Algoritmo de Reunión Espacio-Temporal usando estructura 3DR-tree podada

Anibal J. Díaz *
Departamento de Informática, UTN Universidad Tecnológica Nacional
Concepción del Uruguay, Entre Ríos - Argentina
anibaljdiaz@yahoo.es

and

Gilberto A. Gutiérrez **
Departamento de Auditoría e Informática
Universidad del Bio-Bio - Chillán, Chile
ggutierr@ubiobio.cl

and

Olinda E. Gagliardi
Departamento de Informática - Facultad de Ciencias Físico-Matemáticas
Universidad Nacional de San Luis - Argentina
oli@unsl.edu.ar

Abstract

The join query is one of the most important operation in the Database Management System (DBMS). Because of the modern (DBMS) have incorporated new types of data bases (temporal, spatial, among others), it has become necessary to possess new algorithms to process join queries in these areas. Nowadays the databases have been receiving major importance due to the wide variety of applications that they need to process the spatial history of the objects. The spatio-temporal access methods proposed till now are orientated to trying principally timeslice and interval query and they do not approach the join query. In this work we present and evaluate a join space-temporal algorithm based on it structures 3DR-tree and that benefits by the applied pruned technique from. Also it is introduced an analytical model about the algorithm that estimate the cost in terms of node or disk accesses. Finally the predictive behavior of the proposed model will be compared with the 3DR-tree cost model showing the accuracy of this model.

Keywords: Spatiotemporal Join (STJ), 3DR-tree, Reunión espacio temporal, Modelo de costo, 3DR-tree podado.

*Este trabajo es parcialmente subvencionado por los Proyecto de Tecnologías Avanzadas de Bases de Datos 22/F314, Departamento de Informática, UNSL; y el Proyecto AL06-PF-013 Geometría Computacional.

**Parcialmente financiado por el proyecto Algoritmos para la evaluación de consultas espacio-temporales, código 073218 A/R de la Universidad del Bio-Bio.

Resumen

La consulta de reunión (join) es una de las operaciones fundamentales en un Sistema de Administración de Bases de Datos (DBMS). Debido a que los (DBMS) modernos han incorporado nuevos tipos de bases de datos (temporal, espacial, entre otros), se ha hecho necesario contar con nuevos algoritmos para procesar consultas de reunión en esas áreas. Actualmente las bases de datos espacio-temporales han ido cobrando mayor importancia debido a la amplia variedad de aplicaciones que requieren procesar la historia espacial de los objetos. Los métodos de acceso espacio-temporales propuestos hasta ahora estan orientados a procesar principalmente consultas de tipo time-slice y de tipo interval y no abordan la consulta de reunión. En este trabajo presentamos y evaluamos un algoritmo de reunión espacio temporal basado en la estructuras 3DR-tree y que se beneficia por la técnica de podado aplicada. También se introduce un modelo analítico del algoritmo que estima el costo en términos de accesos a nodo o accesos a disco. Finalmente se evalua el comportamiento predictivo del modelo propuesto con el modelo original de costo del algoritmo de reunión espacio-temporal 3DR-tree mostrando la precisión del mismo.

1 Introducción.

1.1 Conceptos preliminares.

Los objetos espacio-temporales, esto es objetos cuyo atributo espacial cambia a través del tiempo, aparecen en muchas aplicaciones [5,9,11]. La naturaleza de tales aplicaciones requiere del procesamiento de grandes cantidades de datos espacio-temporales generados de manera muy frecuente. El procesamiento eficiente de consultas, sobre estos volúmenes de datos, demanda de nuevos, sofisticados y eficientes esquemas de indexación o métodos de acceso espacio-temporales. Los métodos propuestos hasta ahora apuntan principalmente a resolver dos tipos de consultas espacio temporales conocidas como time-slice y time-interval [12]. El primer tipo recupera todos los objetos que intersectan un rango espacial, en un instante de tiempo especifico, en cambio el segundo extiende la idea a un intervalo de tiempo. Otro tipo de consultas en ambientes espacio-temporales de reconocida importancia en [8,11,10] es el de reunión para el cual, de acuerdo a lo que conocemos, no existen esquemas de índices ni algoritmos, salvo 3DR-tree [20] en donde la reunión espacio-temporal corresponde a una reunión espacial considerando el tiempo como otra dimesión espacial como condición indispensable para su procesamiento eficiente.

1.2 Definición del Problema.

La operación de reunión espacial, $S1 \bowtie_\theta S2$, combina dos conjuntos de objetos espaciales S1 y S2, basado en el predicado θ. En el ámbito espacio temporal se puede definir la consulta de reunión espacio-temporal como $S1 \bowtie_{\theta \wedge \tau} S2$, pero ahora S1 y S2 son dos conjuntos espacio-temporales, donde τ corresponde a un predicado temporal y θ al espacial.
Por lo tanto y de acuerdo a [8] *"Dadas dos relaciones espacio-temporales $S1$ y $S2$, la consulta de reunión espacio-temporal se define como la operación de hallar los pares de objetos que pertenecen a ambos conjuntos y cuyas extensiones se intersectaron durante el intervalo de tiempo t "*

De esta forma podemos responder consultas como : *Encontrar todos los autos que tuvieron una colisión el dia de ayer*. Suponiendo que el conjunto contiene todas las posiciones de los autos, la consulta anterior implica buscar todos los autos cuyas trayectorias se intersectaron en el intervalo de tiempo consultado. Claramente esta consulta se resuelve por medio de una reunión espacio temporal.

2 Antecedentes.

En el modelo de dato espacio-temporal, los objetos espacio-temporales son aquellos que pueden cambiar su forma y su posición en el espacio, o cambiar su forma sin cambiar la posición o cambiar su posición y no su forma. En este campo, y tomando como ejemplo objetos en movimiento cuya geometria propia no se modifica, la operación de reunión espacio-temporal tiene como antecedente la propuesta por [1] y que se aplica sin modificaciones en la estructura de dato 3DR-tree. Como nuestro algoritmo de reunión espacio-temporal se basa en este método de acceso, en esta sección describiremos el mismo junto con el algoritmo de reunión para dicha estructura y su modelo de costo.

2.1 Método de acceso espacio-temporal 3DR-tree

3DR-tree, es un método de acceso espacio-temporal que considera el tiempo como una dimensión adicional a las coordenadas espaciales. De esta forma cuando un objeto, en un espacio de 2 dimensiones, cambia su posición o forma geométrica se inserta un MBR formado por 2 puntos en 3 dimensiones. Un MBR es un "rectángulo mínimo envolvente", el cual es capaz de contener los atributos espaciales del objeto. Este rectángulo mínimo es la aproximación geométrica de un objeto real, que se guarda en el nodo hoja del árbol, y los nodos internos representan una sucesión de regiones rectangulares minimales, cada una de las cuales controla un nodo en el nivel inferior.

La estructura 3DR-tree es muy eficiente en el uso del almacenamiento y en el procesamiento de consultas de tipo time-interval. Sin embargo, una de sus principales desventajas es su ineficiencia para consultas de tipo time-slice debido a la gran altura que puede alcanzar provocada por la gran cantidad de MBR's que se deben insertar. Esta deficiencia se reduce considerablemente gracias a la técnica de podado que se aplica a la estructura de indexación como se explica en la Sección 3.

2.2 Algoritmo de reunión espacio-temporal basado en 3DR-tree

Dado que un 3D R-tree es un R-tree en un espacio de 3 dimensiones, la implementación de la consulta de reunión espacio-temporal se resuelve en forma directa utilizando el procedimiento propuesto en [1] para la reunión espacial y que se muestra en el *Algoritmo 1*. Dicho algoritmo supone que los dos conjuntos de objetos tienen disponible un índice espacial basado en R-tree. Cabe destacar que el procesamiento de la reunión espacial considera dos etapas: filtrado y refinamiento. En la etapa de filtrado se obtiene un superconjunto de objetos los cuales son seleccionados de acuerdo a su aproximación espacial(MBR). Posteriormente, en la etapa de refinamiento (utilizando la geometría exacta de los objetos) se verifica si se cumple o no el predicado espacial. El procedimiento descrito por el Algoritmo 1 (**SJ**) se utiliza para la etapa de filtrado y funciona tanto si los árboles tienen igual o diferente altura.

2.3 Modelo analítico para la reunión espacial usando R-tree

Ya que 3DR-tree es utilizado para comparar nuestro algoritmo de consulta de reunión espacio-temporal, es que en esta sección describimos el modelo de costo que se plantea para la reunión espacial. Primero explicamos un modelo para consultas espaciales de tipo "window query" y luego para la consulta de reunión espacial.

Existen varias propuestas para estimar el rendimiento de una consulta espacial (windows query) sobre un R-tree [6,10,13,16,18,19]. Uno de los modelos más conocido es el descrito en [16,19].

En [19] se propone un método para estimar el costo de una operación de reunión espacial. Dicho modelo asume un espacio multidimensional y que existen dos conjuntos espaciales de cardinalidad N_{R1} y N_{R2} respectivamente y que cada uno de ellos cuenta con un R-tree, $R1$ y $R2$. El objetivo del modelo es obtener el número promedio de nodos (NA) accesados o número promedio de bloques de disco leidos (DA) por una operación de reunión espacial. La diferencia entre NA y DA se deben a que en el cálculo de NA no se

Algorithm 1 Algoritmo para procesar Reunión espacial de objetos espacio- temporales

```
1: SJ(R1, R2) {R1 y R2 corresponden a nodos de R-tree(3DR-tree)}
2: for all E1 in R1 do
3:    for all E2 in R2 do
4:       if IntersectMBR(E1.MBR, E2.MBR) then
5:          if R1 y R2 son hojas then
6:             output(E1.Oid, E2.Oid)
7:          else if R1 es hoja then
8:             ReadPage(E2.ptr)
9:             SJ(E1.ptr, E2.ptr)
10:         else if R2 es hoja then
11:            ReadPage(E1.ptr)
12:            SJ(E1.ptr, E2.ptr)
13:         else
14:            ReadPage(E1.ptr)
15:            ReadPage(E2.ptr)
16:            SJ(E1.ptr, E2.ptr)
17:         end if
18:      end if
19:   end for
20: end for
```

considera un sistema de buffer mientras que para DA si. De esta forma la desigualdad DA $\leq$ NA siempre se cumple. En primer lugar, suponiendo que R1 y R2 tienen la misma altura h y que las dos raíces se almacenadas en memoria, el valor de NA según[19] se define en la Ec.(1) :

$$NA_{total}(R_1, R_2) = \sum_{j=1}^{h-1} \{NA(R_1, j) + NA(R_2, j)\} \tag{1}$$

donde $NA(R_i; j)$ es el número promedio de nodos en el nivel j del R-tree i y está dado por :

$$NA(R_i, j) = N_{R_{\mathbf{2}},j}.N_{R_{\mathbf{1}},j} \prod_{k=1}^{n} min\{1, (s_{R_{\mathbf{1}},j,k} + s_{R_{\mathbf{2}},j,k})\} \tag{2}$$

y según [18] el costo total de la reunión espacial vale:

$$NA(R_1, R_2) = \sum_{j=1}^{h-1} N_{R_{\mathbf{1}},j} \left(N_{R_{\mathbf{2}},j} \prod_{k=1}^{n} [S_{R_{\mathbf{1}},j,k} + S_{R_{\mathbf{2}},j,k}] + 1 + N_{R_{\mathbf{2}},j+1} \left[\prod_{k=1}^{n} (S_{R_{\mathbf{1}},j,k} + S_{R_{\mathbf{2}},j+1,k}) \right]^2 \right) \tag{3}$$

Pero en la expresión para la cantidad de nodos promedio por nivel debemos tener en cuenta el rango de crecimiento de los mismos a través del tiempo en la estructura. Luego debemos considerar el efecto del tiempo como lo expresa [14]:

$$\mathbf{N_{i,T}} = \mathbf{N_{i,0}} + \mathbf{N_{i,0}} \cdot (\mathbf{T} - \mathbf{1}) \cdot \mathbf{a}_* \tag{4}$$

donde $\mathbf{N_{i,T}}$ es el número total de objetos del árbol **i** al cabo de un tiempo **T**, $N_{i,0}$ es el número incial de objetos en el árbol i (T=0) y a_* es el rango promedio de crecimiento en la estructura. En caso de haber variacion de la forma del objeto con el paso del tiempo en $\mathbf{a}_*$ se debe considerar el promedio de cambio de forma a_{ms} con lo cual $\mathbf{a}_* = \mathbf{a_{ext}} + \mathbf{a_{ms}} - \mathbf{a_{ext}} \cdot \mathbf{a_{ms}}$ [14].

3. Algoritmo de Reunión espacio-temporal usando 3DR-tree podado.

3.1 Estructura espacio-temporal APR-tree.

La estrategia de particionar el árbol principal de indexación la aplicamos para reducir su altura, generando varios árboles 3DR-tree-podados. Un árbol 3DR-tree-podado es una estructura arborea 3DR-tree pero de

menor altura la cual se obtiene dividiendo la estructura principal en múltiples 3DR-trees responsables cada uno de un intervalo temporal $L_i < T$.

Una idea similar es la que se propone en [7] donde dividen el árbol principal en árboles más pequeños de acuerdo con el tamaño de la carga de trabajo, llamándo al resultado de la partición estructura APR-tree. En esta estructura el intervalo temporal $\mathbf{L_i}$ que administra cada 3DR-tree-podado depende de la carga de trabajo de la estructura principal. En la Figura 1 se puede observar las longitudes temporales L_1, L_2 y L_i de las estructuras R_1, R_2 y R_i del APR-tree.

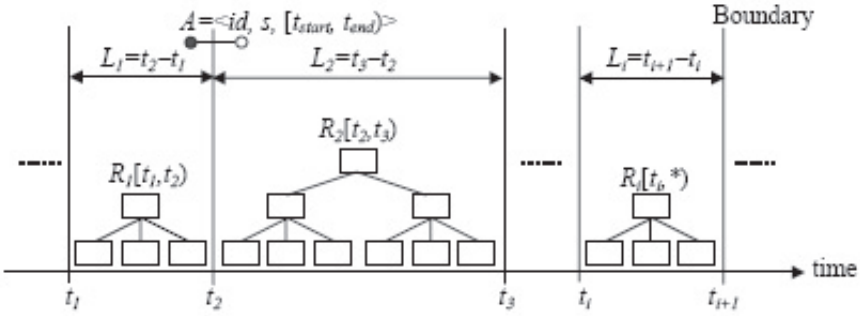

Figura 1: Estructura APR-tree

El APR-tree no comparte nodos, pero el particionado del árbol principal genera duplicados en los árboles podados. Una entrada tiene la forma $A\langle id, s, [t_s, t_e)\rangle$, donde id es el identificador del objeto, s es el MBR espacial, y t_s, t_e son los instantes en que el objeto se ha ingresado a la base de datos y en que se ha modificado. Cuando una entrada por ejemplo A_1 intersecta el límite del intervalo temporal L_1 entre las estructuras R_1 y R_2, se divide el lifespan de la entrada en dos y obtenemos $A_{11}\langle id, s, [t_s, t_2)\rangle$ y $A_{12}\langle id, s, [t_2, t_e)\rangle$ insertándo A_{11} en R_1 y A_{12} en R_2. Esto es una limitación a la hora de realizar la búsqueda en los nodos para determinar los pares de objetos que se solapan. Sin embargo esta limitación se ve minimizada gracias a la reducción en altura obtenida en la división de la estructura principal y por la gran reducción de espacio muerto obtenida por el fraccionamiento de los datos.

3.2 Nuestra propuesta .

Sean los conjuntos S1 con N1 objetos espacio-temporales con una historia T1 y S2 con N2 objetos y una historia T2. Aplicando la metodología de partición de [7] obtenemos K1 y K2 árboles podados de cada conjunto y cuyas raíces e intervalos temporales guardamos en arreglos A y B respectivamente, es decir:
$A = \{S1_1, S1_2, S1_3, ..., S1_{K1}\}$ y $B = \{S2_1, S2_2, S2_3, ..., S2_{K2}\}$
Siendo los intervalos temporales de cada conjunto :
$T1 = \{L1_1, L1_2, ..., L1_{K1}\}$ y $T2 = \{L2_1, L2_2, ..., L2_{K2}\}$
Una vez finalizado el paso anterior se define el intervalo de vida promedio por conjunto, $L1$ y $L2$ como $L1 = \frac{\sum_{i=1}^{K1} L1_i}{K1}$, donde $(L1_i \cap L1_{i+1}) = \emptyset$ y $\bigcup_{i=1}^{K1} L1_i = T1$, análogamente para L2.

Partiendo de los arreglos A y B pasamos a procesar la reunión implementando el *Algoritmo 2* sobre los pares de estructuras de A y B. Este algoritmo utiliza la información de que los conjuntos A y B se encuentran ordenados por el tiempo. El algoritmo comienza eligiendo el intervalo más antiguo entre los conjuntos A y B. Luego verifica todos aquellos intervalos que se intersectan con él y cumplan que el límite inferior del intervalo que intersecta sea inferior al límite superior del intervalo más antiguo. Se verifica entonces que los 3DR-tree-podados responsables de esos intervalos, se intersecten espacialmente y si cumplen esa restricción entonces se procesa la reunión espacial simple (*Algoritmo 1*: SJ(RA_i,RB_j)). En este algoritmo se modifica el final de la línea 4, agregando $AND(E_2.t_e \leq R1.t_e)$ $AND(E_1.lifespan \cap E_2.lifespan)$, a efectos de tener en cuenta la existencia de duplicados generados por la metodología de partición de la estructura principal

y para verificar que las entradas cumplen con el predicado de la reunión espacio temporal. A continuación se selecciona el siguiente intervalo más antiguo y se repite el proceso hasta alcanzar el límite $K1$ o $K2$ del menor de los conjuntos tratados.

Algorithm 2 Algoritmo de reunión mejorado sobre arboles podados de A y B

```
1: Filtrar(A,B) {RA_i y RB_j son 3DR-tree-podados de A y B respectivamente}
2: i=0; j=0;
3: while (i < K1 and j < K2) do
4:    if (RA_i.t_s ≤ RB_j.t_s) then
5:       if (RA_i.t_e > RB_j.t_s) then
6:          if IntersectMBR(RA_i.MBR, RB_j.MBR) then
7:             SJ(< RA_i, RB_j >)
8:          end if
9:          Antiguo(RA_i, RB_j)
10:      else
11:         i++;
12:      end if
13:   else
14:      if (RB_j.t_e > RA_i.t_s) then
15:         if IntersectMBR(RA_i.MBR, RB_j.MBR) then
16:            SJ(< RA_i, RB_j >)
17:         end if
18:         Antiguo(RA_i, RB_j)
19:      else
20:         j++;
21:      end if
22:   end if
23: end while
```

Algorithm 3 Algoritmo que mantiene el arbol podado más antiguo de A y B

```
1: Antiguo(RA_i,RB_j)
2: if (RA_i.t_e > RB_j.t_e) then
3:    j++;
4: else
5:    i++;
6: end if
```

El **algoritmo 2** procesa exactamente aquellos 3DR-tree-podados que se intersectan temporalmente, cuyo costo se reduce por la restricción de intersección espacial que pueda existir entre pares de 3DR-tree-podados.

4. Modelo de costo

Asumimos que todos los objetos están uniformemente distribuidos en un espacio d-dimensional[18]. En segunda instancia, la historia T de la base de datos se ha particionado en K subestructuras de altura H_i, obteniéndose K 3DRp-tree (3DR-tree podado, en adelante árbol). Los que son responsables de un intevalo de vida (*lifespan*) promedio $L_i = \sum_{i=1}^{K} L_i/K$

Sea $N_{R_i,j}$ el número de nodos del árbol i en el j-ésimo nivel, y sea $P_{i,j}$ la probabilidad de que un nodo del árbol i, a nivel j sea visitado para responder la consulta de reunión. Suponiendo que los tamaños promedios espacial y temporales de los nodos de los árboles son iguales. Entonces la cantidad de nodos accesados para responder una consulta de reunión espacio-temporal que involucre un único árbol de A y un único objeto del conjunto B estará dada por la Ec(3).

4.1 Costo de la reunión mejorada

De acuerdo a estas consideraciones aplicadas al algoritmo de reunión espacio-temporal mejorado sobre los árboles de los conjuntos A y B tenemos:

Cuadro 1: Lista de símbolos usados

Símbolo	Definición
M	capacidad máxima del nodo
f	fanout promedio del nodo
c	capacidad promedio de nodo en %
h	altura del arbol primario 3DR-tree .
H	altura promedio del subarbol podado 3DRp.
N_i	Número de objetos en el conjunto inicial de datos
$N_{i,j}$	Número de entradas del arbol i en el nivel j.
D_i	Densidad de nodo a nivel i
$S1$,S_2	Conjuntos de datos 1 y 2
a_i	Agilidad del árbol i
T	Número total de instantes de tiempo en la historia de la base
L_i	amplitud del intervalo de vida de un árbol podado;
$s_{i,j}$	Tamaño promedio del nodo s en el arbol i en el nivel j
NA_i	Número promedio de nodos accesados por la operación en el arbol i
NA	Número total promedio de nodos accesados en la reunión
$C(STJ)_{3DRp}$	Costo de reunión espacio temporal sobre 3DR podado.
K	Número de arboles podados .

$$C_{STJ_{\mathbf{3DR1}}} = [\mathbf{C(A)} + \mathbf{C(B)}] \cdot \mathbf{P}^{'} \cdot \mathbf{C(SJ)_{3DRp}}$$

donde C(A)=K1 y C(B)=K2 son el costo leer los conjuntos A y B respectivamente y $\mathbf{P}^{'}$ es la probabilidad de intersección espacial de los 3DR-tree-podados de A y B. En este caso se trata de hallar la probabilidad de intersección del MBR que cubre todo el 3DR-tree-podado de uno de los árboles del conjunto A, con el MBR que cubre todo un 3DR-tree-podado del conjunto B. Observamos que el área máxima que determinan estos dos 3DR-tree-podados es igual a la suma de sus respectivos MBR ($RA_i.MBR + RB_j.MBR$), siendo el universo muestral el área ocupada por el MBR de todo el conjunto B y cuya expresión esta dada por S2.MBR $= \sum_{j=1}^{K2} \cdot \sum_{n=1}^{H-1} N2_{j,n} \cdot s2_{j,n}$, donde $N2_{j,n}$ es la cantidad promedio de nodos del j-ésimo árbol podado del conjunto B a nivel n y $s2_{j,n}$ es el tamaño promedio del nodo de ese árbol a nivel n. Relacionando ambas expresiones a través del concepto de probabilidad, determinamos que la probabilidad espacial de intersección de los árboles podados RA_i y RB_j esta dada por :

$$P_\theta = \frac{RA_i.MBR + RB_j.MBR}{K2 \cdot \sum_{n=1}^{H-1} N2_n \cdot s2_n}$$

$$\text{donde : } RA_i.MBR = \sum_{n=1}^{H-1} N1_{i,n} \cdot s1_{i,n} \qquad \text{y} \qquad RB_j.MBR = \sum_{n=1}^{H-1} N2_{j,n} \cdot s2_{j,n}$$

$s1_{i,n}$ es el tamaño promedio de los nodos del i-ésimo 3DR-tree-podado del conjunto $A \in S1$ a nivel n y $s2_{j,n}$ es el tamaño promedio de los nodos del j-ésimo 3DR-tree-podado del conjunto $B \in S2$ a nivel n y cuyos valores se obtienen de la ecuación: $s_{i,n} = \sqrt{D_{i,n}/N_{i,n}}$ [19]. En la ecuación utilizaremos la expresión para la probabilidad $P^{'} = min[1; P]$ dada por [14].
Luego reemplazando valores nos queda:

$$\Longrightarrow \mathbf{C_{(STJ1)_{3DRp}}} = [K1 + K2] \cdot min[1; P_\theta] \cdot C(SJ) \tag{5}$$

$C(SJ)$ es el costo de la reunión de los elementos de las estructuras de RA_i y RB_j que pertenecen a los arreglos A y B y que satisfacen el predicado de reunión espacio-temporal.

La expresión para el costo $C(SJ)$ de acuerdo a la Ec(3) considerando el intervalo temporal de los árboles podados, vale:

$$\mathbf{C(SJ)_{3DRp}} = \sum_{n=1}^{H_j-1} N_{RB_{j,n}} \cdot \left\{ N_{RA_{i,n}} \cdot P_{i,n} + 1 + N_{RA_{i,n+1}} \cdot [\ P_{i,n+1}]^2 \right\} \quad (6)$$

donde $N_{RA_{i,n}} = [N_{R_i,n} + N_{R_i,n}(L_i - 1) \cdot a_*]$ es el número de nodos del árbol i al cabo de un período de tiempo L_i (lifespan del árbol i) a nivel n, y $N_{R_i,n} = (N_{i,n}/f)$ es la cantidad inicial de nodos en el i-ésimo árbol podado a nivel n, donde N_i es la cantidad inicial de objetos del árbol podado, cuya expresión esta dada por :

$$N_i = \frac{(N_t/K)}{(L_i - 1) \cdot a\,\% + 1}$$

donde N_t es la cantidad de objetos del 3DR-tree sin podar al cabo de la historia T; con lo cual la cantidad de objetos en un árbol podado al cabo de L_i instantes de tiempo esta dada por :

$$N_{L_i} = N_i + N_i \cdot (L_i - 1) \cdot a_* \quad (7)$$

5 Evaluación.

En esta sección comparamos la eficacia del modelo propuesto con el modelo 3DR-tree. La razón de usar 3DR-tree es que es uno de los métodos más eficientes para procesar consultas considerando intervalos temporales largos teniendo un rendimiento muy similar a MV3R-tree [15]. Consideramos conjuntos de hasta 50.000 objetos de tipo punto los que se distribuyeron uniformemente en el espacio. Consideramos una historia T1 de 100 instantes de tiempo y para T2 de 100 y 200 instantes de tiempo. La unidad de medida utilizada correspondió a la cantidad de bloques de disco (1024 bytes) leídos al ejecutar la operación de reunión espacio temporal.

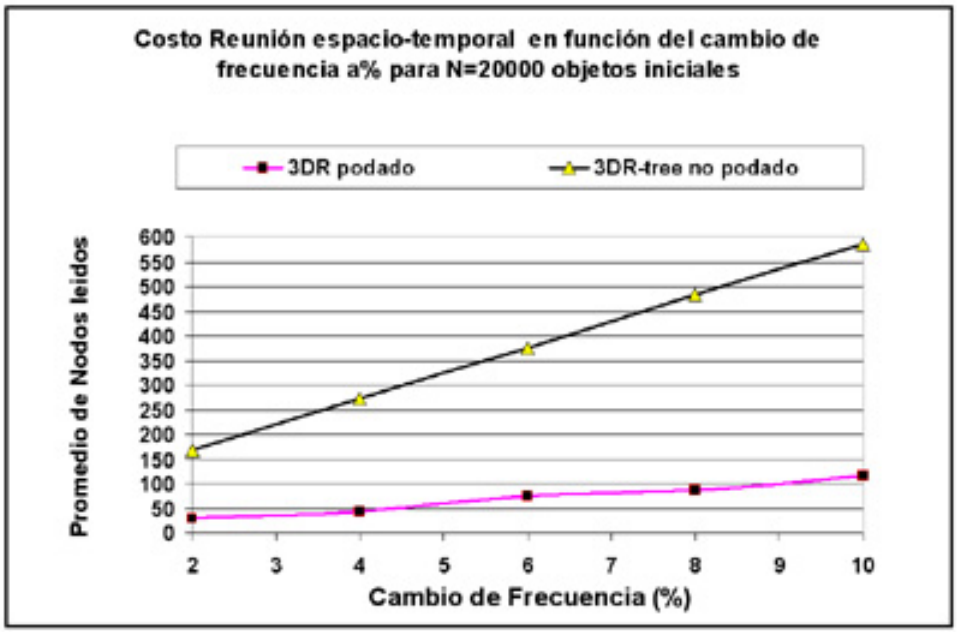

Figura 2: Costo en función del cambio de frecuencia

5.1 Comparación analítica

El rendimiento de 3DR-tree se obtuvo mediante el modelo de costo descrito en la Sección 2.3 Ec.(3).

El total de objetos a insertar en el 3DR-tree se obtuvo con la Ec.(4), y el total de objetos a insertar en los 3DR-tree-podados se obtuvo con la Ec.(7). Por otro lado consideramos una densidad inicial $D_0 = 0{,}2$, con una capacidad por nodo de 36 entradas y una capacidad promedio de llenado de 67 % . Se adoptó un valor K1=10 para la partición del la estructura principal de cada conjunto de datos y se obtuvo un intervalo temporal promedio por árbol podado de $L_i = 10\,\%$ de T . Para el conjunto S2 adoptamos un rango de K2=(5,10,20,40) obteniendo un rango de intervalos temporales promedios para cada valor de K2. Por otro lado tambien se probaron extensiones temporales en un rango de variación desde T1=100 unidades temporales, y T2=100 y 200 unidades temporales.

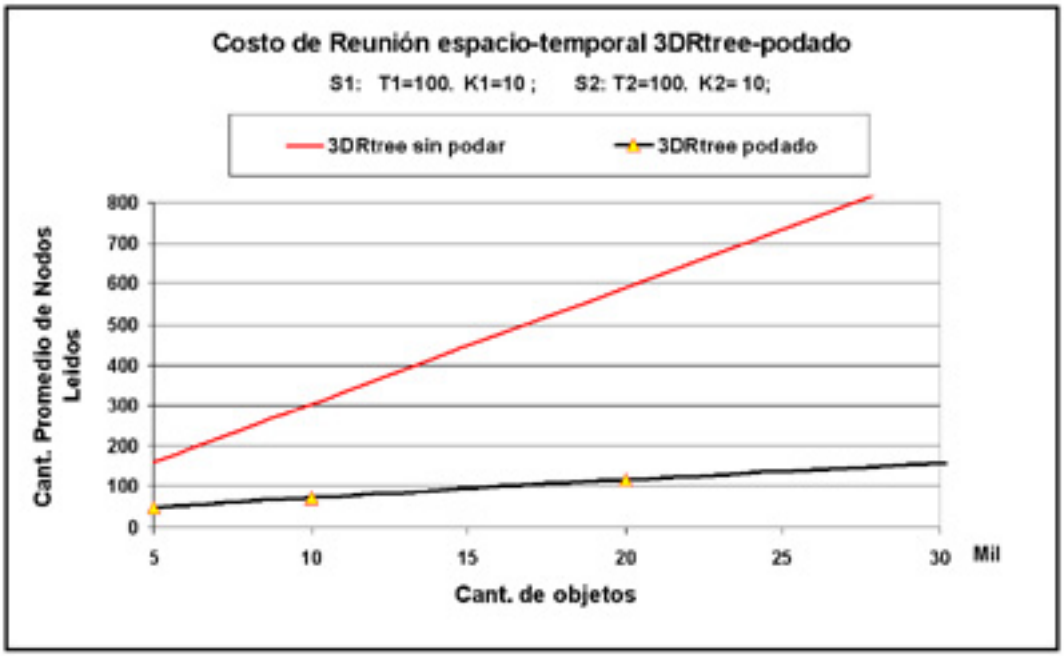

Figura 3: Costo de la reunión espacio-temporal

El gráfico de la Figura 2 se ha obtenido mediante la implementación de 3DR-tree y APR-tree. En dicha figura es posible observar el rendimiento del modelo para la reunión espacio-temporal sobre las estructuras podadas respecto de la consulta sobre una estructura 3DR-tree sin podar. Los resultados están en función del *cambio de frecuencia a %*, normalmente llamado movilidad, la cual representa el porcentaje de objetos que cambian su posición en cada instante de tiempo. La diferencia observable se debe fundamentalmente a la reducción en altura de los árboles respecto de la altura del árbol principal y la reducción del orden de complejidad de $O(K1 \cdot K2)$del *Algoritmo 1* a $O(K1 + K2)$ dada por el *Algoritmo 2*. Esta característica se ve reflejada en el tamaño promedio del nodo que al reducir la altura y la cantidad de objetos de búsqueda por árbol, la probabilidad de intersección entre los nodos crece en proporción al tamaño de los mismos, lo cual redunda en una mayor exactitud en el filtrado de las extensiones espaciales.

En la Figura 3 se puede observar el efecto de mejora que se obtiene al particionar la estructura 3DR-tree principal en diferentes cantidades de intervalos temporales pero manteniendo igual rango temporal para ambos conjuntos de datos e igual cantidad de particiones K. En cambio cuando aumentamos el rango temporal para un conjunto y variamos la cantidad de particiones, el costo es menor para valores de $K2 > K1$. Esto se observa en la Figura 4. Esta diferencia se basa en que los intervalos temporales L_i para

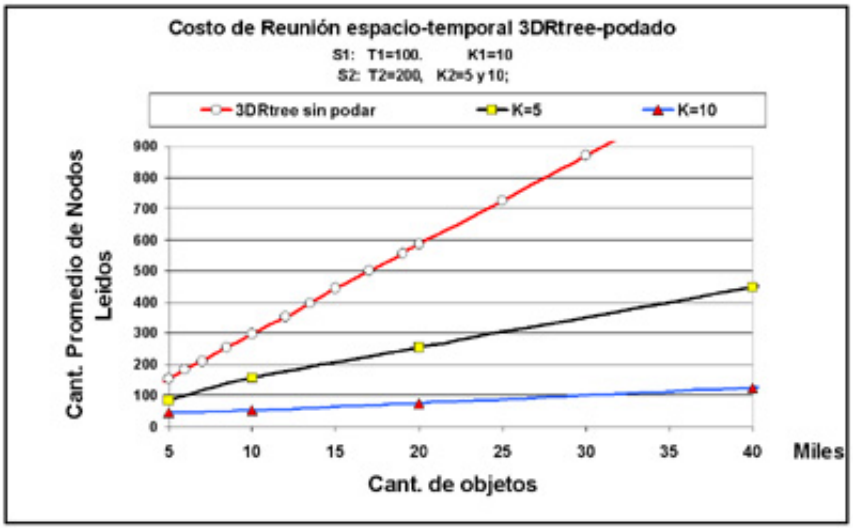

Figura 4: Costo para diferentes valores de K2 y distinto T

valores pequeños de $K2$, aumentan y este aumento genera una gran cantidad de entradas en los árboles podados lo que redunda en un aumento del costo.

6. Conclusiones.

En este trabajo se presenta un algoritmo para realizar la reunión espacio-temporal basado en la estructura 3DR-tree podada (APR-tree). La aplicación de la metodología de poda permite reducir la altura del árbol de búsqueda y por lo tanto reducir el costo de las operaciones de consulta que se realizan. Se definió un modelo de costo para nuestro algoritmo. Usando los modelos de costo de 3DR-tree y el de nuestro algoritmo, se compararon los rendimiento de la reunión espacio-temporal usando ambas estructuras.

Como se ha podido observar el costo de la reunión espacio temporal sobre los árboles podados es bastante menor que la observada para toda la estructura sin podar.

El siguiente paso en este trabajo es comparar la eficacia del modelo descrito, implementando el algoritmo usando el lenguaje c++ sobre una plataforma linux. Una vez realizado esto se determinará cuan ajustado a la realidad se comporta el modelo establecido para ambos algoritmos de reunión (join) sobre 3DR-tree.

Como trabajo futuro se pretende ahondar en el estudio de los algoritmos de reunión (join) espacio-temporales, en particular para consultas del tipo "*Hallar las colisiones de autos que ocurrieron el dia de ayer en la zona céntrica*". Este tipo de consulta evidentemente se responde con la evaluación de la operación de reunión espacio-temporal pero se ha de puntualizar que ahora la consulta especifica un sector determinado donde se consultan las intersecciones y que permite realizar un corte más específico en este tipo de consultas. Tambien enfocaremos el diseño de estructuras de datos y algoritmos que permitan resolver eficientemente este tipo de consultas y realizar trabajos de experimentación que permitan establecer la eficacia de tales modelos comparándolos contra los métodos existentes.

Referencias

[1] Brinkhoff, T., Kriegel, H.-P., and Seeger, B.Eficient processing of spatial joins using R-trees. In ACM SIGMOD Conference on Management of Data (Washington, DC, USA, 1993), ACM, pp. 237-246.

[2] Gutiérrez, G., Navarro, G., and Rodríguez, A Spatio-temporal Access Method based on Snapshots and Events.(GIS'05), November 4,2005, Bremen, Germany. Copyright 2005 ACM 1-59593-146-5/05/0011

[3] Gutiérrez, G., Navarro, G., and Rodríguez, A. SestL: An event-oriented spatio-temporal access method. Tech. Rep. TR/DCC-2006-5, Department of Computer Science, Universidad de Chile. 2006.

[4] Guttman, A. R-trees: A dynamic index structure for spatial searchinng. In ACM SIGMOD Conference on Management of Data (1984), ACM,pp. 47-57.

[5] Hadjieleftheriou, M.,Kollios,G., Tsotras, V.J. and Gunopoulos,D. Efficient indexing of spatiotemporal objects. In Extending Database Technology (2002), pp.251-268.

[6] Huang,Yun-Wu. Ning Jing, Elke A. Rundensteiner: Spatial Joins Using R-trees: Breadth-First Traversal with Global Optimizations. VLDB 1997: 396-405

[7] Hyung-ju Cho, Jun-Ki Min, Chin-Wan Chung. And Adaptive Indexing Technique Using Spatio-Temporal Query Workloads. Korean Advanced Institute of Science. Information and Software Technology, Vol 46, pp.229-241 March 2004.

[8] Kollios, G. N. Indexing Problems in Spatiotemporal Databases. PhD thesis, Polytechnic University, New York, June 2000.

[9] Nascimento,M., Silva, J., and Theodoridis, Y. Access structures for moving points. Tech. Rep.. TR-33, TIME CENTER, 1998.

[10] Pagel, B.U., Six,H., Toben, H. and Widmayer P. Towards an analysis of range query performance in spatial data structures. In PODS'93: ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Databases Systems (New York, USA 1993).ACM Press,pp.214-221.

[11] Seung-Hyun, J., Paton, W., Fernandes, A. A., and Tony, G. An experimental performance evaluation of spatio-temporal join strategies. Transactions in GIS (March 2005),129-156.

[12] Tao, Y., and Papadias, D. Eficient historical R-Tree. In SSDBM International Conference on Scientific and Statical Database Management (2001), pp. 223-232.

[13] Tao, Y., Papadias, D., and Zhang, J. Cost models for overlapping and multiversion structures. ACM Trans. Database Syst. 27, 3 (2002), 299-342.

[14] Tao, Y., Papadias, D. Historical Spatio-Temporal Aggregation. ACM Transactions on Database Systems, Vol.23, No.1, January 2005,pp:61-102.

[15] Tao, Y., Papadias, D. MV3R-tree: A spatio-temporal access method for timestamp and interval queries. In The VLDB *Journal* (2001), pp. 431-440.

[16] Theodoridis, Y., and Sellis, T. A model for the prediction of R-tree performance. In PODS '96: Proceedings of the fifteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems (New York, NY, USA,1996), ACM Press, pp. 161-171.

[17] Theodoridis, Y.,Vazirgiannis, M. and Sellis, T. Spatio-Temporal Indexing for Large Multimedia Applications. In Proc. of the IEEE Conference on Multimedia Computing and systems, ICMCS, june 1996.

[18] Theodoridis, Y., Stefanakis, E., and Sellis, T. Eficient cost models for spatial queries using R-Trees. IEEE Transactions on Knowledge and Data Engineering 12, 1 (2000), 19-32.

[19] Theodoridis, Y., Stefanakis, E., and Sellis, T. An Eficient cost Model for spatial Joins Using R-Trees. IEEE Transactions on Knowledge and Data Engineering (1997).

[20] Varzirgiannis M., Theodoridis, Y., Sellis, T. Spatio-Temporal Composition and indexing for Large Multimedia Applications. ACM Multimedia Systems, 6(5), 1998.