}(s.\text{compus}[c].pN', p) \Rightarrow_L$   
 $\beta(\text{esta}(s.\text{compus}[c], \text{caminoMinimo}(s.\text{red}, s.\text{compus}[\text{indiceOrigen}(p)], s.\text{compus}[\text{posActual}(p)])))$
8.  $\forall s, t : \text{String} \text{ def?}(s, s.\text{CompusPorPref}) \wedge \text{def?}(t, t.\text{CompusPorPref}) \wedge s \neq t \Rightarrow_L$   
 $\text{obtener}(s, s.\text{CompusPorPref}) \cap \text{obtener}(t, t.\text{CompusPorPref}) = \emptyset$
9.  $\forall s : \text{String} \text{ def?}(s, s.\text{CompusPorPref}) \Rightarrow_L \pi_1(\text{obtener}(s, s.\text{CompusPorPref})) =$   
 $\pi_2(\text{obtener}(s, s.\text{CompusPorPref}))$
10.  $\text{Longitud}(s.\text{compus}) = \text{Longitud}(\text{CaminosMinimos}(s)) \wedge$   
 $(\forall i : \text{nat}) 0 \leq i < \text{Longitud}(s.\text{compus}) \Rightarrow_L$   
 $\text{Longitud}(s.\text{CaminosMinimos}[i]) = \text{Longitud}(s.\text{compus})$
11.  $(\forall f, d : \text{nat}) \neg(f = d) \wedge 0 \leq f, d < \text{Longitud}(s.\text{compus}) \Rightarrow_L$   
 $\text{CaminosMinimos}[f][d] =$

$caminoMinimo(s.red, ipACompu(s.red, \pi_1(s.comp[ f ])), ipACompu(s.red, \pi_1(s.comp[ d ])))$   
 12, 13, 14.  $(\forall i, j : \text{nat}), 0 \leq i, j < longitud(s.CaminosMinimos) \Rightarrow_L longitud(s.CaminosMinimos) =$   
 $longitud(s.CaminosMinimos[i]) \wedge longitud(s.CaminosMinimos[i][j]) < longitud(s.CaminosMinimos) \wedge$   
 $(\forall e : \text{nat}), esta?(e, s.CaminosMinimos[i][j]) \Rightarrow pertenece(e, s.Comp[ PorPref ])$   
 15.  $\forall c : \text{compu } esta?(c, s.Comp) \Rightarrow_L \pi_3(c) \leq \pi_3(s.Comp[ s.LaQMasEnvio ])$

## Función de abstracción

$Abs : \widehat{dcnet} \ s \longrightarrow \widehat{DCNet} \qquad \{Rep(s)\}$

$(\forall s : \widehat{dcnet})$

$Abs(s) \equiv dc : \widehat{DCNet} \mid$

$red(dc) =^*(s.red) \wedge (\forall c : \text{compu}, c \in comp[ dc ]) (enEspera(dc, c) =^*(enEspera(s, c)) \wedge$

$cantidadEnviados(dc, c) = cantidadEnviados(s, c)) \wedge$

$(\forall p : \text{paquete}, paqueteEnTransito?(dc, p)) caminoRecorrido(dc, p) =^*(caminoRecorrido(s, p))$

### 1.3 Algoritmos

<b>ICREARSISTEMA</b> ( <b>in</b> $r : \text{red}$ ) $\longrightarrow res : \text{dcnet}$	
$res.red \leftarrow r$	
$n \leftarrow \text{Longitud}(\text{COMPUS}(red))$	$O(1)$
$i \leftarrow 0$	
$j \leftarrow 0$	$O(1)$
$res.Compūs \leftarrow \text{CREARARREGLO}(n)$	$O(n)$
$res.CaminosMinimos \leftarrow \text{CREARARREGLO}(n)$	$O(n)$
<b>var</b> $p : \text{arreglo\_dimensionable de puntero}(\text{conjLog}(\text{paquete}))$	
<b>while</b> $i < n$ <b>do</b>	$O(L * n^5)$
	$O(n)$
$res.CaminosMinimos[i] \leftarrow \text{CREARARREGLO}(n)$	$O(n)$
$s : < nat, conjLog(paquete, <_{id}), conjLog(paquete, <_p),$	
$conjLog(paquetePos, <_{id}), conjLog(paquetePos, <_p) >$	
$\pi_1(s) \leftarrow compu(r, i)$	
$\pi_2(s) \leftarrow \text{NUEVO}()$	
$\pi_3(s) \leftarrow \text{NUEVO}()$	
$\pi_4(s) \leftarrow \text{NUEVO}()$	
$\pi_5(s) \leftarrow \text{NUEVO}()$	
$\text{DEFINIR}(res.CompūsPorPref, compu(r, i), s)$	$O(L)$
$p[i] \leftarrow \pi_3(s)$	
$p'[i] \leftarrow \pi_5(s)$	
$res.Compūs[i] \leftarrow < compu(r, i), p[i], p'[i], 0 >$	$O(1)$
<b>while</b> $j < n$ <b>do</b>	$O(L * n^4)$
	$O(n)$
$res.CaminosMinimos[i][j] \leftarrow caminoMinimo(compu(r, i), compu(r, j), r)$	$O(L * n^3)$
$j++$	
<b>end while</b>	
$i++$	
<b>end while</b>	
$res.LaQMasEnvio \leftarrow 0$	$O(1)$
<hr/>	
	$O(L \times n^5)$
<b>ICREARPAQUETE</b> ( <b>in/out</b> $s : \text{dcnet}$ , <b>in/out</b> $p : \text{paquete}$ )	
$t_1 : < nat, conjLog(paquete, <_{id}), conjLog(paquete, <_p),$	
$conjLog(paquetePos, <_{id}), conjLog(paquetePos, <_p) >$	
$t_1 \leftarrow \text{OBTENER}(\text{origen}(p), s.CompūsPorPref)$	$O(L)$
$t_2 : < nat, conjLog(paquete, <_{id}), conjLog(paquete, <_p),$	
$conjLog(paquetePos, <_{id}), conjLog(paquetePos, <_p) >$	
$t_2 \leftarrow \text{OBTENER}(\text{destino}(p), s.CompūsPorPref)$	$O(L)$
$p' : paquetePos$	
$\text{INDICEORIGEN}(p') \leftarrow \pi_1(t_1)$	$O(1)$
$\text{INDICEDESTINO}(p') \leftarrow \pi_1(t_2)$	$O(1)$
$\text{POSACTUAL}(p') \leftarrow 0$	
$\text{INSERTAR}(\pi_2(t), p)$	$O(\log(k))$
$\text{INSERTAR}(\pi_3(t), p)$	$O(\log(k))$
$\text{INSERTAR}(\pi_4(t), p')$	$O(\log(k))$
$\text{INSERTAR}(\pi_5(t), p')$	$O(\log(k))$
<hr/>	
	$O(L + \log(k))$

## 1.4 Servicios Usados

Del modulo ConjLog requerimos pertenece, buscar, menor, insertar y borrar en  $O(\log(k))$ .

Del modulo Diccionario Por Prefijos requerimos Def?, obtener en  $O(L)$ .

## 2 DCNet

## 2.1 Interfaz

se explica con AS

usa

gêneros as

## Operaciones

$$\text{CREARSISTEMA}(\text{in } r : \text{red}) \longrightarrow res : \text{dcnet}$$
$$\mathbf{Pre} \equiv \{\text{true}\}$$
$$\mathbf{Post} \equiv \{res =_{\text{obs}} \text{iniciar} DCNet(r)\}$$

**Descripción:** Crea un sistema DCNet.

**Complejidad:**  $O(L \times n^5)$

**Aliasing:** res.red es un puntero a la red que recibimos por parámetro

Las complejidades están en función de las siguientes variables:

$n$  : la cantidad total de computadoras que hay en el sistema,

$L$  : el hostname más largo de todas las computadoras,

$k$  : la cola de paquetes más larga de todas las computadoras.

## 2.2 Representación

se representa con sistema

```

donde sistema es tupla⟨Compus : arreglo(tupla⟨IP : String,
pN : puntero(conjLog(paquete)),
pN' : puntero(conjLog(paquetePos)),
#PaquetesEnviados : nat⟩
CompusPorPref : diccPref(compu, tupla⟨PorNom : conjLog(paquete),
PorPrior : conjLog(paquete),
PorNom' : conjLog(paquetePos),
PorPrior' : conjLog(paquetePos))⟩
CaminosMinimos : arreglo(arreglo(arreglo(compu))),
LaQMasEnvio : nat,
Red : red⟩

```

## Invariante de representación

1. Todos los IP de *compus* pertenecen al conjunto de claves de *CompusPorPref* y la longitud de dicho arreglo es igual al cardinal de las claves del diccionario.
2. Los pN de las tuplas que tiene el arreglo *compus* apuntan al conjunto de paquetes(PorNom) de un significado en *CompusPorPref* cuya clave es igual al IP de esa posición en el arreglo.
3. Los pN' apuntan al conjunto de paquetes(porNom') de un significado en *CompusPorPref* cuya clave es igual al IP de esa posición en el arreglo
4. Los paquetes del significado pN' son iguales a los paquetes de pN
5. El origen de pN' es distinto al destino de pN' y ambos son posiciones válidas del arreglo *compus*
6. PosActual de pN' es una posicion válida del arreglo *compus*
7. La *#PaquetesEnviados* de cada compu es mayor o igual a la actual cantidad total de paquetes que pasaron por esa compu
8. Todos los conjuntos de los significados de *CompusPorPref* son disjuntos dos a dos.
9. Los conjuntos de los campos de la tupla PorNom, PorPrior son iguales.
10. La matriz de caminosMinimos es cuadrada de lado n, con n igual al tamaño del arreglo de *compus*.
11. Para cualquier compu en el sistema f,d caminosMinimos[f][d] se corresponde con caminoMinimo(red,f,d)
12. La longitud de *CaminosMinimos* es igual a la longitud del arreglo que tiene *CaminosMinimos* en cada posición.
13. La longitud del arreglo, que tiene un arreglo de *CaminosMinimos* es menor o igual a la longitud de *CaminosMinimos*.
14. Los elementos del arreglo anteriormente mencionado son IPs del diccionario *CompusPorPref* y no tiene repetidos.
15. La computadora que más paquetes envió es aquella cuyo índice es igual a *LaQMasEnvio*

Rep :  $\widehat{\text{sistema}} \rightarrow \text{boolean}$

( $\forall s : \widehat{\text{sistema}}$ )

Rep(s)  $\equiv$

1.  $\forall s : \text{String } \text{def?}(s, s.\text{CompusPorPref}), (\exists c : \text{compu}), \text{esta?}(c, s.\text{Compus}) \wedge \pi_1(c) = s \wedge \text{longitud}(s.\text{Compus}) = \#\text{CLAVES}(s.\text{CompusPorPref})$
2.  $\forall c : \text{compu } \text{esta?}(c, s.\text{Compus}), * \pi_2(c) = \text{obtener}(\pi_1(c), s.\text{CompusPorPref})$
3.  $\forall c : \text{compu } \text{esta?}(c, s.\text{Compus}), * \pi_3(c) = \text{obtener}(\pi_3(c), s.\text{CompusPorPref})$
- 4, 5, 6.

( $\forall c : \text{nat}$ )  $0 \leq c < \text{Longitud}(s.\text{compus}) \Rightarrow_L$

$\text{Longitud}(s.\text{compus}[c].\text{pN}) = \text{Longitud}(s.\text{compus}[c].\text{pN}') \wedge$

( $\forall p : \text{paquetePos}$ )  $\text{esta?}(p, s.\text{compus}[c].\text{pN}') \Rightarrow_L$

$\text{esta}(\pi_1(p), s.\text{compus}[c].\text{pN}) \wedge 0 \leq \text{indiceOrigen}(p) < \text{Longitud}(s.\text{compus})$

$\wedge 0 \leq \text{indiceDestino}(p) < \text{Longitud}(s.\text{compus})$

$\wedge 0 \leq \text{posActual}(p) < \text{Longitud}(s.\text{compus})$

$\wedge \neg(\text{indiceDestino}(p) = \text{indiceOrigen}(p))$

7. ( $\forall c : \text{nat}$ )  $0 \leq c < \text{Longitud}(s.\text{compus}) \Rightarrow_L$

( $\forall p : \text{paquetePos}$ )  $\text{pertenece}(s.\text{compus}[c].\text{pN}', p) \Rightarrow_L$



- $\beta(\text{esta}(s.\text{compus}[c], \text{caminoMinimo}(s.\text{red}, s.\text{compus}[\text{indiceOrigen}(p)], s.\text{compus}[\text{posActual}(p)])))$
8.  $\forall s, t : \text{String} \text{ def?}(s, s.\text{CompusPorPref}) \wedge \text{def?}(t, s.\text{CompusPorPref}) \wedge s \neq t \Rightarrow_L$   
 $\text{obtener}(s, s.\text{CompusPorPref}) \cap \text{obtener}(t, s.\text{CompusPorPref}) = \emptyset$
  9.  $\forall s : \text{String} \text{ def?}(s, s.\text{CompusPorPref}) \Rightarrow_L \pi_1(\text{obtener}(s, s.\text{CompusPorPref})) =$   
 $\pi_2(\text{obtener}(s, s.\text{CompusPorPref}))$
  10.  $\text{Longitud}(s.\text{compus}) = \text{Longitud}(\text{CaminosMinimos}(s)) \wedge$   
 $(\forall i : \text{nat}) 0 \leq i < \text{Longitud}(s.\text{compus}) \Rightarrow_L$   
 $\text{Longitud}(s.\text{CaminosMinimos}[i]) = \text{Longitud}(s.\text{compus})$
  11.  $(\forall f, d : \text{nat}) \neg(f = d) \wedge 0 \leq f, d < \text{Longitud}(s.\text{compus}) \Rightarrow_L$   
 $\text{CaminosMinimos}[f][d] =$   
 $\text{caminoMinimo}(s.\text{red}, \text{ipACompu}(s.\text{red}, \pi_1(s.\text{compus}[f])), \text{ipACompu}(s.\text{red}, \pi_1(s.\text{compus}[d])))$
  - 12, 13, 14.  $(\forall i, j : \text{nat}), 0 \leq i, j < \text{longitud}(s.\text{CaminosMinimos}) \Rightarrow_L \text{longitud}(s.\text{CaminosMinimos}) =$   
 $\text{longitud}(s.\text{CaminosMinimos}[i]) \wedge \text{longitud}(s.\text{CaminosMinimos}[i][j]) < \text{longitud}(s.\text{CaminosMinimos}) \wedge$   
 $(\forall e : \text{nat}), \text{esta?}(e, s.\text{CaminosMinimos}[i][j]) \Rightarrow \text{pertenece}(e, s.\text{CompusPorPref})$
  15.  $\forall c : \text{compu} \text{ esta?}(c, s.\text{Compus}) \Rightarrow_L \pi_3(c) \leq \pi_3(s.\text{Compus}[s.\text{LaQMasEnvio}])$

### Función de abstracción

$\text{Abs} : \widehat{\text{dcnet}} s \longrightarrow \widehat{\text{DCNet}} \quad \{\text{Rep}(s)\}$

$(\forall s : \widehat{\text{dcnet}})$   
 $\text{Abs}(s) \equiv dc : \widehat{\text{DCNet}} \mid$   
 $\text{red}(dc) =^*(s.\text{red}) \wedge (\forall c : \text{compu}, c \in \text{compus}(dc))(\text{enEspera}(dc, c) =^*(\text{enEspera}(s, c)) \wedge$   
 $\text{cantidadEnviados}(dc, c) = \text{cantidadEnviados}(s, c)) \wedge$   
 $(\forall p : \text{paquete}, \text{paqueteEnTransito?}(dc, p)) \text{caminoRecorrido}(dc, p) =^*(\text{caminoRecorrido}(s, p))$

## 2.3 Algoritmos

<b>ICREAR SISTEMA</b> ( <b>in</b> $r : \text{red}$ ) $\longrightarrow res : \text{dcnet}$	
$res.red \leftarrow r$	
$n \leftarrow \text{Longitud}(\text{COMPUS}(red))$	$O(1)$
$i \leftarrow 0$	
$j \leftarrow 0$	$O(1)$
$res.Compbus \leftarrow \text{CREARARREGLO}(n)$	$O(n)$
$res.CaminosMinimos \leftarrow \text{CREARARREGLO}(n)$	$O(n)$
<b>var</b> $p : \text{arreglo\_dimensionable de puntero}(\text{conjLog}(\text{paquete}))$	
<b>while</b> $i < n$ <b>do</b>	$O(L * n^5)$
	$O(n)$
$res.CaminosMinimos[i] \leftarrow \text{CREARARREGLO}(n)$	$O(n)$
$s : < nat, conjLog(paquete, <_{id}), conjLog(paquete, <_p),$	
$conjLog(paquetePos, <_{id}), conjLog(paquetePos, <_p) >$	
$\pi_1(s) \leftarrow compu(r, i)$	
$\pi_2(s) \leftarrow \text{NUEVO}()$	
$\pi_3(s) \leftarrow \text{NUEVO}()$	
$\pi_4(s) \leftarrow \text{NUEVO}()$	
$\pi_5(s) \leftarrow \text{NUEVO}()$	
$\text{DEFINIR}(res.CompbusPorPref, compu(r, i), s)$	$O(L)$
$p[i] \leftarrow \pi_3(s)$	
$p'[i] \leftarrow \pi_5(s)$	
$res.Compbus[i] \leftarrow < compu(r, i), p[i], p'[i], 0 >$	$O(1)$
<b>while</b> $j < n$ <b>do</b>	$O(L * n^4)$
	$O(n)$
$res.CaminosMinimos[i][j] \leftarrow \text{caminoMinimo}(compu(r, i), compu(r, j), r)$	$O(L * n^3)$
$j++$	
<b>end while</b>	
$i++$	
<b>end while</b>	
$res.LaQMasEnvio \leftarrow 0$	$O(1)$
<hr/>	
	$O(L \times n^5)$
<b>ICREARPAQUETE</b> ( <b>in/out</b> $s : \text{dcnet}$ , <b>in/out</b> $p : \text{paquete}$ )	
$t_1 : < nat, conjLog(paquete, <_{id}), conjLog(paquete, <_p),$	
$conjLog(paquetePos, <_{id}), conjLog(paquetePos, <_p) >$	
$t_1 \leftarrow \text{OBTENER}(\text{origen}(p), s.CompbusPorPref)$	$O(L)$
$t_2 : < nat, conjLog(paquete, <_{id}), conjLog(paquete, <_p),$	
$conjLog(paquetePos, <_{id}), conjLog(paquetePos, <_p) >$	
$t_2 \leftarrow \text{OBTENER}(\text{destino}(p), s.CompbusPorPref)$	$O(L)$
$p' : paquetePos$	
$\text{INDICEORIGEN}(p') \leftarrow \pi_1(t_1)$	$O(1)$
$\text{INDICEDESTINO}(p') \leftarrow \pi_1(t_2)$	$O(1)$
$\text{POSACTUAL}(p') \leftarrow 0$	
$\text{INSERTAR}(\pi_2(t), p)$	$O(\log(k))$
$\text{INSERTAR}(\pi_3(t), p)$	$O(\log(k))$
$\text{INSERTAR}(\pi_4(t), p')$	$O(\log(k))$
$\text{INSERTAR}(\pi_5(t), p')$	$O(\log(k))$
<hr/>	
	$O(L + \log(k))$

## 2.4 Servicios Usados

Del modulo ConjLog requerimos pertenece, buscar, menor, insertar y borrar en  $O(\log(k))$  .

Del modulo Diccionario Por Prefijos requerimos Def?, obtener en  $O(L)$ .