Procesamiento de Consultas Métrico-Temporales

Anabella De Battista , Andrés Pascal, Alejandra Díaz, Pablo Gancharov, Adrian Planas

Departamento de Sistemas de Información, Fac. Reg. Concepción del Uruguay

Universidad Tecnológica Nacional, Entre Ríos, Argentina

{debattistaa, pascalj}@frcu.utn.edu.ar

{alejandraalciradiaz,pablogancharov,pladnic}@gmail.com

Norma Edith Herrera

Departamento de Informática

Universidad Nacional de San Luis

San Luis, Argentina

nherrera@unsl.edu.ar

Gilberto Gutierrez

Facultad de Ciencias Empresariales

Universidad del Bio-Bio

Chillán, Chile

ggutierr@ubiobio.cl

Resumen

Las bases de datos actuales permiten almacenar datos no estructurados tales como imágenes, sonido, video, datos geométricos, etc. Las tecnologías tradicionales de bases de datos no son aplicables en este ámbito. El modelo de bases de datos métrico-temporal permite abordar aquellas situaciones en las que resulta necesario realizar búsquedas por similitud sobre datos no estructurados pero teniendo en cuenta también la componente temporal. En este modelo se combinan los espacios métricos con las bases de datos temporales, permitiendo así procesar consultas por similitud restringidas a un intervalo o a un instante de tiempo. Nuestro área de investigación es el diseño de índices eficientes para este tipo de bases de datos.

Palabras Claves: Espacios Métricos, Bases de Datos Temporales, Bases de Datos Métrico-Temporales, Índices

1. Contexto

El presente trabajo se desarrolla en el ámbito del Grupo de Investigación en Bases de Datos (Proy. Nro 25-D040) perteneciente al Departamento de Sistemas de la Universidad Tecnológica Nacional, Facultad Regional Concepción del Uruguay, cuyo objetivo principal es el estudio de métodos de acceso, procesamiento de consultas y aplicaciones de bases de datos no tradicionales.

2. Introducción

Las bases de datos clásicas se organizan bajo el concepto de búsqueda exacta sobre datos estructurados. Esto significa que la información se organiza en registros los cuales se dividen en campos que contienen valores completamente comparables. Una búsqueda en la base retorna todos aquellos registros cuyos campos coinciden con los aportados en la consulta (búsqueda exacta). Una característica importante de las bases de datos clásicas es que cap-

turan sólo un estado de la realidad modelizada, usualmente el más reciente. Por medio de las transacciones, la base de datos evoluciona de un estado al siguiente descartando el estado previo.

Actualmente las bases de datos han incluido la capacidad de almacenar datos no estructurados tales como imágenes, sonido, texto, video, datos geométricos, etc. La problemática de almacenamiento y búsqueda en estos tipos de base de datos difiere de las bases de datos clásicas en varios aspectos: los datos no son estructurados por lo tanto no es posible organizarlos en registros y campos; la búsqueda exacta carece de interés; resulta de interés mantener todos los estados de la base de datos y no sólo el más reciente a fin de poder consultar el instante o intervalo de tiempo de vigencia de los objetos.

Es en este contexto donde surgen nuevos modelos de bases de datos capaces de cubrir eficaz y eficientemente las necesidades de almacenamiento y búsqueda de estas aplicaciones.

El modelo de ***espacios métricos*** [3, 1, 6], permite trabajar con objetos no estructurados y realizar búsquedas por similitud sobre los mismos. Un espacio métrico es un par (U, d) donde U es un universo de objetos y $d : U \times U \rightarrow R^+$ es una función de distancia definida entre los elementos de U que mide la similitud entre ellos. Una de las consultas típicas en este modelo es la búsqueda por rango, denotado por $(q, r)_d$, que consiste en recuperar los objetos de la base de datos que se encuentren como máximo a distancia r de un elemento q dado.

El modelo de ***bases de datos temporales*** [11, 5] incorpora al tiempo como una dimensión, por lo que permite asociar tiempos a los datos almacenados y consultar por los objetos vigentes en un intervalo o en un instante de tiempo dado.

Existen aplicaciones donde resulta de interés realizar búsquedas por similitud teniendo en cuenta también la componente temporal. Es en este ámbito donde surge el ***modelo métrico temporal***. En este modelo se puede trabajar con objetos no estructurados con tiempos de vigencia asociados y realizar consultas por similitud y por tiempo en forma simultánea. Formalmente un *Espacio Métrico-Temporal* es un par (U, d), donde $U = O \times N \times N$, y la función d es de la forma $d : O \times O \rightarrow R^+$. Cada elemento $u \in U$ es una triupla (obj, t_i, t_f), donde obj es un objeto (por ejemplo, una imagen, sonido, cadena, etc) y $[t_i, t_f]$ es el intervalo de vigencia de obj. La función de distancia d, que mide la similitud entre dos objetos, cumple con las propiedades de una métrica (positividad, simetría y desigualdad triangular). Como un ejemplo de aplicación podemos mencionar una base de datos de rostros de delincuentes donde cada foto tiene una intervalo de vigencia asociado, que representa el intervalo de tiempo en que el delincuente tenía el aspecto representado en esa foto; en este caso sería de interés, dada una foto y un intervalo de tiempo, poder recuperar de la base todos aquellos rostros parecidos al dado en el intervalo de tiempo especificado. Formalmente una *consulta métrico-temporal* se define como una 4-upla $(q, r, t_{iq}, t_{fq})_d$, tal que $(q, r, t_{iq}, t_{fq})_d = \{o/(o, t_{io}, t_{fo}) \in X \wedge d(q, o) \leq r \wedge (t_{io} \leq t_{fq}) \wedge (t_{iq} \leq t_{fo})\}$

3. Índices en BDMT

Varios índices métrico-temporales se han propuesto en este ámbito; todos ellos han tomado como base el Fixed Height Queries Tree[1], un índice para espacios métricos.

FHQT-Temporal[9]. Este índice es una adaptación del Fixed Height Queries Tree (FHQT) en la que se agrega un intervalo de tiempo en cada nodo del árbol. Este intervalo representa el período máximo de vigencia para todos los objetos del subárbol cuya raíz es dicho nodo. En cada nodo hoja, este intervalo es el período total de vigencia de los objetos que contiene. Para cada nodo interior, el intervalo se calcula

tomando el tiempo inicial mínimo y el tiempo final máximo de sus hijos. Cuando se realiza una consulta métrico-temporal se procede de la siguiente manera: en cada nivel del árbol se filtran los subárboles hijos por el intervalo de tiempo de la consulta y luego de acuerdo a la distancia entre la consulta y el pivote. Al llegar al último nivel, se realiza una búsqueda secuencial sobre las hojas que no fueron descartadas seleccionando los objetos que cumplen con las condiciones temporales y de similitud.

Historical-FHQT [2]. Consiste en una lista de instantes válidos donde cada uno contiene un FHQT correspondiente a todos los objetos vigentes en dicho instante. Los FHQT de cada instante de tiempo tienen distintas profundidades en función de la cantidad de elementos que deban indexar pero todos trabajan sobre sobre el mismo conjunto base de pivotes; esto significa que si en el instante i necesito k_i pivotes y en el instante j necesito k_j pivotes con $k_i < k_j$, entonces los primeros k_i pivotes de ambos instantes son iguales. Las consultas métrico-temporales se efectúan de la siguiente manera: en primer lugar se seleccionan los instantes incluidos en el intervalo de consulta. Luego se realizan consultas por similitud usando cada uno de los FHQT correspondientes, y finalmente se unen los conjuntos resultantes.

Event-FHQT [7]. Consiste en una lista de intervalos de tiempo válido consecutivos de tamaño fijo. Cada intervalo contiene un FHQT que indexa los objetos vigentes en el primer instante de dicho intervalo. Las hojas del FHQT contienen listas de eventos que indican los cambios que se produjeron entre dos intervalos. Presenta una ventaja respecto del Historical-FHQT, y es que no necesita duplicar los objetos vigentes en más de un instante de tiempo. Ante una consulta métrico-temporal primero se filtran los intervalos de la lista que se intersectan con el intervalo de consulta, luego por cada intervalo se realiza la consulta por similitud sobre el FHQT, se recorren las listas de eventos para determinar que objetos cumplen la restricción temporal de la consulta, por último se unen los conjuntos resultantes y se compara cada elemento de ese conjunto con la consulta.

Pivot-FHQT [8]. Se propone como una mejora del *Historical-FHQT*, en el cual se usan conjuntos disjuntos de pivotes para FHQT consecutivos. Si bien esto aumenta la cantidad de evaluaciones de distancias al momento de calcular la firma de la query q, también aumenta la probabilidad de disminuir la cantidad de falsos positivos en la lista de candidatos con los que deberá compararse q. Este índice mostró ser más competitivo que el H-FHQT, en la totalidad de las consultas por intervalo y en la mayoría de las consultas instantáneas

.

4. Líneas de Investigación y Desarrollo

Nuestra principal línea de estudio e investigación es el desarrollo de índices métrico-temporales eficientes. Damos a continuación una descripción de las líneas de investigación que actualmente estamos desarrollando.

Índices en Memoria Secundaria

Los índices desarrollados hasta el momento se basan en el supuesto de que la memoria principal tiene capacidad suficiente como para mantener tanto el índice como la base de datos. Si esto no es así, la cantidad de accesos a memoria secundaria realizados durante el proceso de búsqueda es un factor crítico en la performance del índice [12].

En [4] se presenta el *Compact Pat Tree*, un índice en memoria secundaria para búsquedas de patrones en texto. Básicamente este índice

consiste en una representación compacta de un árbol binario (un Pat-Tree) en disco.

La técnica de paginación consiste en particionar el árbol en componentes conexas, llamadas *partes*, cada una de las cuales se almacena en una página de disco El algoritmo propuesto por los autores para particionar este árbol binario en partes es un algoritmo greedy que procede en forma bottom-up tratando de condensar en una única parte un nodo con uno o los dos subárboles que dependen de él. En este proceso de particionado las decisiones se toman en base a la profundidad de cada nodo involucrado, donde la profundidad de un nodo a es la cantidad máxima de páginas que se deben acceder en un camino que comience en a y termine en una hoja del subárbol con raíz a.

Para particionar un árbol se comienza asignando cada hoja a una parte con profundidad 1 y luego, en forma bottom-up, se van procesando cada uno de los nodos del árbol según las siguientes reglas:

a) si ambos hijos tienen la misma profundidad h y las partes que contienen a los hijos y el nodo corriente entran en una página de disco, se unen ambas partes junto con el nodo corriente y se establece la profundidad del nodo corriente y de esta nueva parte en h.

b) si ambos hijos tienen la misma profundidad h y las partes que contienen a los hijos y el nodo corriente no entran en una página de disco, se cierran las partes de los hijos y se crea una nueva parte para el nodo corriente con profundidad $h + 1$.

c) si los hijos tienen profundidades h y k con $h < k$ y el nodo corriente y el hijo de mayor profundidad (k) entran en una página de disco, se cierra la parte del hijo con menor profundidad (h), se une el nodo corriente con la parte del hijo de mayor profundidad y se establece la profundidad de esta nueva parte en k.

d) si los hijos tienen profundidades h y k con $h < k$ y el nodo corriente y el hijo de mayor profundidad (k) no entran en una página de disco, se cierran las partes de ambos hijos y se crea una nueva parte para el nodo corriente con profundidad $k + 1$.

En [10] se presenta una modificación de esta técnica de paginación para árboles r-arios, la cual es fácilmente adaptable para la paginación de los índices presentados en el punto anterior.

En el caso particular del FHQT-Temporal la aplicación de esta técnica requiere de pocas modificaciones. Cabe señalar que cada vez que una subárbol del FHQT-Temporal se pagina, se reemplaza por una hoja que contiene el número de página donde se almacenó dicho subárbol. Esto implica que hay hojas de dos tipos: la hojas reales de FHQT-Temporal original y la hojas que representan subárboles almacenados en otras páginas de disco. Por lo tanto se hace necesario agregar una marca a cada hoja que permita distinguir su tipo, lo que implica un pequeño overhead en espacio.

Otras Consultas Métrico Temporales

En las aplicaciones en las que el modelo métrico-temporal tiene interés, existen otros tipos de consultas que resultan interesantes, tales como: búsqueda del vecino o de los k-vecinos más cercanos en un intervalo de tiempo, consultas instantáneas puras como por ejemplo *encontrar todas las fotografías vigentes en un instante de tiempo* y consultas por clave, por ejemplo *encontrar las diferentes fotografías de una persona a lo largo del tiempo*. Actualmente estamos diseñando los algoritmos para cada una de estas consultas en los índices en memoria principal presentados en la sección anterior.

5. Resultados Esperados

Se espera contar con un índice métrico-temporal en memoria secundaria que sea eficiente para resolver las consultas planteadas tanto en los tiempos de respuesta como en el espacio ocupado por el mismo.

6. Formación de Recursos Humanos

El trabajo desarrollado hasta el momento forma parte del desarrollo de dos Tesis de Maestría en Ciencias de la Computación, una de ellas fue defendida y aprobada en marzo del 2009. Uno de los integrantes del grupo está desarrollando su Tesis Doctoral sobre la temática de indexación en memoria secundaria de bases de datos textuales, que está íntimamente relacionado a la temática de estudio de este grupo. El grupo cuenta además con tres alumnos becarios que están iniciando su formación en estas temáticas.

Referencias

[1] R. Baeza-Yates, W. Cunto, U. Manber, and S. Wu. Proximity matching using fixed-queries trees. In *Proc. 5th Combinatorial Pattern Matching (CPM'94)*, LNCS 807, pages 198–212, 1994.

[2] De Battista, A. Pascal, G. Gutierrez, and N. Herrera. Un nuevo índice métrico-temporal: el historical fhqt. In *Actas del XIII Congreso Argentino de Ciencias de la Computación*, Corrientes, Agentina, 2007.

[3] E. Chávez, G. Navarro, R. Baeza-Yates, and J.L. Marroquín. Searching in metric spaces. *ACM Computing Surveys*, 33(3):273–321, September 2001.

[4] D. Clark and I. Munro. Efficient suffix tree on secondary storage. In *Proc. 7th ACM-SIAM Symposium on Discrete Algorithms*, pages 383–391, 1996.

[5] C. S. Jensen. A consensus glossary of temporal database concepts. *ACM SIGMOD Record*, 23(1):52–54, 1994.

[6] G. Navarro. Searching in metric spaces by spatial approximation. In *Proc. String Processing and Information Retrieval (SPIRE'99)*, pages 141–148. IEEE CS Press, 1999.

[7] A. Pascal, A. De Battista, G. Gutierrez, and N. Herrera. Indice metrico-temporal event-fhqt. In *Actas del XIIII Congreso Argentino de Ciencias de la Computación*, La Rioja, Argentina, 2008.

[8] A. Pascal, A. De Battista, G. Gutierrez, and N. Herrera. Métodos de acceso para bases de datos métrico - temporales. In *Actas del XV Congreso Argentino de Ciencias de la Computación*, pages 1061–1070, Jujuy, Argentina, 2009.

[9] A. Pascal, De Battista, G. Gutierrez, and N. Herrera. Procesamiento de consultas métrico-temporales. In *XXIII Conferencia Latinoamericana de Informática*, pages 133–144, San José de Costa Rica, 2007.

[10] D. Ruano, N. Herrera, C. Ruano, and A. Villegas. Representación en memoria secundaria del trie de sufijos. In *Actas del XVI Congreso Argentino de Ciencias de la Computación*, Morón, Buenos Aires, 2010.

[11] B. Salzberg and V. J. Tsotras. A comparison of access methods for temporal data. *ACM Computing Surveys*, 31(2), 1999.

[12] J. Vitter. External memory algorithms and data structures: Dealing with massive data. *ACM Computing Surveys*, 33(2):209–271, 2001.