Procesamiento de consultas espacio-temporales sobre un índice eficiente

María G. Dorzán, Juan G. Gómez Barroso, Edilma O. Gagliardi
Facultad de Ciencias Físico, Matemáticas y Naturales,
Departamento de Informática, Universidad Nacional de San Luis,
San Luis, D5700HHW, Argentina
{ mgdorzan, jggomez, oli }@unsl.edu.ar
y
Gilberto A. Gutiérrez Retamal
Facultad de Ciencias Empresariales,
Departamento de Auditoría e Informática, Universidad del Bío-Bío
Chillán, Chile
ggutierr@ubiobio.cl

RESUMEN

Las Bases de Datos Espacio-Temporales permiten almacenar y consultar los cambios de posición, forma y/o tamaño de objetos a lo largo del tiempo. Para responder muchas consultas que involucran predicados espacio-temporales, es fundamental contar con métodos de acceso que permitan seleccionar los objetos que forman parte de la respuesta en forma eficiente. Entre los diferentes tipos de consultas espacio-temporales, los más considerados, en general, son *TimeSlice*, *Eventos*, *Intervalo* y *Trayectoria*. En la literatura afín, podemos encontrar una gran variedad de métodos, los cuales intentan optimizar el desempeño de las consultas, pero siempre apuntando a un subconjunto de las antes mencionadas. Como objetivo de nuestra investigación, nos propusimos buscar un método que fuera propicio para resolver estos cuatro tipos de consultas, sin aumentar la complejidad espacio-temporal. Por ello, nos dedicamos al estudio y diseño de una estructura de datos, la que llamamos *D*R-Tree*, y de sus respectivos algoritmos de consulta, de la cual mediante evaluaciones experimentales ya hemos obtenido resultados, logrando obtener un mejor desempeño respecto de una estructura similar de comparación.

Palabras claves: bases de datos espacio-temporales, métodos de acceso espacio-temporal.

1. INTRODUCCIÓN

La mayoría de las bases de datos se conciben para describir algún aspecto del mundo real sobre un período de tiempo dado. Cualquier valor que se almacene describe el pasado; es decir, que un hecho definido en un registro de una base de datos es una afirmación sobre el estado del mundo en un momento dado. Existen muchas aplicaciones que requieren manejar objetos espacio-temporales, es decir, objetos cuya posición espacial y/o forma cambia en distintos instantes de tiempo. Tales cambios deben ser manejados por un Sistema de Administración de Bases Datos que debe proveer, entre otros servicios, métodos de acceso para procesar en forma eficiente consultas cuyos predicados contemplan relaciones espacio-temporales. Para ello es fundamental contar con métodos de acceso que permitan seleccionar los objetos que forman parte de la respuesta, en lugar de recorrer la base de datos completa.

Los sistemas de bases de datos que administran objetos espaciales y temporales han recibido mayor interés en los últimos años. Las bases de datos que guardan objetos espaciales que cambian su tamaño y/o su posición a través del tiempo se llaman Bases de Datos Espacio-Temporales (BDET's). Éstas permiten almacenar y consultar los cambios de posición, forma y/o tamaño de objetos a lo largo del tiempo.

En estas bases de datos, el presente, el pasado como así también la anticipación de la posición y extensión futura de los objetos, son de

frecuente interés en investigación y en la aplicación. Las aplicaciones que tratan con los objetos espacio-temporales incluyen cambios globales (datos climáticos), transporte (supervisión de tráfico), telecomunicaciones (telefonía celular), aspectos sociales (demografía, salud), multimedia (películas animadas) e información geográfica (cambios de límites en terrenos), entre otras.

Actualmente, las BDET's en conjunto representan un tópico de estudio reciente y de mucho interés debido a la estrecha relación existente entre espacio y tiempo en una gran cantidad de aplicaciones en donde se debe modelar datos con componentes tanto espaciales como temporales. Por ello, es necesario que una BDET sea capaz de representar modelos muy cercanos al mundo real, con todo el dinamismo que éste implica, y administrar objetos que, básicamente, cambian su ubicación y/o forma a través del tiempo.

En la siguiente sección detallamos los distintos métodos en los que se basa nuestra propuesta. En la sección 3 presentamos el D*R-Tree, incluyendo una descripción de la estructura de datos. En la sección 4 explicamos en forma general el procesamiento de las consultas soportadas por nuestra estructura. En la sección 5 presentaremos el contexto de experimentación detallando las restricciones que se plantearon, como así también las conclusiones de las comparaciones y pruebas realizadas. Finalmente se describe el trabajo realizado hasta el momento y los futuros avances del mismo.

2. ANTECEDENTES

En particular, el método de acceso espacio-temporal elegido se basa en la idea expuesta en [3]. El método propuesto pretende mantener un equilibrio entre el espacio de disco utilizado por la estructura y el tiempo de acceso empleado en responder los distintos tipos de consulta. A continuación detallamos las estructuras subyacentes.

Gran parte de los métodos de acceso espacio-temporal propuestos hoy en día usan de base a R-Tree. [5, 6]. Esta es una estructura de datos jerárquica, basada en B-Tree [1], balanceado por altura. En un R-Tree se almacena su *MBR* (Minimum Bounding Rectangle), es decir el menor rectángulo que contiene el objeto en cuestión y no los objetos espaciales en forma directa. Cada nodo en R-Tree corresponde al MBR que contiene a sus hijos. Los nodos hoja de R-Tree contienen punteros a los objetos en la base de datos. Los nodos hoja del R-Tree contienen entradas de la forma <*MBR*; *O*> donde *MBR* es el menor rectángulo que contiene al objeto apuntado por *O*. Los nodos internos contienen entradas de la forma < *MBR*; *P* > donde *P* es un puntero a un hijo del nodo y *MBR* contiene a todos los *MBR* del nodo apuntado por *P*. R-Tree cumple con las siguientes propiedades: cada nodo (a excepción, quizás, del nodo raíz) contiene entre *m* y *M* entradas donde *m* <= *M*/2 y *M* es el número máximo de entradas por nodo; el nodo raíz tiene al menos dos hijos a menos que sea una hoja; y todas las hojas están al mismo nivel.

Por otro lado, *SEST-Index* es un método de acceso espacio-temporal de tipo histórico [3]. La estructura tiene puntos de referencia para ciertos instantes de tiempo, tal que los objetos *vivos* en esos instantes se almacenan en una estructura de datos R-Tree. Las modificaciones, producto del movimiento de los objetos entre puntos de referencia temporal consecutivos se mantienen en una lista de bloques denominada bitácora. Ésta se encuentra ordenada de acuerdo al tiempo y permite reconstruir cualquier estado de la base de datos entre dos puntos de referencia consecutivos. La bitácora es una lista de bloques. Las entradas en los bloques son tuplas con la siguiente estructura: < *t*; *Mbr anterior*; *Mbr actual*; *Oid* > donde *t:* tiempo en que se produjo la modificación; *Mbr anterior*: aproximación al objeto en el tiempo *t-1*; *Mbr actual*: aproximación al objeto en el tiempo *t*; *Oid*: identificador del objeto. Las entradas en la bitácora se encuentran ordenadas por el atributo *t*. Se proporciona un parámetro *d* correspondiente a un valor de umbral del tamaño de las bitácoras, así la estructura puede decidir en qué momento debe crearse un nuevo punto de referencia. Éste se crea cada vez que en la bitácora actual sea imposible almacenar los cambios producidos en un instante *t*, ya que al hacerlo el número de bloques de ésta superaría el valor *d.*

Este método está diseñado para responder eficientemente las consultas de tipo *TimeSlice*, *Eventos* e *Intervalo.*

3. D*R-TREE

Nuestro trabajo de investigación se dedica al estudio y análisis del diseño de estructuras de datos y algoritmos que permitan resolver los distintos tipos de consulta en diversos métodos de acceso espacio-temporal.

En particular, se ha abordado el método SEST-Index que apunta a la resolución de los tipos de consultas *Timeslice*, *Intervalo* y *Eventos*. Hemos propuesto extensiones a las estructuras de datos subyacentes de forma tal que permitan resolver el tipo de consultas *Trayectoria*. A continuación damos un resumen de la estructura propuesta.

D*R-Tree [2] es un método de acceso espacio-temporal que se basa en la idea expuesta en [3]. Mantiene puntos de referencia para ciertos instantes de tiempo en la base de datos, tal que para esos instantes los objetos se almacenan en una estructura de datos R-Tree. Las modificaciones ocurridas entre los puntos de referencia de tiempo consecutivos se mantienen en una estructura llamada bitácora.

Nuestra propuesta tiene las siguientes características:

• Se utiliza un Índice para almacenar los instantes de tiempo que se establezcan como puntos de referencia. Es una lista secuencial que almacena tuplas de la forma <*t*, *R*, *B*> donde: *t*: tiempo del punto de referencia; *R*: mantiene la raíz del R-Tree asociado a *t*; y *B*: mantiene la cabecera de la bitácora asociada a *t*.

• Los R-Tree's son utilizados para almacenar la ubicación espacial de los objetos. Cada R-Tree está asociado a un punto de referencia de tiempo, según el índice descripto en el punto anterior.

• Se usan Bitácoras para almacenar los movimientos realizados en instantes intermedios entre puntos de referencia almacenados en el índice de instantes; además, se guardan las referencias a los movimientos anteriores inmediatos por cada objeto. Es una lista secuencial ordenada por el tiempo que almacena tuplas de la forma <*Oid*, *t*, *Ref_ant*, *Mbr_act*> donde: *Oid*: es el identificador del objeto; *t*: es el tiempo en que se produjo el movimiento; *Ref_ant*: mantiene una referencia a la ubicación del movimiento anterior del objeto en la bitácora correspondiente; *Mbr_act*: mantiene la posición actual del objeto por medio de un par de coordenadas (*x*, *y*), donde *x* e *y* son puntos que corresponden a la diagonal del MBR.

• Se agrega un Índice de trayectorias para acceder a los últimos registros de movimiento en cada bitácora por cada objeto. Con ello se posibilita la entrada directa a las bitácoras involucradas en la trayectoria de los objetos. Es un direccionamiento directo que fuerza dependencia donde en la cabecera se mantienen todos los objetos que son accedidos por *Oid* (direccionamiento directo) y se mantiene una referencia a la cabecera de una lista vinculada. Cada nodo de esta lista hace referencia a una bitácora distinta y almacena información correspondiente a la última entrada en dicha bitácora para el objeto correspondiente al índice de la lista de primer nivel. Cada bitácora tendrá, a lo sumo, una única entrada en la lista.

• Se utiliza un Índice de tiempos para facilitar el procesamiento de la consulta *Eventos*. Es una estructura que permite encontrar cualquier instante de tiempo en las distintas bitácoras. Puede ser implementado mediante un arreglo de tuplas de la forma < *t*, *Puntero b* >, donde *t* es el tiempo y *b* es el puntero a la posición de la bitácora que contiene la primer entrada del tiempo *t*.

Para realizar la actualización de la estructura suponemos que disponemos de una imagen de los objetos en el espacio. Se creará un nuevo punto de referencia cuando, al querer ingresar un nuevo conjunto de movimientos en la bitácora, ésta no dispone de espacio suficiente para albergarlos, dando origen a una nueva entrada en el Índice de instantes y un nuevo R-Tree, el que se construye de acuerdo a las diferencias entre el R-Tree y la bitácora del último punto de referencia almacenado. Estas diferencias se refieren a los cambios que se deben hacer en la imagen inmediatamente anterior para alcanzar la imagen actual.

Para conocer más detalles acerca de la estructura se puede consultar [2].

Así, las principales diferencias con respecto al método original, son que el D*R-Tree posee una estructura adicional, un *Índice de Trayectorias*, que nos brinda una ventaja, permitiéndonos guardar información para poder recuperar la trayectoria de los objetos en forma eficiente. Esto se complementa con la reestructuración de la información almacenada en las bitácoras.

4. CONSULTAS

El tipo de *Consultas de selección* es de la forma "encontrar todos los objetos que se encontraban en un área o punto específico, durante un intervalo o instante de tiempo específico". Es el tipo de consulta más frecuente en una base de datos espacio-temporal. Las consultas típicas son:

TimeSlice: el resultado de la consulta consiste de los objetos que se encuentran en una determinada área en un instante de tiempo dado.

Intervalo: a diferencia de la consulta *TimeSlice* se tiene en cuenta un intervalo y no un instante de tiempo.

Otro tipo es la consulta *Eventos*, en la cual se recuperan eventos que sucedieron en una región en un instante dado. Estos eventos pueden ser objetos que han *aparecido* o *desaparecido* en una región en un cierto instante.

La consulta *Trayectoria*, es la que recupera el conjunto de posiciones espaciales en las que un objeto ha permanecido en un intervalo de tiempo dado.

Con nuestra estructura se logran responder los cuatro tipos de consulta nombrados anteriormente, con un buen desempeño. En líneas generales, cada consulta se trata de la siguiente manera:

• *TimeSlice*(*R*, *t*): Para procesar una consulta de este tipo primero ubicamos el punto de referencia adecuado, de acuerdo al instante de tiempo especificado en la consulta (*t*). Luego se hace una búsqueda espacial en el R-Tree del punto de referencia encontrado. El conjunto de objetos obtenidos por esta consulta es actualizado con las entradas de la correspondiente bitácora.

• *Intervalo*(*R*, t_o, t_f): El procesamiento de este tipo de consulta es muy simple. Primero recuperamos el estado de los objetos en el límite inferior del intervalo (t_o). Para lograr esto usamos el método propuesto para *TimeSlice*. Una vez establecido este conjunto recorremos todas las entradas de las bitácoras cuyo atributo *t* es menor o igual al límite superior (t_f) especificado en el intervalo, actualizando el conjunto.

• *Eventos*(*R*, *t*): Para responder este tipo de consultas primero ubicamos la bitácora donde están los cambios del tiempo consultado. Recorremos todas las entradas con tiempo igual a *t* para indicar cuántos objetos entraron o salieron del área de consulta.

• *Trayectoria*(*Oid*, t_i, t_k): Para procesar una consulta de tipo *Trayectoria* primero buscamos el objeto en la estructura Índice de trayectoria y recorremos la lista secuencialmente considerando el límite superior del intervalo de consulta (t_k), es decir que la búsqueda se detiene en el nodo donde el tiempo asociado a la bitácora apuntada sea el menor más cercano a t_k. Luego comenzamos a recorrer las referencias dentro de las bitácoras obteniendo las posiciones correspondientes, teniendo en cuenta el intervalo de entrada.

En general, la mayoría de los métodos existentes no apuntan a responder todos los tipos de consultas más comunes, sino que sus propuestas se basan en estructuras que sólo dan soporte a un subconjunto de ellos. Por consiguiente, y reconociendo la importancia de conocer la trayectoria de un objeto determinado, D*R-Tree intenta aprovechar las bondades del resto de los métodos, respondiendo eficientemente los principales tipos de consultas e incluyendo las consultas de tipo *Trayectoria*.

5. EVALUACIÓN EXPERIMENTAL

En las pruebas se comparó D*R-Tree con SEST-Index [3] respecto de la utilización de almacenamiento y tiempo de acceso a disco en las consultas de tipo *TimeSlice*, *Intervalo*, *Eventos* y *Trayectoria*.

Se analizaron ambos índices utilizando lotes con 1000, 3000 y 5000 objetos (puntos en el plano), moviéndose en el espacio durante 50 instantes de tiempo consecutivos, con un porcentaje de movilidad de 1, 3, 5, 7, 9, 11, 13 y 15 por ciento por instante, teniendo en cuenta la cantidad de bloques por bitácora. Estos puntos se distribuyeron uniformemente dentro del cuadrado unitario en el instante de tiempo inicial. Luego, se movieron aleatoriamente durante los próximos 50 instantes de tiempo hasta llegar al instante final. La cantidad de objetos que se mueven por instante de tiempo no sobrepasó la capacidad de cada bitácora. Para medir la utilización de disco se contó el número de bloques usado por el índice con el total de datos almacenados. Para medir el tiempo de acceso en las consultas se calculó el promedio de bloques leídos tras realizar 100

consultas aleatorias.

Los algoritmos de las consultas consideraron las siguientes condiciones: la cantidad fija de objetos en movimiento; porcentaje de movilidad alto; los objetos informan los cambios de posición; un espacio acotado y conocido; se utilizaron distribuciones de probabilidades para generar los movimientos de los objetos y las consultas son rectángulos ortogonales.

Los resultados obtenidos se observó que tanto D*R-Tree como SEST-Index se comportan de manera similar con respecto al espacio utilizado para el almacenamiento de la estructura.

Con respecto a la cantidad de bloques leídos en el procesamiento de las consultas *TimeSlice*, *Intervalo* y *Eventos* se observó que no hay significativas variaciones con respecto a SEST-Index, ya que ambas estructuras trabajan de forma análoga para resolverlas.

D*R-Tree siempre consulta un menor número de bloques que SEST-Index en consultas de tipo *Trayectoria*. Como SEST-Index no responde a esta consulta en forma eficiente por no estar diseñada para este fin, se deben recorrer secuencialmente las posiciones de las bitácoras correspondientes al intervalo de consulta para retornar la trayectoria del objeto de consulta.

Cabe destacar que SEST-Index puede no retornar un resultado ya que en intervalos de tiempo donde no se han registrado movimientos del objeto, en el recorrido secuencial no se encontrará información del objeto de consulta.

6. CONCLUSIONES Y TRABAJO FUTURO

En esta propuesta de investigación, proponemos un nuevo índice, D*R-Tree, para poder almacenar y responder eficientemente consultas de tipo espacio-temporal. Está basado en el modelo propuesto en [3], con el fin de aprovechar sus ventajas respecto de una buena utilización de espacio en disco y un buen tiempo para responder consultas. Hemos evaluado experimentalmente nuestro método contra el propuesto en [3], dado que fue utilizado como método base para el diseño y la comparación posterior, obteniendo un mejor desempeño y ampliando el conjunto de consultas espacio-temporales original. Nuestro trabajo futuro pretende continuar en esta dirección, dedicándonos al estudio y análisis del diseño de estructuras de datos y algoritmos que permitan resolver distintos tipos de consulta espacio-temporales.

Este trabajo está enmarcado dentro de la Línea de investigación Geometría Computacional y Bases de Datos Espacio-Temporales, perteneciente al Proyecto Tecnologías Avanzadas de Bases de Datos 22/F314, Departamento de Informática, Universidad Nacional de San Luis; en el Proyecto AL06_PF_013 Geometría Computacional, subvencionado por la Universidad Politécnica de Madrid; y en el marco de la Red Iberoamericana de Tecnologías del Software (RITOS2), subvencionado por CYTED. Por todo ello, se ha establecido un grupo de interés en el tema conformado por docentes investigadores y alumnos avanzados de la Universidad Nacional de San Luis.

7. REFERENCIAS

[1] Bayer, R. and McCreight, E. "Organization and Maintenance of Large Ordered Indexes." Acta Informática 1, 173-189, (1972).

[2] Gagliardi E., Dorzán M., Gómez Barroso J. D*R-Tree: un método de acceso espacio-temporal. XI Congreso Argentino de Ciencias de la Computación ISBN 950-698-166-3 (2005).

[3] Gutiérrez Gilberto. A Spatiotemporal Access Method based on Snapshots and Events. ACM GIS'05, Bremen, Germany. (November 4, 2005).

[4] Güting, R.H., An introduction to Spacial Database System. VLDB Journal (1994).

[5] Guttman A. R-Trees: A dynamic index structure for spatial searching. In ACM SIGMOD Conference on Management of Data, pages 47-57, Boston, ACM. (1984).

[6] Manolopoulos Y., Nanopoulos A., Papadopoulos A. y Theodoridis Y. R-trees have grown everywhere. (2003).

[7] Wang X., Zhou X. y Lu S. Spatiotemporal data modeling and management: a survey. Proceedings 36th International Conference on Technology of Object-Oriented Languages and System, pp. 202-211. (2000).