ISSN 1409-2441

Ingeniería

Revista de la Universidad de Costa Rica

Enero/Diciembre 1998 VOLUMEN 8 Nos. 1 y 2

Ingeniería 8(1,2): 21-40 1998 San José, Costa Rica

COMPORTAMIENTO DEL EVALUADOR DE PARTICIONES EN BASES DE DATOS DISTRIBUIDAS ORIENTADAS A OBJETOS

Elzbieta Malinowski G.[1]

Resumen

Las bases de datos distribuidas orientadas a objetos presentan dificultades en el proceso de fragmentación. No existen criterios claros acerca de como evaluar la "bondad" de una fragmentación con respecto a otra. El evaluador de particiones para las bases de datos orientadas a objetos es la generalización del evaluador de particiones desarrollado para las bases de datos relacionales y permite medir esta "bondad". Además, demuestra que es posible la generalización de los algoritmos desarrollados para las bases de datos relacionales y su aplicación a las bases de datos orientadas a objetos. Los experimentos presentes en este artículo demuestran estos aspectos. También, se presentan algunos casos y se justifica la posibilidad de excluir los atributos complejos del proceso de fragmentación.

Summary

Object-oriented databases show difficulties in the process of fragmentation. There are not clear criterias how to evaluate "goodness" of the fragmentation scheme. Partition evaluator for object-oriented databases is the generalization of partition evaluator developed for relational databases and it allows to measure this "goodness". Moreover, it shows that it is possible the generalization of the algorithms developed for relational databases and their implementation in object-oriented databases. The experiments presented in this article show these aspects. Besides, we present the cases and justification of the exclusion of the complex attributes from the fragmentation scheme.

1. INTRODUCCION

La amplia aceptación del enfoque relacional en el procesamiento de datos y el continuo mejoramiento de los sistemas de gestión de bases de datos (DBMS, *Database Management System*) aumentó el interés de usar las base de datos (BD) en las aplicaciones no convencionales, como lo son CAD (*computer-aided design*, diseño asistido por computadora), sistemas de información geográfica, BD de imágenes, BD gráficas y otros. Sin embargo, estos sistemas manejan estructuras de datos muy complejas para almacenar los datos no tradicionales que son difíciles de representar en el modelo relacional [14]. En consecuencia, otro enfoque de orientación a objetos ampliamente usado en los lenguajes de programación comienza a expandirse en las BD. La posibilidad de usar las clases propias definidas por los usuarios y de presentar al usuario los objetos complejos de la misma forma como se presenta los objetos simples motiva a intensificar los estudios relacionados con el desarrollo de las bases de datos orientadas a objetos (BDOO).

Además, la amplia aceptación de redes de las computadoras y los avances en la tecnología de la comunicación cambiaron la dirección del desarrollo de las bases de datos relacionales (BDR) hacia las bases de datos distribuidas relacionales (BDDR). Estas BDDR reflejan en forma más natural la estructura de muchas organizaciones y abren la posibilidad de compartir los datos en forma eficiente. Es de esperar que las BDOO también se dirigen hacia el ambiente de redes abriendo así el nuevo campo de investigación para la implementación de las bases de datos distribuidas orientadas a objetos.

[1] M.Sc. Esc. de Ciencias de la Computación e Informática, Univ. de Costa Rica

Las BDDR fueron investigadas por muchos años. De acuerdo a Karlapalem et al. [9] son dos los factores más importantes para la ejecución de las aplicaciones en los sistemas de las bases de datos distribuidas: las operaciones de entrada/salida y la transferencia de los datos. El desempeño del sistema de la base de datos puede ser mejorado si se aplica un diseño distribuido adecuado, incluyendo la fragmentación, replicación y localización, porque estos elementos influyen en los dos factores mencionados anteriormente, en especial la fragmentación.

El enfoque relacional distingue dos tipos de fragmentación: la horizontal y la vertical. También se puede considerar la fragmentación híbrida que está formada por los dos tipos de fragmentación mencionados anteriormente. Cada una de estas fragmentaciones tiene sus propios criterios para determinar el mejor esquema de partición para el sistema relacional a diseñarse [1, 3, 4, 7, 12, 13].

Aunque se han hecho un número considerable de investigaciones sobre las BDOO, muy pocas se relacionan con la distribución de los objetos y, específicamente, con la fragmentación de éstos [6, 8, 10]. Ezeife y Barker [5] especifican los beneficios de la fragmentación en las BDR y mencionan que éstos deberían también ser reconocidos en las BDOO. Sin embargo, existe un aspecto muy importante que hace difícil el desarrollo de los algoritmos: para el enfoque relacional la granularidad de la fragmentación era fácil de identificar (atributos para la fragmentación vertical, tuplas/instancias para la fragmentación horizontal). En contraste, en BDOO se pueden considerar diferentes criterios, como estructura jerárquica, "afinidad" de las clases, "afinidad" de los objetos, la frecuencia del uso de los objetos complejos y otros.

Aunque existe una diferencia significativa entre las BDR y BDOO, debe ser claro que el enfoque relacional puede ser visto (ignorando - por la simplicidad - la existencia de los métodos) como un caso especial de los sistemas orientados a objetos sin la jerarquía de clases y atributos complejos (atributos que contienen la referencia a otros objetos). Si este es el caso, los algoritmos desarrollados para BDDR con respecto a la distribución de los datos podrían ser generalizados y aplicados para BDOO. Uno de los posibles ejemplos de esta generalización es el evaluador de particiones (PE, *partition evaluator*) desarrollado para la BDR [2] y generalizado para la BDOO (PEOO, *partition eveluator for object-oriented databases*, evaluador de particiones para las bases de datos orientadas a objetos) [11].

Uno de los componentes más importantes usados en el desarrollo de los algoritmos de fragmentación presentados para BDDR (también en el desarrollo de PE) es la matriz del uso de los atributos (AUM, *attribute usage matrix*) o la matriz de la afinidad de los atributos (AAM, *attribute afinity matrix*). Estas matrices usadas para el enfoque relacional no podrían ser adaptadas en forma automática al enfoque orientado a objetos. De acuerdo a Malinowski [11] se deben usar otras matrices que reflejen la complejidad del sistema y permitan "aplanar" la estructura jerárquica de las clases presentada en BDOO.

En este artículo se expone en forma breve la descripción de la representación de las clases y las matrices resultantes que reflejan las frecuencias de uso tanto de los atributos como de los métodos. Además, se especifica el PEOO y se describe su esperado comportamiento. Posteriormente, se presentan algunos ejemplos de clases, sus respectivas matrices y los resultados de aplicar a estas clases el PEOO.

2. ASPECTOS CONSIDERADOS PARA EL USO DE EVALUADOR DE PARTICIONES PARA BASES DE DATOS ORIENTADAS A OBJETOS.

Para una mayor claridad de los experimentos presentados en este artículo se resumen los siguientes aspectos:

- Presentación usada para diferentes estructuras jerárquicas de clases [5].
- Existencia de matrices necesarias para reflejar las estadísticas del uso de transacciones, métodos y atributos [11].
- Componentes de PEOO y su comportamiento[11].

Además, se consideran las siguientes suposiciones:

1. Se utiliza el enfoque basado en objetos no en valores.
2. Se tiene la información sobre las clases incluyendo los métodos y los atributos que estos métodos usan.
3. Se accesan los atributos solamente usando los métodos de acuerdo a los principios de encapsulamiento.
4. Se dispone de la información sobre la frecuencia de acceso de los métodos por las transacciones.
5. Se especifica las frecuencias de acceso de las transacciones.
6. Se ignora la existencia de los tipos tupla, conjunto y lista.

2.1 Las estructuras jerárquicas de clases

En el presente artículo se toma en cuenta la división propuesta por Ezeife y Barker [5] según la cual se distinguen cuatro casos separados de atributos simples y métodos simples, atributos simples y métodos complejos, atributos complejos y métodos simples, atributos complejos y métodos complejos.

La categoría de los atributos simples y métodos simples dispone de una estructura jerárquica muy simple, donde los métodos pueden usar los atributos de su propia clase o sus superclases y no se tienen los métodos anidados o atributos que apuntan a otros atributos

Para los atributos simples y métodos complejos la diferencia con el caso previo consiste en tener la posibilidad de usar las llamadas anidadas entre los métodos o de la misma clase o entre las clases que pertenecen a los niveles antecedentes del mismo camino del árbol de la jerarquía.

En el caso de atributos complejos y métodos simples se definen los objetos complejos como los objetos que tienen los atributos apuntando a otros objetos. En este caso, cuando el atributo complejo es accesado en realidad lo que se accesa es algún método de la clase donde este atributo apunta. Como se sigue el principio de encapsulación, no existe la situación donde el atributo complejo accesa el atributo de otra clase en forma directa.

La explicación de la categoría de atributos simples y métodos complejos se puede notar que fue incluida en los casos anteriores.

2.2 Matrices usadas en bases de datos orientadas a objetos

De acuerdo a Malinowski [11] la información sobre las frecuencias de uso que existe en BDOO se debe reflejar en las siguientes matrices:

- *Transaction-Method Usage Matrix* (TMUM, matriz de uso transacción-método): la matriz de frecuencias que indica si la transacción invoca un método específico. Los valores usados de cero o uno representan la situación que la transacción no llama o llama el método respectivamente.
- *Method-Method Usage Matrix* (MMUM, matriz de uso método-método): la matriz de frecuencias que indica el número de veces que el método invoca otro método (en el caso de métodos simples esta matriz no existe). Parecido a TMUM los valores 0 o 1 indican la existencia de llamadas anidadas de los métodos.
- *Method-Attributes Usage Matrix* (MAUM, matriz de uso método-atributo): la matriz que representa el número de invocaciones de los atributos específicos

- en una ejecución de método. Se tiene la posibilidad de tener los siguientes valores:
  - cero - indica que el método no accesa ningún atributo,
  - uno - indica que el método lee el valor del atributo (recuperar),
  - dos - indica que el método lee y escribe el valor del atributo (modificar).

La información especificada arriba puede ser dada por el diseñador del sistema.

Adicionalmente, en la misma manera como para el enfoque relacional, se puede considerar la frecuencia de uso de las transacciones por diferentes aplicaciones

Como ejemplo, se puede decir que si se multiplica la TMUM y la MAUM, se puede obtener la matriz que representa las frecuencias de uso de los atributos por las transacciones (TAUM, *trasaction attribute usage matrix*). Estas frecuencias pueden servir como base para el esquema de fragmentación y corresponden a AUM del enfoque relacional. La diferencia entre las matrices AUM y TAUM consiste que la TAUM tiene las frecuencias de acceso diferentes, contrario a la original AUM, en la cual se usan los mismos valores de las frecuencias para todos los atributos accesados por la misma transacción.

Al lector debe ser claro que dependiendo de la categoría de la jerarquía de clases algunos matrices pueden no existir.

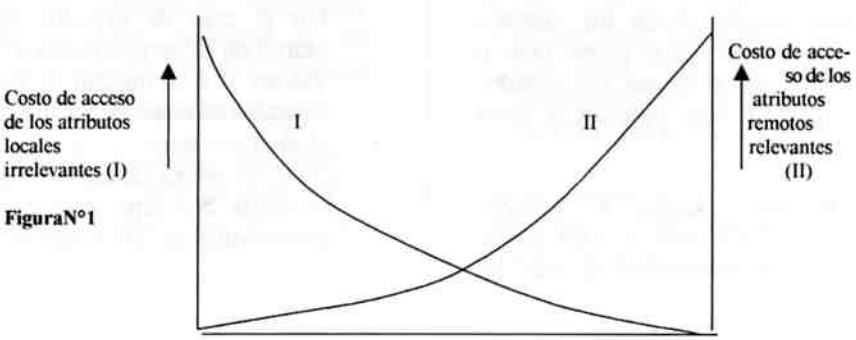
2.3 Presentación de evaluador de particiones para bases de datos orientadas a objetos.

La meta de la fragmentación de los atributos es obtener el costo mínimo de procesamiento para un conjunto dado de transacciones con respecto al acceso de sus respectivos atributos.
Es poco probable obtener un esquema de fragmentación ideal donde cada transacción accesa localmente solo a los atributos que necesita y no necesita accesar ninguno de los atributos de los lugares remotos. La función objetiva, propuesta aquí, trata de balancear el costo del acceso local y remoto para una transacción específica. De acuerdo con Chakravarthy et al. [2] el costo general en el ambiente distribuido que debemos minimizar consiste en dos elementos:

1. El primer componente, llamado costo de acceso de los atributos locales irrelevantes, representa el costo del acceso de los atributos irrelevantes cuando se accesa el objeto en el sitio local, asumiendo que todos los datos requeridos por la transacción son localmente accesibles.

2. El segundo componente, llamado el costo de acceso de los atributos remotos relevantes, incluye el costo de acceso de los atributos relevantes requeridos por la transacción desde los sitios remotos.

Chakravarthy et al. [2] demostraron el comportamiento esperado de la función objetiva usando los componentes especificados anteriormente. Este comportamiento se presenta en el gráfico de la figura N°1.



FiguraN°1

Como es evidente, el evaluador de particiones para BDOO debería tener el mismo comportamiento que se presentó antes para BDDR.

Tanto para el desarrollo de PEOO como para la realización de las pruebas se consideró las siguientes su¡x>siciones:

- Se asume que no existen réplicas, aunque es ¡x>sible considerar la replicación de los datos.
- Se asume que después de la fase de fragmentación sigue la fase de localización de los fragmentos.
- Se considera que cada fragmento tiene los atributos no sobrepuestos. Esto indica que n atributos se particionan en M fragmentos (PI, P_2, ... P_M) con ni atributos en cada fragmento. Entonces $\sum_{j=1}^{M} n_j = n$.
- Se asume que el acceso simple al dato en el fragmento local corres¡x>nde a la unidad del costo.
- Se calcula el costo relacionado con los atributos remotos relevantes asumiendo que los fragmentos de datos necesitados por la transacción son localizados en diferentes sitios porque la localización de los datos no es conocida durante la partición.

Además, los siguientes son los parámetros propuestos ¡x>r Chakravarthy [2] y Malinowski. [lI]:

n Número total de atributos en la relación considerada para la partición.

T Número de transacciones que se considera.

M Número total de fragmentos en la partición.

ni Número de atributos en el fragmento i

n^{r}_{itk} Número total de atributos en el fragmento k accesados remotamente con respecto al fragmento i por la transacción t.

f^{i}_{tj} Frecuencia de la transacción t de acceso del atributo j en el fragmento i; note que f_{tj} tiene el valor de O o de qt para el enfoque relacional.

A_{ij} Vector de Atributos para el atributo j en el fragmento i.

S_{it} Conjunto de los atributos relevantes en el fragmento i que la transacción t accesa; es vacío si la transacción t no necesita el fragmento i.

$|S_{it}|$ Número de atributos relevantes en el fragmento i que la transacción t accesa.

R_{itk} Conjunto de los atributos relevantes en el fragmento k accesados remotamente con respecto al fragmento i por la transacción t.

$|R_{itk}|$ Número de atributos relevantes en el fragmento k accesados remotamente con respecto al fragmento i por la transacción t.

CC_{ik} Costo de comunicación emre los sitios i y k.

El componente de costo de acceso local se presenta de la siguiente forma:

$$e\sim= \sum_{j=1}^{ni} L(\sim j \; -\sim ?(Aij \; -V_j) \qquad (1)$$

$$E_L^2 = \sum_{i=1}^{M} e_i^2 \qquad (2)$$

donde:

$$1\backslash, \quad "[f;, \ f;, .. fJ^{T} \tag{3}$$

$$vi=l\sim \left[\begin{array}{ccc} {}^{¡Sil\ |}_{"1} & {}^{!Si21}_{f\sim l} & {}^{¡Sil\ |}_{ni} \\ ¿.ft\sim & \frac{¿fL}{11_1} \cdots & \frac{¿fui}{FI} \end{array} \right]^{T} \tag{4}$$

El costo de acceso remoto se presenta como sigue:

$$E_R^2 = \sum_{i=1}^{T} t:_{k=1}^{M} \left[CC_{ik} * \sum_{p_k} \left(f_{tp_k}^{k} \right)^2 * \frac{|R_{itk}|}{n_{itk}^{rem}} \right] \tag{5}$$

donde:

Pk indica los atributos relevantes en el fragmento k accesados remotamente con respecto al fragmento i por la transacción t. Similarmente con la fórmula relacional PE. t:„ es el operador de llÚnimum. máximum o de promedio sobre todos fragmentos L presentando el costo de acceso optimista, pesimista o promedio respectivamente.

El evaluador de particiones para BDOO. PEOO. se presenta de la siguiente forma:

$$PEOO = E_L^2 + E_R^2 \tag{6}$$

Basándose en las fórmulas (2) y (5) Y en el análisis similar presentado por Chakravarthy et al. [2] el comportallÚento esperado de esta función es igual al presentado en la figura N° l.

3. IMPLEMENTACIÓN Y LOS RESULTADOS DE LOS EXPERIMENTOS

Para analizar el comportallÚento de PEOO se desarrollan varias pruebas. De acuerdo a la explicación previa se espera que PEOO y sus componentes de costo local y costo remoto tendrán el mismo comportamiento que el PE presentado por Chakravarthy et al. [2].

Se hacen las pruebas de los casos de atributos simples y métodos simples, atributos complejos y métodos complejos, y atributos

complejos y métodos simples. El caso de atributos simples y métodos complejos ofrece solamente una multiplicación adicional de las matrices (TMUM y MMUM). Se considera que los experimentos que representan este caso no aportan ninguna información adicional y son difíciles de seguir en el nivel teórico.

Con respecto al caso de atributos complejos y métodos simples, y atributos complejos y métodos complejos de acuerdo a la e}.lllicación presentada ¡x>r Malinowski [11] se pueden presentar como el caso de atributos simples y métodos complejos. Pues, se presentan algUnos ejemplos para demostrar esta hipótesis.

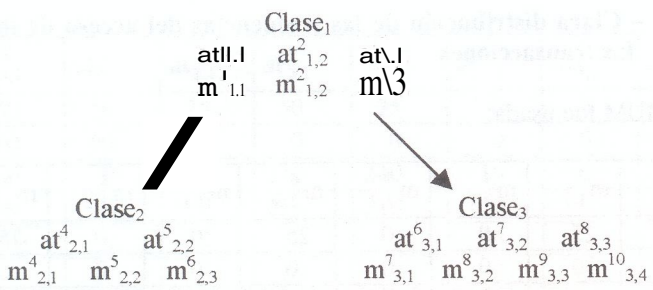
En adición, se lillÚta en los experimentos el número de clases y subclasesss para permitir al lector entendimiento del proceso de fragmentación. Además, como se usa enumeración exhaustiva de todos posibles casos de fragmentación de los atributos, se estableció el límite de ocho atributos. Con una estructura jerárquica más complicada y un esquema complejo de llamadas entre métodos sería dificil usar el análisis teórico para calcular los resultados esperados. La idea principal consiste en demostrar el comportallÚento de PEoo y comparar la mejor fragmentación obtenida en el análisis teóricoy de experimento práctico.

El programa escrito en C++ que calcula el costo local y remoto para el enfoque orientado a objetos se ejecuta para todas las posibles combinaciones (4140) de atributos variando el número de fragmentos de una a ocho. Se aplica el PEoo para cada uno de estas combinaciones y dependiendo de la selección (promedio, llÚnimum o máximum) se calcula el valor de PEoo. El resultado del programa presenta los valores óptimos para el esquema de particionallÚento.

Se asume, que si la transacción debe ser ejecutada en algún fragmento por lo menos un atributo que esta transacción necesita debe estar presente en este fragmento. Si este no es el caso, se asume que la transacción no es ejecutada en este fragmento.

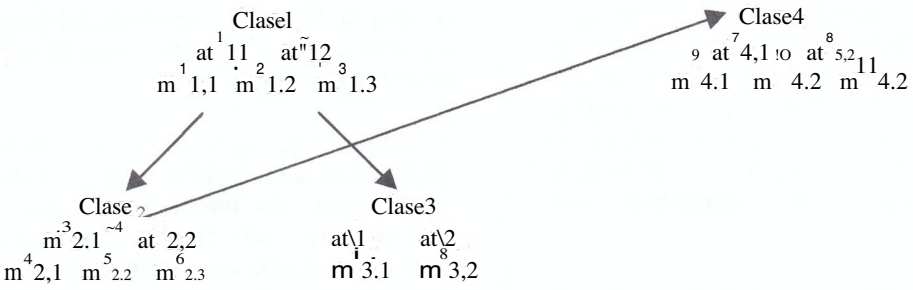
Debido a la falta de la información específica acerca de las características de la red de comunicación, en la mayoría de los experimentos se ignora el costo real de la transmisión tomándolo como valor uno por cada acceso al fragmento remoto. En estos casos se puede ver que las frecuencias de acceso son el factor dOllÚnante para la obtención de mejor esquema de fragmentación. Otros experimentos demuestran que el incremento del costo de comunicación puede cambiar considerablemente el esquema de fragmentación.

Para el caso de atributos simples y métodos simples se usa la siguiente representación jerárquica de clases:



Se asume que el acceso a los atributos se da ¡x>rmedio de los métodos siguiendo los principios de encapsulallÚento y ocultamiento de la información. Así, el método de cada clase solo puede usar los atributos de su propia clase o su superclases.

Para el caso de los atributos complejos y métodos complejos se usa la siguiente estructura jerárquica de las clases.



Se asume que el atributo at; 1es el atributo complejo el cual apunta a la Clase4. También, el método m ~.1 de la Clase2 llama los métodos m :,1 y m ~\ de la Clase3 ¡x>rmedio de este atributos complejo at ~ 1 Y m; 2 llama el método m ~2 por medio del mismo atributo. Esto significa que en alguna transacción el método m ~ 1 podría tener la llamada como at ~ 1.m: 1 .

Por la razón del número limitado de atributos que se puede usar en el algoritmo del particionamiento exhaustivo, las presentes clases tienen solamente dos o tres atributos.

Para cada ejemplo, la matriz TMUM demuestra los valores obtenidos después de la multiplicación de esta matriz por las frecuencias de acceso de las transacciones.

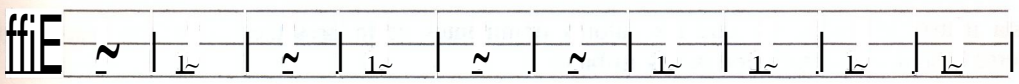
3.1 Atributos simples y métodos simples

Para cada caso se desarrollan varios ejemplos que permiten hacer la comparación de la fragmentación obtenida por medio del análisis teórico previo a la ejecución del programa. Solamente se presenta dos de ellos:

3.1.1 Ejemplo N°1 - Clara distribución de las frecuencias del acceso de los métodos por las transacciones

La siguiente matriz TMUM fue usada:

	1 m1,1	2 m1,2	3 m1,3	4 m2,1	5 m2,2	6 m2,3	7 m3•1	8 m3,2	9 m3,3	10 m3,4
trI	25	25	25	0	0	25	0	0	25	0
tr2	30	30	30	0	0	0	0	0	0	0
tr3	0	0	0	80	80	80	0	0	0	0
tr4	0	4	0	70	70	70	0	0	0	0



Esta matriz seílala IDI segmento visible de las frecuencias más altas y su separación de las frecuenciasmás bajas. Esto facilita la predicción del esquema de fragmentación.

La matriz MAUM refleja la situación cuando los métodos usan los atributos solamente de su propiaclase:

	1 at 1,1	2 at 1,2	3 at1,3	4 at1,1	5 at 2,2	6 at 3,1	7 at 3.2	8 at 3.3
1 ID1,1	2	0	0	0	0	0	0	0
2 1111,2	0	2	0	0	0	0	0	0
3 1111,3	0	0	1	0	0	0	0	0
4 m2,1	0	0	0	1	0	0	0	0
5 m 2.2	0	0	0	0	2	0	0	0
6 1112,3	0	0	0	2	1	0	0	0
7 ID3,1	0	0	0	0	0	2	1	0
8 1113,2	0	0	0	0	0	1	2	0
9 1113•3	0	0	0	0	0	0	1	2
10 1113,4	0	0	0	0	0	0	1	2

Losvalores en MAUM son cero, uno o dos para reflejar no acceso. solamente lectura o lectura y escrituraEstos valores se pueden cambiar si fuera necesario.

Despuésde multiplicar las matrices presentadas anterionnente, la matriz resultante TAUM queda como:

	1 at1,1	2 at1,2	3 at1,3	4 at 2,1	' at2,2	6 at3.1	7 at3.2	8 at3.3
trI	50	50	25	50	25	0	0	0
tr2	60	60	30	0	0	0	0	0
tr3	0	0	0	240	240	10	0	0
tr4	0	8	0	210	210	0	0	0
tr5	10	0	0	0	0	15	25	20
tr6	0	20	0	10	0	30	50	40

En esta matriz se puede ver que los \'alores dominantes están presentes de manera que debería obtenerse la siguiente agrupación de los atributos:

- $at^{1}_{1,1}$, $at^{2}_{1,2}$ y $at^{3}_{1,3}$
- $at^{4}_{2.1}$ y $at_{2.2}$
- $at^{6}_{3.1'}$ $at.'^{7}$-2 y $at^{8}_{3.3}$

En realidad los resultados del programa. donde se usa el criterio del mínimo. dan esta fragmentación como la óptima con el costo mÍlúmo de 5307. Además, a continuación se presenta el costo local. remoto y total para el enfoque optimista con los valores mínimos y su respectiva esquema de fragmentación:

Número Frag	Costo Local	Costo Remoto	Costo Total	Particiones
1	165028	0	165028	(12345678)
2	10408	2862	13270	(123678)(45)
3	1966	3341	5307	(123)(45)(678)
4	863	4950	5813	(12)(3)(45)(678)
5	676	6075	6751	(12)(3)(45)(6)(78)
6	613	8075	8688	(12)(3)(45)(6)(7)(8)
7	363	13800	14163	(1)(2)(3)(45)(6)(7)(8)
8	0	116175	116175	(1)(2)(3)(4)(5)(6)(7)(8)

Los resultados del costo mínimo para cada una de las com¡x>nentes y el costo mínimo total se presentan en la figura N°2.

El comportamiento del PEoo corresponde a las suposiciones teóricas: se obtiene el valor máximo del costo local cuando el número de pariciones es igual al número de atributos. Este valor es igual a cero para un fragmento: el costo remoto en estos dos extremos tiene valores opuestos (mínimo y máximo) respectivamente.

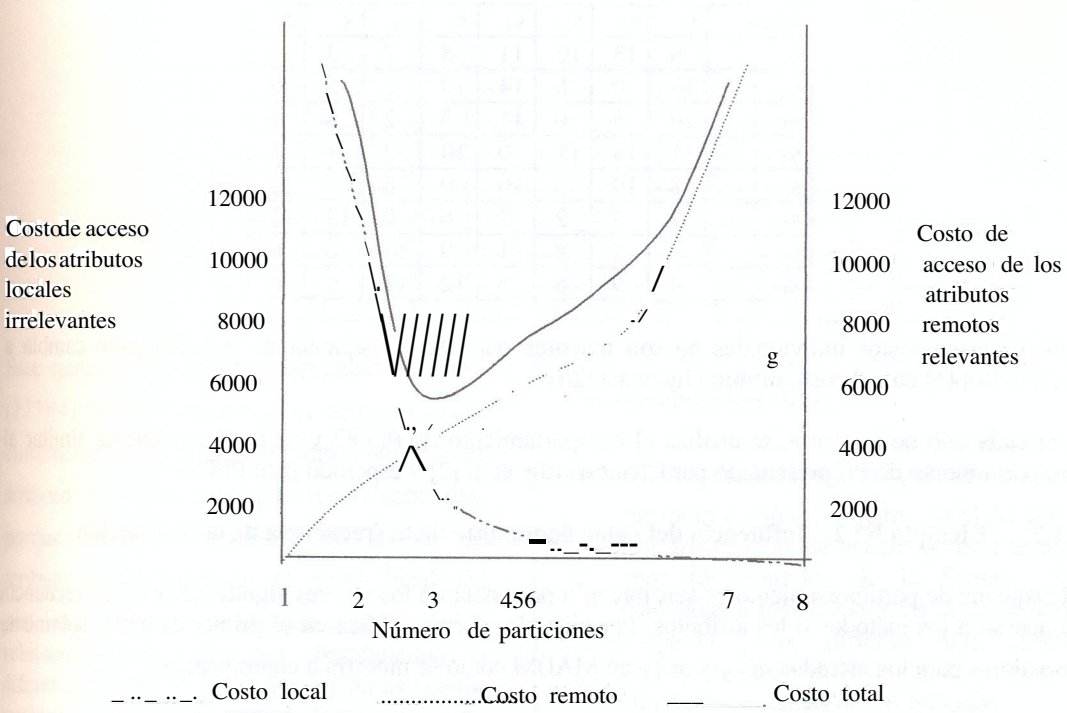


Figura N°2

En este caso el costo de comunicación entre todos los sitios se establece igual a uno para todas las pares de los sitios. Para hacer el análisis de la influencia del costo de comunicación (CC) en el esquema de fragmentación se realizan varios experimentos con CC diferente a uno, pero con el mismo valor para todos los sitios. Se obtienen los siguientes resultados:

CC_{ik}	Costo Mínimo	Esq. Fragment.
17	58763	(123)(45)(678)
18	61924	(123678)(45)
100	149040	(123678)(45)
105	150140	(12345)(678)
180	165028	(12345678)

Aquí, se puede notar como el incremento de costo de comunicación afecta la distribución de los atributos en el esquema óptimo de fragmentación. Por ejemplo, variando el costo de comunicación de uno a diecisiete, el esquema de fragmentación se mantiene igual. El esquema de fragmentación canlbia para $CC_{ik} = 18$. Sin embargo, si se distribuye el costo de comunicación en forma diferente, como por ejemplo:

sitios	SI	S2	S3	S4	Ss	S6	S7	S8
SI	0	13	10	11	8	7	3	4
Sl	13	0	6	14	10	3	5	9
S3	10	6	0	11	3	2	8	6
S4	11	14	11	0	10	2	1	5
Ss	8	10	3	10	0	6	9	12
S6	7	3	2	2	6	0	13	11
57	3	5	8	1	9	13	0	6
S8	4	9	6	5	12	11	6	0

Aunque estos costos individuales no son mayores que 18, el esquema de fragmentación cambia a (12)(3678)(45) con el costo llÚnimo igual a 43261.

Para cada rulO de los casos se analiza el comportamiento de PEoo y se cOllfinna que es sillÚlar al comportallÚento de PE presentado por Chakravarthy et al [2] y esperado para PEoo.

3.1.2 Ejemplo N° 2 - Influencia del valor dominante de la frecuencia de la transacción

El esquema de particionamiento es sensible a la presencia de los valores significantes de la frecuencia de acceso a los métodos o los atributos. Por ejemplo, si se modifica en el primer ejemplo, solamente los valores para los métodos $m_{\sim,3}$ y $at_{\sim,1}$ en MAUM como se muestra a continucación:

	1 $at_{1,1}$	2 $at_{1,2}$	3 $at_{1,3}$	4 $at_{2,1}$	3 $at_{2,2}$	6 $at_{3,1}$	7 $at_{3•2}$	8 $at_{3.3}$
1 $m_{1.1}$	2	0	0	0	0	0	0	0
2 $m_{1,1}$	0	2	0	0	0	0	0	0
3 $m_{1,3}$	0	0	1	0	0	0	0	0
4 $m_{2,1}$	0	0	0	1	0	0	0	0
3 $m_{2,2}$	0	0	0	0	2	0	0	0
6 $m_{2,3}$	0	0	0	0	1	0	0	0
7 $m_{3,1}$	0	0	0	0	0	2	1	0
8 $m_{3,2}$	0	0	0	0	0	1	2	0
9 $m_{3•3}$	0	0	0	0	0	0	1	2
10 $m_{3,4}$	0	0	0	0	0	0	1	2

y se deja la matriz TMUM sin ningunos cambios, se obtiene la matriz TAUM con los correspondientes valores:

	$at_{1\bullet 1}^{1}$	$at_{1.2}^{2}$	$at_{1\bullet 3}^{3}$	$at_{2,1}^{4}$	$at_{2.2}^{5}$	$at_{3.!}^{6}$	$at_{3,2}^{7}$	$at_{3,3}^{8}$
trI	50	50	25	0	25	0	0	0
tr2	60	60	30	0	0	0	0	0
tr3	0	0	0	80	240	0	0	0
tr4	0	8	0	0	210	0	0	0
tr5	10	0	0	0	0	5	25	20
tr6	0	20	0	10	0	30	50	40

Este ejemplo da la mejor fragmentación como (123)(4)(5)(678), poniendo el atribüto at;.2 claramente en el fragmento diferente que el atributo at~.!. Este resultado fue el esperado, porque una alta frecuencia de acceso de atributo $at_{;,2}$ incrementa el costo local (la parte que corresponde al acceso de atributos relevantes con diferentes frecuencias) a diferencia de que si estos dos atributos fueron puestos juntos en el llÚsmo fragmento, este valor cambia de 1603 (con el costo total de 13794)para la fragmentación (123)(4)(5)(678) a 24566 (con el costo total de 25094) para la fragmentación (123)(45)(678).

3.2 Atributos complejos y métodos complejos

Los ejemplos presentados en esta parte confirman la hipótesis presentada por Malillowski [11] según la cual se puede eliminar el atributo complejo del análisis, permitiendo así disminuir el número de atributos en la ejecución del programa y en consecuencia el tiempo de ejecución del prograll1a de la búsqueda exhaustiva. En la vida real, cuando se usan muchos atributos complejos, este mejoramiento es de mayor importancia. Además, si se desarrollan algunos algoritmos heurísticos, este hecho puede también tener influencia en el tiempo de ejecución de los mencionados algoritmos.

Se presentan tres casos:

- El atributo complejo $at_{;,1}$ se trata como un atributo común. La matriz MAUM refleja las frecuencias de acceso de este atributo por los métodos de la Clase2 Las llanladas a los métodos de la clase donde "apunta" este atributo son también reflejadas en MMUM como el caso de llamadas anidadas de los métodos.
- Este atributo complejo se presenta en MAUM, sin embargo, las frecuencias de acceso de este atributo por los métodos m~.1 y $m_{;,2}$ son eliminadas (poniéndoles el valor de cero). Esto perllÚte considerar la posibilidad de elillÚnación de este atributo. La idea que respalda esta decisión es que este atributo no representa ningún almacenallÚento fisico y el acceso a él podría ser ignorado.
- El atributo complejo está eliminado del análisis, presentando el caso de solamente siete atributos en lugar de ocho. Todas las combinaciones que varían de uno a siete se consideran aquí.

En los tres casos se usa los siguientes valores de las matrices:

	1 IIII,1	2 m 1,2	3 m1,3	4 m2J	5 II1 2,2	6 II12,3	7 m3•1	8 m3,2	9 m4,1	10 1114,2	11 m4•3
trl	0	25	0	25	0	0	0	0	25	25	0
tr2	50	0	0	50	0	50	0	0	0	50	0
tr3	0	15	15	0	0	0	15	15	0	0	15
tr4	0	0	0	0	35	35	35	0	0	35	35
trS	25	25	0	0	0	0	25	0	0	25	0

	1 1111,1	2 m1,2	3 m 1.3	4 m2•1	5 m2•2	6 m2•3	7 m3,1	8 m3•2	9 m4,1	10 m4•2	11 m4,3
1 m 1,1	1	0	0	0	0	0	0	0	0	0	0
2 1Ili,2	1	1	0	0	0	0	0	0	0	0	0
3 m 1.3	1	1	1	0	0	0	0	0	0	0	0
4 *mZ.I*	0	0	0	1	0	0	0	0	1	0	1
5 m2,2	0	0	0	0	1	0	0	0	0	1	0
6 m2•3	0	0	0	1	0	1	0	0	0	0	0
7 m3,1	0	1	0	1	0	0	1	0	0	0	0
8 m3,2	0	0	0	0	0	0	0	1	0	0	0
9 m4,1	0	0	0	0	0	0	0	0	1	0	0
10 m4,2	0	0	0	0	0	0	0	0	0	1	0
11 m4,3	0	0	0	0	0	0	0	0	0	0	1

Es importante notar la existencia de los valores en MMUM en las posiciones (4,9), (4,11) Y (5,10) que reflejan las llamadas anidadas entre los métodos:

- $m^{4}_{2,1}$ Y $m^{9}_{4,1}$
- $m^{4}_{2,1}$ Y $m^{11}_{4,3}$
- $m^{5}_{2,2}$ Y $m^{10}_{4•2}$

También se presentan otras llamadas anidadas como: m ~.1y m ~2 ' m :,1 y m ~.1 ' Y otros.

La matriz MAUM es diferentes en cada uno de los casos y en este primer caso se presenta con los siguientes valores:

	$at_{1,1}^{1}$	$at_{1,2}^{2}$	$at_{2,1}^{3}$	$at_{2,2}^{4}$	$at_{3,1}^{5}$	$at_{3,2}^{6}$	$at_{4,1}^{7}$	$at_{4,2}^{8}$
mu^{1}	2	1	0	0	0	0	0	0
$m_{1,2}^{2}$	0	2	0	0	0	0	0	0
$1U1.3^{3}$	1	1	0	0	0	0	0	0
$1U2,1^{4}$	0	0	2	1	0	0	0	0
$m_{2.2}^{5}$	0	0	1	2	0	0	0	0
$m_{2.3}^{6}$	0	0	0	1	0	0	0	0
$m_{3.1}^{7}$	0	0	0	0	2	0	0	0
$m_{3.2}^{8}$	0	0	0	0	1	1	0	0
$1U_{4.1}^{9}$	0	0	0	0	0	0	1	0
$m_{4,2}^{10}$	0	0	0	0	0	0	0	2
$m_{4,3}^{11}$	0	0	0	0	0	0	2	0

Nota la presencia de las frecuencias en la posiciones (4,3) y (5,3) que refleja el uso del atributo at~.1 en 1os ll1eto os $m_{2.1}^{4}$ y $m_{2.2}^{5}$.

En el segundo caso esta matriz tiene un único cambio en las posiciones (4,3)y (5,3)donde en lugar del valor 2 y 1 se ponen los valores iguales a cero. Para el tercer caso se elimina de todo la columna que representa el atributo at ~.I ' quedando así la matriz con siete columnas y once filas.

Es im¡x>rtante analizar los valores en las matrices TAUM después de multiplicar las matrices TMUM, MMUM y MAUM en los tres casos.

Para el primer caso tenemos la siguiente matriz TAUM:

	$at_{1,1}^{1}$	$at_{1,2}^{2}$	$at_{2,1}^{3}$	$at_{2,2}^{4}$	$at_{3,1}^{5}$	$at_{3,2}^{6}$	$at_{4,1}^{7}$	$at_{4,2}^{8}$
trI	50	75	50	25	50	0	100	50
tr2	100	50	200	150	0	0	150	100
tr3	75	135	30	15	45	15	30	30
tr4	0	70	175	175	70	0	70	210
trs	100	150	50	25	50	0	0	50

En el segtmdo caso los valores en esta matriz son exactamente iguales excepto que la tercera colunilla ahora tiene solamente los valores iguales a cero.

En el tercer caso. donde se elimina el atributo complejo, los valores calculados de la matriz TAUM son iguales al segundo caso. excepto que desaparece de todo la tercera colunilla:

	$at^{1}_{1.1}$	$at^{2}_{1.2}$	$at^{-t}_{2.2}$	$at^{\sim}_{3.1}$	$at^{6}_{3.2}$	at^{7}-t.l	$at^{8}_{4.2}$
trI	50	75	25	50	0	100	50
tr:!	100	50	150	0	0	150	100
tr3	75	135	15	45	15	30	30
tr_4	0	70	175	70	0	70	210
trs	100	150	25	50	0	0	50

Los esquemas óptimos de fragmentación para cada uno de los casos son las sigtLÍentes:

Caso	Esq. de Fraglnent.	Mínimo PE00
1	(12)(3478)(5)(6)	70314
2	(12)(36)(478)(5) (12)(3)(478)(5)(6)	62998 62998
3	(12)(478)(5)(6)	62998

En el primer caso el valor de PEoo es más alto por la inclusión de las frecuencias del acceso de atributo complejo por los métodos correspondientes. Sin embargo. este atributo complejo en realidad no representa ningún almacenamiento fisico y se podría usar como un puntero en la memoria principal cuando la base de datos se abre [9]. Como el PEoo no distingue los atributos de la memoria principal y todos los accesos son tratados como accesos al disco, por lo tanto hasta los accesos a la memoria principal se consideran de la llÚsma manera como accesos al disco, aunlentando el valor núnimo de PEoo. En consecuencia se puede decir que los valores de acceso al atributo complejo deberían ser iguales a cero, si se quiere conservar el número total de atributos para análisis de esquemas de fragmentación. En otro caso, tratando este atributo como el atributo simple el costo de fragmentación se incrementará innecesariamente. De acuerdo al desarrollo de PEoo [11], el costo local refleja el costo en unidades de acceso al disco; pues este PEoo no puede ser aplicado de la misma forma a los atributos accesados desde la memoria principal.

El segundo y tercer caso dan los mismos valores de PEoo. Adicionalmente, el atributo complejo presentado en el segtmdo caso (atributo 3) está localizado junto con el atributo 6 porque en la matriz TAUM los valores presentados en la tercera y sexta columnas son más cercanos y disminuyen el costo de penalidad a diferencia de que si estos atributos fueran puestos en diferentes fragmentos. Cabe recordar, que en este ejemplo se ignora el valor real del costo de la comunicación y el factor dominante a considerar son las frecuencias de acceso.

Para el tercer caso de atributos complejos y métodos complejos cuando el atributo complejo no está presente en el algoritmo de fragmentación, después de obtener el esquema de fragmentación se puede poner este atributo en el llÚsmo fragmento donde está localizado el atributo 6 o crear otro fragmento para él. Así se obtiene exactamente los mismos

esquell1as de fragmentación que presentados para el segundo caso. Pues, se puede concluir

el esquema de fragmentación. La ventaja en esta caso es de usar menos atributos y en consecuencia menos combinaciones para encontrar el esquema óptimo y este hecho en la presencia real de varios atributos complejos puede disminuir considerablemente el tiempo de ejecución.

Adicionalmente, siguiendo la lógica, se puede asumir que el atributo complejo puede ser puesto en cualquier fragmento que lo necesita, porque él puede tener varias copias sin ningunas consecuencias con respecto a las modificaciones y el tiempo de acceso.

que la eliminación del atributo complejo del análisis no afecta.

3.3 Atributos complejos y métodos simples

Para confirmar de nuevo la hipótesis sobre la eliminación del atributo complejo se presenta un experimento más donde los métodos complejos (llamadas anidada) son ausentes, excepto los métodos que reflejan las llamadas por medio del atributo complejo. Para los mismo valores de la matriz TMUM y MAUM presentados en el caso previo, se usa la siguiente matriz MMUM:

	1 m1,1	2 m1•1	3 m 1.3	4 *m1,1*	5 *m1,2*	6 *m1•3*	7 m3,1	8 *m3,1*	9 m4,1	10 m4•2	11 m4.3
1 m1,1	1	0	0	0	0	0	0	0	0	0	0
2 m1,2	0	1	0	0	0	0	0	0	0	0	0
3 m1•3	0	0	1	0	0	0	0	0	0	0	0
4 m2,1	0	0	0	1	0	0	0	0	1	0	1
5 *m2,1*	0	0	0	0	1	0	0	0	0	1	0
6 *m1,3*	0	0	0	0	0	1	0	0	0	0	0
7 m3,1	0	0	0	0	0	0	1	0	0	0	0
8 m3,2	0	0	0	0	0	0	0	1	0	0	0
9 m4,1	0	0	0	0	0	0	0	0	1	0	0
10 *m4,1*	0	0	0	0	0	0	0	0	0	1	0
11 m4,3	0	0	0	0	0	0	0	0	0	0	1

Esta matriz se presenta solamente para reflejar el acceso al atributo de Clase4 por medio del atributo de la Clasez. De acuerdo a Malinowski [11] este caso se convierte al caso de atributos simples y métodos simples.

Tomando en cuenta la matriz MAUM se realiza el experimento para los siguientes dos casos: primero donde se presentan los ocho atributos y el segundo donde se elimina la columna que representa el atributo complejo (columna 3).

En estos dos casos los valores obtenidos de la matriz TAUM difieren solamente en la tercera columna, donde en el segundo experimento el atributo compuesto fue eliminado y de la misma forma no aparece en la matriz resultante de TAUM.

Para el primer caso los valores de TAUM obtenidos son los siguientes:

	1 at 1.1	2 at1•2	3 at 2.1	4 at2,2	5 at .1.1	6 at.␣	7 at4J	8 at4,2
trI	0	50	50	25	0	0	100	50
tr2	100	50	100	100	0	0	150	100
tr3	15	45	0	0	45	15	30	0
tr4	0	0	35	105	70	0	70	140
trs	50	75	0	0	50	0	0	50

Si se elimina la tercera columna. se obtienen los resultados del segundo caso.

Los costos mínimos de la fragmentación con su respectivo esquema para cada caso son los siguientes:

Caso	Esq. de Fragment.	MínimoPEoo
1	(125)(3478)6)	36349
2	(125)(478)(6)	32412

El com¡x>rtanliento del PEoo para los dos casos se presenta en la forma gráfica en la figura N°3.

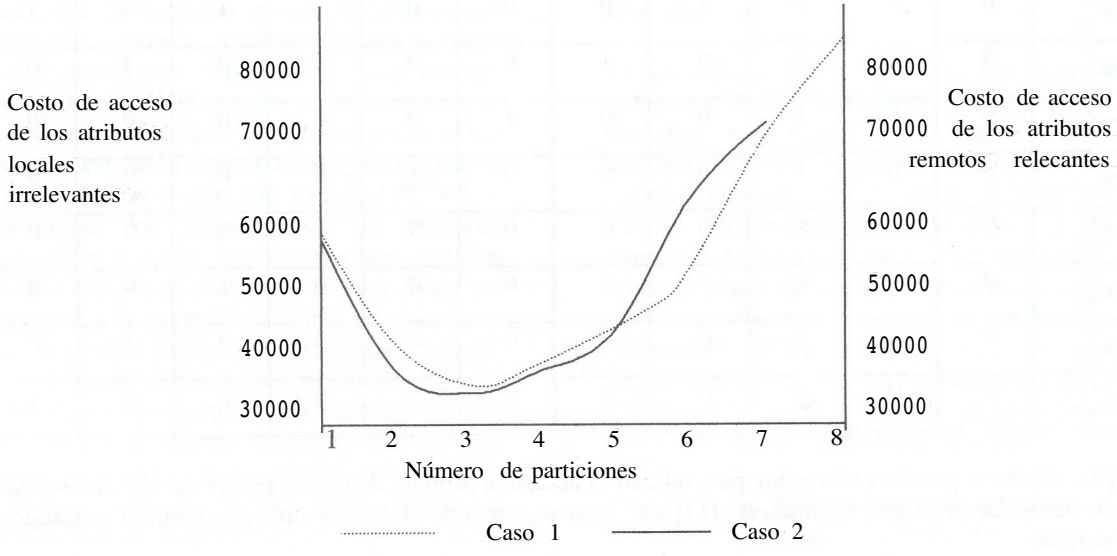


Figura N° 3

La explicación del aumento del costo mínimo en el caso del uso de ocho atributos en comparación con los siete atributos después de la eliminación del atributo complejo es exactamente la misma que la presentada en el caso de atributos complejos y métodos complejos.

Así, podell1osconcluir que se demuestra como verdaderala lúpótesis de considerar el caso de atributos complejos y métodos simples y el casode atributos complejos y métodos simples comoel caso de atributos simples y métodos complejos después de la eliminación del atributocOll1plejodel proceso de análisis

4. CONCLUSIONES

Las bases de datos relacionales usadas en las redes de las computadoras abrieron las posibilidades de compartir datos y distribuirlos de acuerdo a su uso, dando así el paso a las bases de dátos distribuidas relacionales. Siendolas bases de datos orientadas a objetos una generalización de las bases de datos relacionales es de esperar que la distribución los objetos sobre la red pueda ser una alternativainteresante a investigar.

La complejidad de estas BDoo se puede simplificar si se presentan las clases en las cuatro categorías de atributos simples y métodos simples, atributos simples y métodos complejos, atributos complejos y métodos simples, y atributos complejos y métodos complejos. Para cada una de estas categorías se puede recopilar la información sobre las frecuencias del uso de los atributos y métodos de la forma similar como se hizo en la BDDR. Teniendo a la dis¡x>sición esta información se puedeaplicar la función de evaluación llamada PEOO para obtener la fragmentación óptima del conjunto de todas posibles fragmentaciones de los atributos de las clases.

Basándose en los ejemplos simples desarrollados para cada una de las categorías de las clases se puede notar que este evaluador de particiones tiene el mismo comportamiento que el evaluador de particiones desarrollados para BDDR. Además. se confimla la hipótesis de que los atributos complejos pueden ser excluidos de los algoritmos de fragmentación porque en sí no representan el acceso físico al disco y su replicación en cualquier nodo que los necesite no demanda costos adicionales tanto de comunicación como de la modificación.

Los experimentos basados en PEoo demuestran que es posible generalizar algunos algoritmos desarrollados para BDR y aplicarlos a las BDoo.

5. REFERENCIAS

[1] Comell D., et. al., *Vertical Partitioning AIgorithm jar Relational Databases*. Proc. Third Ibtemational Conference on Data Engineering, Febrero 1987.

[2] Chakravarthy S., et. al., *An Objective Function for Vertically Partitioning Relations in Distributed Databases and its Analysis*. Distributed and Parallel Databases, Vol. 2 , No. 1. 1993.

[3] Chu Pai-Cheng. *A lransaction Oriented Approach fo Attribute PartitioninK* Inforrnation Systems, Vol. 17. NO.4. 1992.

[4] Chu W.. et. al. *A Transction Based Approach fo Vertical Partitioning for Relational Dalabase Sysfems*. IEEE Transactions on Software Engineering, Vol. 19, No. 8, Agosto 1993.

[5] Ezeife C.I., et. al., *Comprehensive Approach lo Horizontal Class Fragmentation in a Dislributed Object Based System*. Technical report, Advanced Database Systems Laboratory, Department of Computer Science, University of Manitoba, Canada, October 1994.

[6] Gruber O., et. al., *Object Grouping in EOS* en M.I. Ozsu, U.Dayal y P.Valduriez (eds.) Distributed Object Management. Morgan Kaufman Publishers, San Mateo, Ca 1994.

[7] Hammer M., et. al., *A Heuristic Approach to Attribute PartitioninK* Proc. ACM SIGMOD International Conference on Management of Data. Boston, MA, 1979.

[8] Karlapalem K., et. al., *Partitioning Schemes for Object Oriented Databases*. Technical repon., University of Science and Technology, Department of Computer Science Clear Water Bay, Kawloon, Honk Kong, Agosto 1994a.

[9] Karlapalem K., et. al., *lssues in Distribution Design* en M.T. Ozsu, U.Dayal y P.Valduriez (eds.) Distributed Object Management. Morgan Kaufman Publishers, San Mateo, Ca 1994.

[10] Lin X., et. al., *A Graph Based Cluter Approach for 'Vertical Partitioning in Database System*. Data & Kllowledge Engineering. Vol. 11, 1993.

[11] Malinowski E. *Fragmentation Techniques for Distributed Object-Oriented Databases*. Tesis de Maestría. Universidad de Florida en Gainesville, 1996.

[12] Navathe S., et. al., *Vertical Partitioning AIgorithm for Database Design*. ACM Transaction on Database Systems. Vol.9, No.4, Diciembre 1984.

[13] Navathe S., et. al., *Vertical Partitioning for Database Design: A Graphical Algorithm*. ACM SIGMOD. Portland, June 1989.

[14] Stonebraker *M.lnclusion of New Types in Relational Data BaseSystems* en M. Stonebraker (ed) Readings in Database Systems. Morgan Kaufman Publishers, 1994.