UN ALGORITMO DISTRIBUIDO DE DETECCION DEL INTERBLOQUEO PARA EL MODELO UNICO-RECURSO

Alberto Córdoba,
Informática Ercilla
Rodriguez Arias, 23
48011 Bilbao, España

José R. González de Mendívil,
Dept. Electricidad y Electrónica
Universidad del País Vasco
48080 Bilbao, España

José R. Garitagoitia
Dept. Informática
Universidad del País Vasco
48080 Bilbao, España

RESUMEN: En este artículo se presenta un algoritmo distribuido de detección del interbloqueo para un Sistema Distribuido de Base de Datos con modelo de ocupación único-recurso. El algoritmo de detección propuesto sigue la filosofía de los algoritmos arco a cazar y presenta las siguientes ventajas: (a) Solo utiliza información local del Grafo de Esperas; (b) todos los mensajes utilizados en el proceso de detección tienen longitud constante y mínima información; (c) el algoritmo se demuestra formalmente correcto y no genera falsos interbloqueos; (d) por cada interbloqueo sólo un único agente de transacción es responsable de la resolución; (e) el número de mensajes intercambiados para detectar el interbloqueo es menor que el de otros algoritmos de características similares.

PALABRAS-CLAVE: Sistemas Distribuidos; Sistemas de Bases de Datos; Modelo de Ocupación Unico-Recurso; Detección del Interbloqueo; Detección Distribuida.

I.- INTRODUCCION

El *Interbloqueo* (*Deadlock*) es un problema muy importante en los Sistemas Distribuidos de Bases de Datos. Las tres técnicas básicas para abordar dicho problema son [1]: (i) *Prevención*; (ii) *Exclusión*; (iii) *Detección* y *Resolución*. Reconociendo que en los Sistemas Distribuidos de Bases de Datos, las técnicas de Prevención y Exclusión no pueden ser aplicadas [2] [3], centraremos el problema en las técnicas de detección.

Los mecanismos de control de concurrencia usuales, garantizan la consistencia de las operaciones sobre los recursos de la Base de Datos, pero fuerzan a las transacciones a esperar si los recursos que solicitan no se encuentran disponibles [4]. Si estas esperas no son controladas se puede producir una situación en la que un conjunto de transacciones esperan indefinidamente por los recursos asignados a transacciones en el mismo conjunto. Esta situación no deseable es la que se denomina *situación de interbloqueo* [5]. Las técnicas de detección deben realizar dos importantes tareas [3]: (1) mantener actualizada la información de las interacciones entre las transacciones; y (2) buscar sobre dicha información grupos de transacciones en espera circular. Cuando una situación de interbloqueo es detectada se requiere un *mecanismo de resolución* que permita a ciertas transacciones en el interbloqueo continuar su ejecución [1] [6]. Los algoritmos de detección para Sistemas Distribuidos se clasifican según la forma en que se realicen las tareas (1) y (2) del proceso de detección [2] [3]: (i)*Centralizados*. Un único lugar del sistema se encarga de mantener actualizada la información y de realizar las búsquedas de transacciones en espera circular. (ii) *Jerárquicos*. Los lugares se encuentran organizados de forma jerárquica y se detecta el interbloqueo en los lugares inferiores. (iii) *Distribuidos*. Todos los lugares del sistema participan de la misma manera en el proceso de detección. Las ventajas e inconvenientes de estos tipos de algoritmos se pueden estudiar en [2]

[3] [7]. Además en el caso de las Bases de Datos Distribuidas, la complejidad de los algoritmos de detección depende del tipo de modelo de ocupación de recursos que la Base de Datos permite [2].

En este artículo presentamos un algoritmo distribuido de detección y resolución del interbloqueo para un Sistema Distribuido de Base de Datos con un modelo de ocupación único-recurso. El modelo de la Base de Datos sobre el cual se formalizan las acciones del algoritmo es un modelo basado en Grafos debido a Menasce y Muntz [8]. El algoritmo es del tipo *arco a cazar* (*edge-chasing*) y está basado en la idea original de Mitchell y Merritt de asociar a cada agente de transacción dos etiquetas [9]. Mediante la introducción de un nuevo mecanismo que permite distinguir la formación secuencial del interbloqueo de la formación simultánea se reduce el número de mensajes necesarios para la detección. Este coste es variable e igual a n(n+2)/4 en un peor caso de formación del interbloqueo y n en el mejor caso (n- número de lugares en el interbloqueo).

El resto del artículo se organiza de la siguiente forma: En la Sección II se muestra el modelo de Base de Datos utilizado y se caracteriza la situación de interbloqueo. En la Sección III se introduce el algoritmo de detección propuesto. En la Sección IV se muestra un ejemplo de funcionamiento del algoritmo, así como el costo en número de mensajes necesarios para la detección. Por último, la Sección V se dedica a las conclusiones del trabajo.

II.- UN MODELO DE BASE DE DATOS

Un Sistema Distribuido de Base de Datos se puede considerar como un conjunto de lugares que se encuentran conectados a través de una red de comunicaciones intercambiando información mediante el envío y la recepción de mensajes [6]. Cada lugar tiene un único identificador, asumiendo que este identificador es un número natural, el conjunto de lugares se define como $\mathbf{L}= \{L_j / L_j \in \mathcal{N}, j:1..N\}$. La red de comunicaciones se supone perfectamente fiel: los mensajes enviados a través de la red no se pierden, se reciben en un tiempo finito y están libres de error. Cada lugar L_j se puede considerar como una Base de Datos central que gestiona una porción de la Base de Datos global. Los recursos almacenados en L_j sólo pueden ser accedidos por medio del controlador de recursos del lugar C_j.

Los usuarios interaccionan con los recursos de la Base de Datos por medio de la ejecución de transacciones [6]. Una transacción se puede considerar, de forma simplificada, como un conjunto de operaciones de *lectura* y *escritura* sobre los recursos de la Base de Datos. Cada transacción tiene un único identificador que la distingue del resto de transacciones en el sistema. Asumiendo que este identificador es un número natural, el conjunto de transacciones en el sistema se define como $\mathbf{T}= \{T_i / T_i \in \mathcal{N}, i:1..M\}$. Una transacción T_i es local en L_j si actúa sólo sobre los recursos gestionados por C_j. Una transacción es distribuida si solicita recursos a distintos controladores. Para realizar las operaciones sobre los distintos controladores, cada transacción sea local o distribuida tiene un único representante en cada lugar denominado *agente de transacción* [6] [8]. Un agente de transacción, denotado de la forma $a_{ij}= (T_i, L_j)$ es un proceso que actúa en cada lugar en nombre de la transacción.

Para representar el estado de ejecución de todos los agentes de transacción se utiliza el modelo denominado Grafo de Espera entre Transacciones [8]. Este grafo, denotado **G**, es un

grafo dirigido y dinámico que está formado por una pareja (X, U), donde X es el conjunto de nodos y U es el conjunto de arcos. El conjunto de nodos X está formado por todos los agentes de transacción, de forma que X= TxL. El conjunto de arcos verifica en todo instante U⊂ XxX. Sólo son posibles dos tipos de arcos en U: (i) *Arcos internos*. Son arcos de la forma u= (a_{ij}, a_{kj}), establecidos en un mismo lugar L_j entre dos agentes de transacciones distintas. La existencia de un arco de este tipo indica que el agente a_{ij} espera por algún recurso asignado al agente a_{kj}. (ii) *Arcos externos*. Son arcos de la forma u= (a_{ij}, a_{ik}), establecidos entre los agentes de una misma transacción en lugares diferentes. En un arco externo u= (a_{ij}, a_{ik}), el agente a_{ij} se denomina *saliente* del lugar L_j y el agente a_{ik} se denomina *entrante* en el lugar L_k.

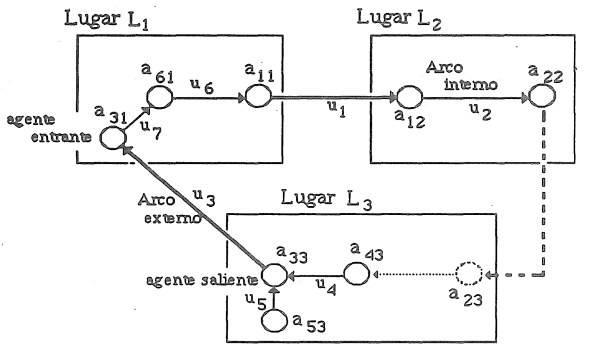

Figura 1Grafo de Espera entre Transacciones. Si se crea el arco externo (a22, a23) y el agente a 23 se queda a la espera del agente a 43 entonces se produce un circuito en el grafo y por lo tanto una situación de intrebloqueo que envuelve a los tres lugares.

En el modelo de ocupación único-recurso, un agente sólo puede adquirir un único recurso cada vez. Por lo tanto, un agente sólo puede esperar por otro agente. Esto queda caracterizado sobre el grafo G por el hecho de que para cada nodo a∈ X se verifica estrictamente la relación $0 \leq d^+(a) \leq 1$, donde $d^+(.)$ representa el semigrado positivo de un nodo [10].

La condición necesaria y suficiente para que exista interbloqueo en el sistema es que exista un circuito de nodos en el grafo G [8] (**figura 1**). Un algoritmo de detección se encarga básicamente de encontrar circuitos sobre el grafo.

III.- EL ALGORITMO DE DETECCION

El algoritmo de detección propuesto se clasifica en la categoría de los algoritmos arco a cazar [2]. Estos algoritmos basan su funcionamiento en el envío de mensajes especiales de detección denominados *sondas*, a través de los canales representados por los arcos externos en el grafo. No existe ningún lugar en el que se centraliza la información del grafo. Todos los lugares mantienen actualizada la información de aquella parte del grafo correspondiente únicamente al lugar. Cada lugar tiene una copia del algoritmo de detección y todos los lugares cooperan equitativamente en el proceso de detección enviando las correspondientes sondas en la dirección contraria a la dirección de los arcos en el grafo. Se supone que las sondas de un lugar se reciben en el mismo orden en que fueron enviadas, en un tiempo finito y libres de error. Se supone que no existen suspensiones espontáneas de los agentes de transacción con objeto de evitar la detección de falsos interbloqueos [3].

Cada nodo establecido en un lugar tiene asignados los siguientes elementos representativos: (i) Identificador privado (*identificador*). Es aquel número natural que se corresponde con el identificador de la transacción que el nodo representa en el lugar. Este número es fijo y no

puede ser modificado. (ii) Identificador público (*valor*). Es un número entero positivo que puede modificarse por el algoritmo de detección. (iii) *Marca*. Es un valor booleano asociado al nodo que puede modificarse por el algoritmo de detección.

En todo instante de tiempo se marcan aquellos nodos que son el final de un camino abierto de esperas. Cuando un nodo marcado se convierte en un nodo saliente, su correspondiente nodo entrante en otro lugar se marcará, generando sondas especiales con marca si el nodo entrante se queda a la espera directa o transitiva de otro nodo que también es saliente. En caso contrario, el nodo entrante marcado cederá la marca a los nodos que espera, volviendo a establecer la marca en los nodos que son el final de un camino abierto de esperas. Las sondas marcadas generadas en este proceso no se pierden porque sus valores quedan almacenados en los valores de los nodos salientes. Los valores de las sondas marcadas siempre se comparan con los valores asociados a los nodos marcados. Las posibles sondas que se produzcan cuando no existe ningún nodo marcado, estarán sin marcar y sus valores, en este caso identificadores, se compararán con los identificadores de los nodos no marcados. En la **figura 2** se muestran las definiciones de las principales funciones utilizadas por el algoritmo de detección.

L= {j / j:1..N} conjunto de lugares del sistema.
T= {i / i:1..M} conjunto de identificadores.
G= (**X**, **U**) Grafo de Espera entre Transacciones
V= {v / v∈ Z, v ≥ 0} conjunto de valores

Funciones:

Identificador. id: **X** ---> **T**, ∀aij ∈ **X**, id(aij)= i.
Valor. val: **X** ---> **V**, ∀aij ∈ **X**, val(aij)= v.
Marca. marca: **X** ---> {true,false}, ∀aij ∈ **X**, marca(aij)
Sucesor local. akj es sucesor local de aij, akj suc aij, sii:
(i) ∃ u∈**U** / u=(aij, akj) ó
(ii) ∃ amj ∈ **X** y u'∈ **U** / u'=(aij, amj) y akj suc amj
Antecesor local. aij es antecesor local de akj, aij ant akj sii:
(i) ∃ u∈**U** / u=(aij, akj) ó
(ii) ∃ amj ∈ **X** y u'∈ **U** / u'=(aij, amj) y akj ant amj.
Entrante. entrante: **X** ---> {true,false}, entrante(aij)= true sii:
∃ u∈**U** / u=(aik, aij) con k≠ j.
Saliente. saliente: **X** ---> {true,false}, saliente(aij)= true sii:
∃ u∈**U** / u=(aij, aik) con j≠ k.

Figura 2. Definiciones utilizadas por el algoritmo de detección.

Las sondas que se utilizan para detectar el interbloqueo contienen la siguiente información s = (marca, id_s, receptor). El primer campo booleano indica si la sonda es una sonda marcada o no. El identificador de la sonda id_s es un valor adquirido del valor asociado a un nodo o bien un identificador de un nodo. Por último *receptor* es el nodo al cual va dirigida la sonda. Dado que una sonda se transmite a través de un arco externo $(a_{ij}, a_{ik}) \in U$ en la dirección opuesta al arco y dado que una sonda se genera por un nodo entrante a_{ik} y el receptor es el correspondiente nodo saliente a_{ij}, se puede obviar la actualización del campo receptor en la sonda en la presentación del algoritmo por criterios de simplicidad. Por otra parte, asumiremos que s* indica una sonda marcada, siendo s*= id_s; y s indica una sonda sin marcar, siendo s= id_s (id_s = identificador o valor).

En el modelo único recurso todos los circuitos que se pueden formar son disjuntos es decir no tienen arcos comunes. Por lo tanto se puede plantear un esquema de detección en cada lugar que trate sobre los circuitos locales y un mecanismo de detección independiente para circuitos implicado distintos lugares. Vamos a definir un circuito [10] implicando n lugares por las relaciones de los arcos externos que contiene o de forma equivalente por la secuencia de nodos entrante y saliente que forman parte del circuito en cada lugar.Sea $[a_{ij}, a_{kj}]$ una relación entre dos nodos $a_{ij}, a_{kj} \in X$, tal que verifica: (i) entrante(a_{ij})= true; (ii) saliente(a_{kj})= true; (iii) a_{kj} suc a_{ij}. Se define un circuito $\mathscr{C}$ implicando n lugares de la forma:

$\mathscr{C} \equiv [a_{11}, a_{21}][a_{22}, a_{32}][a_{33}, a_{43}]... [a_{ii}, a_{(i+1)i}] ... [a_{(n-1)(n-1)}, a_{n(n-1)}][a_{nn}, a_{1n}]$

donde se han utilizado índices correlativos de lugares y de identificadores de transacción, pero en este caso y para no perder generalidad los identificadores de transacción deben interpretarse en general como $T_i \in T$. Es muy sencillo demostrar que la definición anterior de circuito corresponde realmente a un circuito en el grafo G.

El proceso de detección completo realiza dos tareas: (1) actualización del grafo G y (2) búsqueda de circuitos implicando varios lugares. La tarea (1) también es en parte gestionada por los mecanismos de control de la transacciones en lo que respecta a la creación y desaparición de agentes de transacción (nodos del grafo). El algoritmo de detección completo comprende tres pasos denominados: (i) Paso de actualización de valores y marcas (**figura 3**); (ii) Paso de generación de sondas y (iii) Paso de emisión de sondas y detección (**figura 4**).

A1: $\exists$ u= (aij, akj) en un instante t sii el nodo aij espera por akj por algún recurso.
A2: $\exists$ u= (aij, aik) en un instante t sii el nodo aij espera por algún mensaje de su agente aik en el lugar k.
A3: En el instante en que aparece el arco externo u= (aij, aik) saliente(aij)= true, val(aij)= 0, entrante(aik)= true, val(aik)=id(aik).
A4: Si un nodo aij cumple entrante(aij)= false y d+(aij)=0 entonces val(aij)=0.
A5: Sea un nodo aij tal que saliente(aij)=true y $\forall$ akj / akj ant aij se cumple entrante(akj)=false, entonces si aij recibe s* y se cumple s* > val(aij) el nuevo valor de aij es val(aij)= s*.
A6: Si en un instante se cumple entrante(aij)= true, marca(aij)= false y se crea un arco por **A1** entonces $\forall$ akj / akj suc aij:
if d+(akj)= 0 then marca(akj)= true;
if saliente(akj)= true then para u=(akj, akp)
    if d+(akp)= 0 then marca(akp)= true.
A7: Sea akj / entrante(akj)= false, saliente(akj)= false, marca(akj)= true por **A6** entonces $\forall$ amj / amj suc akj:
if d+(amj)= 0 then marca(amj)= true, marca(akj)= false;
if d+(amj)= 1 and saliente(amj)= true then $\forall$ aij / entrante(aij)= true y verifica aij ant akj: marca(aij)= true, marca(akj)= false.
A8: Si en un instante se cumple para un nodo entrante(aij)= true y marca(aij)= true entonces:
$\forall$ akj / akj suc aij: if d+(akj)= 0 then marca(akj)=true, marca(aij)=false.
A9: Si para un nodo se cumple marca(aij)=true por **A5**, **A6** o **A7** y por **A2** se crea un arco externo u=(aij, aik) entonces marca(aij)= false, mrca(aik)= true.

Figura 3. Paso de Actualización de valores y marcas

Las reglas de la **figura 3** se utilizan para distinguir la formación secuencial de la formación simultanea de parejas de relaciones $[a_{ij}, a_{kj}]$, además por las reglas A6, A7, A8 y A9 cuando se crea una relación $[a_{ij}, a_{kj}]$ entonces sólo puede verificarse que marca(a_{ij})= true o false y que siempre marca(a_{kj})= false. La regla A5 asegura que los valores de las sondas marcadas s* nunca se pierden y son almacenados en los valores de los nodos salientes. El paso de generación de sondas y de emisión de sondas (**figura 4**) sólo funciona a través de las relaciones $[a_{ij}, a_{kj}]$ establecidas en un lugar. Ciertas actualizaciones de valores distintas a A3, A4 y A5 se realizan en los pasos de generación y de emisión de sondas. Una especificación formal del algoritmo completo basado en la Teoría de Autómatas se encuentra en [11] y bajo esta formulación se demuestra la prueba formal de corrección. El algoritmo asegura que por cada circuito sólo un nodo detecta el interbloqueo y por lo tanto dicho nodo puede ser víctima con objeto de resolver de manera eficaz el interbloqueo.

IV.- FUNCIONAMIENTO DEL ALGORITMO

En esta Sección mostramos a través de un ejemplo el funcionamiento del algoritmo de detección propuesto. Así mismo, se presentan los costos en número de mensajes necesarios para la detección. Supongamos que en cada lugar L_1, L_2, L_3 y L_4 (**figura 5**) existe inicialmente un agente activo, denominados a_{11}, a_{22}, a_{33}, a_{44} respectivamente, con id(a_{11})=1, id(a_{22})=2, id(a_{33})=3, id(a_{44})=4. Supondremos el instante t_i anterior al t_{i+1}.

En el instante t1 se crea el arco externo (a_{11},a_{12}) por A2, y el arco interno (a_{12},a_{22}) por A1. Por A3 val(a_{11})= 0 y val(a_{12})= id(a_{12})= 1, y por A6 marca(a_{22})= true.

Paso de Generación de sondas: Se activa en un lugar en un instante de tiempo t cuando aparece una relación [aij, akj]:

Paso 1.1. Generación de sondas marcadas

```
if marca(aij) then
  if val(aij)> val(akj) then
    begin s*:= val(aij); emitir(s*) end
  else
    begin val(aij):= val(akj); s*:= val(aij); emitir(s*) end
else
  if val(akj)>0 then begin s*:= val(akj); emitir(s*) end;
```

Paso 1.2. Generación de sondas sin marcar

```
if not marca(aij) and v(akj)= 0 then
  if id(aij)> id(akj) then
    begin s:= id(aij); emitir(s) end;
```

Paso de emisión de sondas y detección: Se activa en un lugar en un instante de tiempo t cuando para una relación existente [aij, akj] el nodo akj es receptor de una sonda:

Paso 2.1. Sondas marcadas (se recibe s*)

```
if marca(aij) then
  begin
    if s*= val(aij) then victima(aij); { Interbloqueo}
    if s*> val(aij) then
      begin val(aij):= s*; val(akj):= s*; emitir(s*) end
  end
else
  if s*> val(akj) then
    begin val(akj):= s*; emitir(s*) end;
```

Paso 2.2. Sondas sin marcar (se recibe s)

```
if not marca(aij) and val(akj)= 0 then
  begin
    if s= id(aij) then victima(aij);{ Interbloqueo}
    if s> id(aij) then emitir(s)
  end;
```

Figura 4. Paso de Generación de sondas, Paso de Emisión de sondas y detección .

En el instante t2 se crean simultaneamente los arcos externos (a_{22},a_{23}), (a_{33},a_{34}), (a_{44},a_{41}) por A2, y posteriormente los arcos internos (a_{23},a_{33}), (a_{34},a_{44}), (a_{41},a_{11}). Por A3 val(a_{22})= 0, val(a_{23})= id(a_{23})= 2, val(a_{33})= 0, val(a_{34})= id(a_{34})= 3, val(a_{44})= 0, val(a_{41})= id(a_{41})= 4. Además por A9, marca(a_{22})= false y marca(a_{23})= true. Esta situación de interbloqueo es la reflejada en la **figura 5**.

En cada lugar aparece una relación [entrante,saliente] que activa el paso de generación de sondas. En L1 se genera una sonda s1 por el Paso 1.2, tal que s1= id(a_{41})= 4, que se envía a L4. En L2 por el Paso 1.2 no se genera sonda. En L3 se genera una sonda s3* por el Paso 1.1, tal que s3*= id(a_{23})= 2, que se envia a L2. En L4 por el Paso 1.2 no se genera sonda.

La sonda s1 emitida por L1 al llegar a L4 activa en ese lugar el Paso 2.2 de emisión y detección de sondas. Por este Paso 2.2 se genera una sonda s'1= s1 que se emite hacia L3. cuando s'1 llega a L3 activa el Paso 2.2 y se pierde. La sonda s3* al llegar a L2 activa el Paso 2.1 y genera una sonda s'3* hacia L1. De la misma forma s'3* genera s"3* en L1 hacia L4 y la s"3* genera la s"'3* en L4 hacia L3. Finalmente al llegar s"'3* a L3 por el Paso 2.1 se detecta el intebloqueo, siendo elegido el agente a_{23} como víctima. Las sondas emitidas son s1, s'1, s3*, s'3*, s"3*, s"'3*, en total 6 sondas.

La evaluación de la complejidad de un algoritmo de detección distribuido viene determinada por los siguientes criterios [3]: (a) número de mensajes intercambiados para detectar la situación de interbloqueo; (b) longitud de los mensajes; (c) retardo promedio en la detección del interbloqueo.En nuestro algoritmo se pueden distinguir varios casos de formación del circuito. En la formación simultánea de los arcos del circuito, el costo puede variar entre 2n-1, en el mejor caso, y n(n+2)/4, en el peor (supuesto n número par de lugares en interbloqueo).

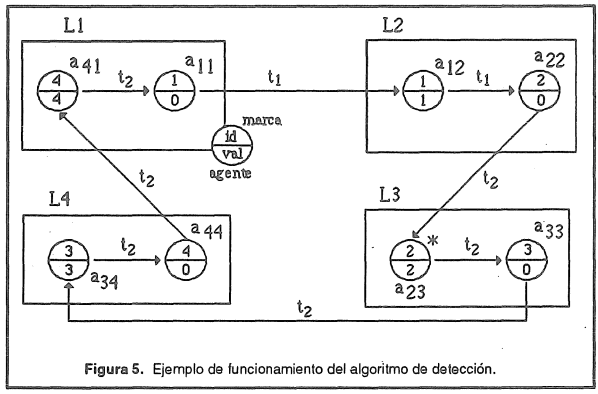

Figura 5. Ejemplo de funcionamiento del algoritmo de detección.

En el caso de la formación secuencial de los arcos del circuito, el costo puede variar entre n, en el mejor caso, y n(n+2)/4, en el peor (supuesto n número par de lugares en el interbloqueo).

El ejemplo corresponde a un caso compuesto (secuencial y simultáneo) de formación del circuito. Son necesarias n= 4 sondas marcadas para detectar el circuito (corresponde al mejor caso secuencial), perdiendose dos sondas no marcadas (corresponde y está acotado por el peor caso simultaneo n(n+2)/4= 4).

Una comparación del algoritmo de detección propuesto con otros algoritmos presentes en la literatura se muestra en la siguiente **tabla 1**.

Algoritmo	Nº de mensajes	Retardo	Tamaño de mens.
Goldman [15]	<m n	t+ nT	variable (medio)
Isloor-Marsland [5]	r (N-1)	0	constante (bajo)
Menasce-Muntz [8]	m (n-1)	nT	variable (bajo)
Obermarck [14]	m (n-1)/2	nT	variable (bajo)
Chandy [12]	<m n	t+ nT	constante (bajo)
Sinha-Natarajan [13]	mejor caso 2 (n-1)	2 (n-1) T	constante (bajo)
	peor caso m (n-1)		
Mittchell-Merritt [9]	m (n-1)/2	(n-1)T	constante (bajo)
Nuestro algoritmo	mejor caso n		
	peor caso n(n+2)/4	nT	constante (bajo)

N: número de lugares; n: número de lugares en el interbloqueo
m: número de procesos en el interbloqueo; T: retardo promedio de comunicación entre lugares; t: retardo en la inicialización del proceso de detección; r: velocidad de actualización del Grafo de estado; d: diámetro del Grafo de estado.

Tabla 1. Comparación de algoritmos de detección distribuidos.

V.- CONCLUSIONES

En este artículo se presenta un algoritmo distribuido de detección del interbloqueo para un Sistema Distribuido de Base de Datos con modelo de ocupación único-recurso. El modelo de la Base de Datos utilizado, basado en un Grafo de Esperas entre Transacciones, es lo suficientemente general como para eludir las descripciones particulares de los mecanismos de control de concurrencia de las transacciones. El algoritmo de detección propuesto sigue la filosofía de los algoritmos arco a cazar y parte de la idea original de Mitchell y Merritt de asociar a cada nodo dos etiquetas una privada y otra pública. El algoritmo presenta las siguientes ventajas: (a) Solo utiliza información local del Grafo de Esperas; (b) todos los

mensajes utilizados en el proceso de detección tienen longitud constante y mínima información; (c) el algoritmo se demuestra formalmente correcto y no genera falsos interbloqueos; (d) por cada interbloqueo sólo un único agente de transacción es responsable de la resolución; (e) el número de mensajes intercambiados para detectar el interbloqueo es menor que el de otros algoritmos de características similares.

REFERENCIAS

[1] E. G. Coffman, M. J. Elphick, A. Shoshani, "Systems Deadlocks", ACM Computing Surveys, vol. 3, no. 2, pp. 67-78, Jun. 1971.

[2] E. Knapp, "Deadlock Detection in Distributed Databases", ACM Computing Surveys, vol. 19, no. 4, pp. 303-328, Dec. 1987.

[3] M. Singhal, "Deadlock Detection in Distributed Systems", IEEE Computer, pp. 37-48, Nov. 1989.

[4] K. P. Eswaran, J. N. Gray, R. A. Loire, I. L. Traiger, "The Notions of Consistency and Predicate Locks in a Database System", Commun. ACM, vol. 19, no. 11, pp. 623-633, Nov. 1976.

[5] S. S. Isloor, T. A. Marsland, "An Effective On-Line Deadlock Detection Technique for Distributed Database Management Systems", Proceedings Computer Science 78, pp. 283-288, Nov. 1978.

[6] P. A. Bernstein, V. Hadzilacos, N. Goodman, "Concurrency Control and Recovery in Database Systems", Adison-Wesley, Reading, Mass. 1987.

[7] V. D. Gligor, S. H. Shattuck, "On Deadlock Detection in Distributed Systems", IEEE Trans. on Softw. Engineering, vol. SE-6, no. 5, pp. 435-440, Sep. 1980.

[8] D.A. Menasce, R.R. Muntz, "Locking and Deadlock Detection in Distributed Databases", IEEE Trans. on Softw. Engineering, vol. SE-5, no. 3, pp. 195-202, May. 1979.

[9] D. P. Mitchell, M. J. Merritt, "A Distributed Algorithm for Deadlock Detection and Resolution", Proc. ACM Conf. Principles of Distributed Computing, pp. 282-284, Aug. 1984.

[10] C. Berge, "Graphs and Hypergraphs", North-Holland Publishing Company, Amsterdam, 1973.

[11] A. Córdoba, "Algoritmos para la Detección del Interbloqueo en Bases de Datos Distribuidas", Tesis Doctoral, Universidad del País Vasco, Leioa 1990.

[12] K. M. Chandy, J. Misra, "A Distributed Algorithm for Detecting Resource Deadlocks in Distributed Systems", Proceedings of the ACM Symposium on Principles of Distributed Computing, pp. 157-164, August 1982.

[13] M. K. Sinha, N. Natarajan , "A Priority-Based Distributed Deadlock Detection Algorithm", IEEE Trans. on Softw. Engineering, pp. 67-80, Jan. 1985.

[14] R. Obermark, "Distributed Deadlock Detection Algorithm", ACM Trans. on Database Systems, vol. 7, no. 2, pp. 187-208, Jun. 1982.

[15] B. Goldman, "Deadlock Detection in Computers Networks", MIT/LCS/TR-185, Laboratory of Computer Science, MIT, Cambridge, Mass., Sep. 1977.