Métodos de Acceso para Bases de Datos Métricas

Jorge Arroyuelo, Maria E. Di Genaro, Damián Escudero, Alejandro Grosso, Verónica Ludueña, Cintia Martínez, Nora Reyes
Dpto. de Informática, Fac. de Cs. Físico-Matemáticas y Naturales, Universidad Nacional de San Luis
{*bjarroyu, digeme, agrosso, vlud, nreyes*}@unsl.edu.ar, *escudero.damian.ez*@gmail.com, *cintiavmartinez*@hotmail.com
Edgar Chávez
Centro de Investigación Científica y de Educación Superior de Ensenada, México
elchavez@cicese.mx
Karina Figueroa
Fac. de Cs. Físico-Matemáticas, Universidad Michoacana de San Nicolás de Hidalgo, México
karina@fismat.umich.mx
Rodrigo Paredes
Dpto. de Cs. de la Computación, Fac. de Ingeniería, Universidad de Talca, Chile
raparede@utalca.cl

Resumen

En la actualidad se ha incluído, en mayor o menor medida y en casi todo ámbito, a la ciencia de la computación. Esto a provocado todo tipo de requerimientos de usuarios de distinta índole, y desde campos muy disímiles. Para satisfacer estas demandas se deben desarrollar aplicaciones capaces de manipular eficientemente datos no convencionales muy dispares como: audio, huellas digitales, texto, video, imágenes, secuencias de ADN, etc. Además es necesario utilizar depósitos especializados y búsquedas no exactas sobre estos tipos de datos, porque las soluciones tradicionales no suelen enfrentar tales requerimientos.

Por otro lado, la gran cantidad de datos que se deben manipular para lograr respuestas adecuadas y eficientes, hace necesario un uso eficaz del espacio disponible, lo que implica que las estructuras utilizadas para acceder a este tipo de base de datos, deben ser *estructuras de datos conscientes de la jerarquía de memoria*. Un modelo en el cual se puede utilizar estructuras de datos especializadas o métodos de acceso que contemplen estos aspectos son las *Bases de Datos Métricas*. Con todas estas consideraciones en mente, esta investigación pretende contribuir a consolidar este nuevo modelo de bases de datos desde varias perspectivas.

Palabras Claves: bases de datos no convencionales, índices, lenguajes de consulta.

Contexto

La investigación que se realiza en este ámbito, está enfocada en lograr que las bases de datos destinadas a manipular datos no estructurados, no convencionales, alcancen la madurez de las bases de datos tradicionales. Esto incluye además, plantear nuevas arquitecturas del procesador que mejoren a muy bajo nivel los administradores de estas bases de datos. Se busca así contribuir a distintos campos de aplicación: sistemas de información geográfica, robótica, visión artificial, diseño asistido por computadora, computación móvil, entre otros.

Los estudios que dieron lugar al presente trabajo se realizan en el marco del Proyecto *Tecnologías Avanzadas de Bases de Datos*, en la línea *Bases de Datos no Convencionales* de la Universidad Nacional de San Luis, en colaboración con investigadores de otros grupos de: Universidad Michoacana de San Nicolás de Hidalgo (México) y Centro de Investigación Científica y de Educación Superior de Ensenada (México). Este proyecto finalizó a fines de 2017 y su continuación está en proceso de evaluación.

Introducción

El advenimiento de la computación a todos los ámbitos de la sociedad, tanto el laboral, como el productivo, recreativo, científico, artístico, de la salud, etc., ha exigido el desarrollo de aplicaciones capaces de adaptarse tanto a estos nuevos entornos como a los diversos usuarios de las mismas. Para ello, las bases de datos han debido evolucionar hasta ser capaces de administrar todo tipo de datos y responder consultas sobre los mismos de una manera totalmente diferente a la tradicional, muchas veces más intuitiva. Estos avances se han visto reflejados en áreas como: comparación de huellas digitales, reconocimiento de voz, reconocimiento facial, bases de datos médicas, reconocimiento de imágenes, minería de

datos, recuperación de texto, biología computacional, clasificación y aprendizaje automático, etc.

El modelo de *espacios métricos* resulta adecuado para englobar ciertas características que comparten todas estas aplicaciones, a pesar de ser tan diversas. Formalmente un espacio métrico consiste de un universo de objetos y una función de distancia, definida entre ellos, que mide cúan diferentes son los objetos. Este escenario es propicio para resolver demandas tales como, ingresar una imagen a un buscador y esperar que éste muestre imágenes parecidas a la provista, o un trozo de canción y se encuentren las canciones similares a dicho trozo. Para este tipo de problemas, las búsquedas exactas carecen de sentido y son más naturales sobre estos tipos de datos las *búsquedas por similitud* que provee este modelo.

Para evitar la examinación secuencial de los datos al responder eficientemente a este tipo de búsquedas, se utilizan los *Métodos de Acceso Métricos* (MAMs). Sin embargo, es esencial su optimización, ya que la mayoría de estos métodos no admiten actualizaciones, ni están diseñados para soportar conjuntos masivos de datos y tampoco para resolver operaciones de búsquedas complejas.

El estudio de los lenguajes de consulta, es otra de las áreas exploradas, se busca incrementar su expresividad para formular consultas más precisas. Además, se busca caracterizar nuevas arquitecturas que permitan reducir el flujo de bits entre el procesador y la memoria en relación a la cantidad de datos utilizados por cada programa, para mejorar el desempeño en administradores de bases de datos (DBMS) a bajo nivel.

Líneas de Investigación y Desarrollo

Bases de Datos no Convencionales

Como se mencionó, se utilizarán los espacios métricos para modelizar aquellas bases de datos no convencionales, que administran videos, imágenes, texto libre, secuencias de ADN, audio, etc. En este ámbito es necesario responder eficientemente consultas por similitud, haciendo uso de MAMs y debido a lo costoso que resultan los cálculos de distancia, el número de cálculos realizados al crear el índice o al realizar búsquedas es usado como medida general de complejidad. Por ello, se analizan aquellos MAMs que han mostrado buen desempeño en las búsquedas, para optimizarlos, siendo conscientes para ello de la jerarquía de memorias.

En general, un espacio métrico consta de un universo $\mathbb{U}$ y una función de distancia d y dada una base de datos $X \subseteq \mathbb{U}$ y una consulta $q \in \mathbb{U}$, las consultas por similitud son de dos tipos: por *rango* o de *k-vecinos más cercanos* (k-NN).

Grafo de los k Vecinos

Una de las búsquedas por similitud tradicionales, la de los k vecinos más cercanos, es utilizada por aplicaciones como la predicción de funciones, la cuantificación y compresión de imágenes, la clasificación y aprendizaje automático, entre otras. En este caso, dado un elemento $u \in \mathbb{U}$ y sea $X \subseteq \mathbb{U}$ la base de datos, se recuperan los k elementos en X-$\{u\}$ que tengan la menor distancia d a u. Una generalización de esta búsqueda es la obtención de los k-vecinos más cercanos de *todos* los elementos de la base de datos (*All-k-NN*). La solución burda a este problema, comparar cada elemento de la base de datos con todos los demás, tiene una complejidad de n^2 cálculos de distancia, con $|X|= n$. Por ejemplo, una solución más eficiente resulta al preprocesar los datos por medio de un índice para reducir el número de cálculos de distancia en las búsquedas.

Entre las soluciones propuestas para espacios métricos generales, algunas se basan en la construcción del *Grafo de los k-vecinos más cercanos* (kNNG) [10], cuyo desempeño supera algunas de las técnicas clásicas. El kNNG indexa un espacio métrico y luego se emplea en la resolución de las consultas por similitud. Sin embargo, cuando la función de distancia es demasiado costosa de calcular, o si se tiene una base de datos masiva, el costo de la construcción de un índice, para luego obtener los vecinos más cercanos, puede resultar excesivo. Al igual que resolver consultas en espacios métricos de alta dimensión, donde muchas veces se requiere revisar casi todo el conjunto de datos sin importar la estrategia utilizada. Tanto para hacer frente a éstas situaciones, como para satisfacer los requerimientos de algunas aplicaciones que priorizan la velocidad sobre la precisión [11, 6, 12, 7], es que se consideran las *búsquedas por similitud aproximadas*. Es decir, se aceptan algunos "errores" en la respuesta, si con esto se mejora la complejidad de la misma. Teniendo en cuenta estas consideraciones, se han desarrollado algunas propuestas que construyen una aproximación del kNNG, que se denominó *Grafo de vecinos cercanos* (knNG) [4], el cual conecta cada objeto de la base de datos con k vecinos *cercanos*, relajando la condición que los k vecinos devueltos sean los más cercanos a u de toda la base de datos. Entonces, se puede perder alguno muy cercano y en su lugar devolver otro un poco más lejano.

En una primera aproximación, se consideró un caso particular del knNG, cuando $k = 1$, y se obtuvo el 1nNG, el grafo que conecta a cada elemento con un elemento cercano de la base de datos, que puede ser, o no, su vecino más cercano. Entonces, aprovechando el profundo conocimiento que se tiene de *DiSAT*, se propuso en [4] un enfoque novedoso al problema, que utiliza la información obtenida durante la construcción del índice para construir el 1nNG; cada objeto es vinculado con el elemento más cercano de la base de datos con el que se comparó durante la construcción del *DiSAT*. Esto retorna una aproximación del 1NNG, la cual, pese a que es bastante buena, puede mejorarse mediante reconstrucciones adicionales. Esta propuesta permite recuperar el 1nNG con bajo costo, una muy buena precisión y un error bajo, logrando un buen compromiso calidad/tiempo, y llamativamente *sin realizar ninguna búsqueda.*

Otras propuestas abordadas resuelven el knNG sin recurrir a ningún índice, ni siquiera a su construcción. Se plantean distintas maneras de seleccionar muestras de la base de datos, a las que se le calculan sus vecinos más cercanos, y diferentes formas de utilizar la información conseguida en ese proceso, para calcular vecinos aproximados para el resto de los objetos; utilizando propiedades de la función de distancia, como la desigualdad triangular. Estos planteos resultan muy prometedores.

Métodos de Acceso Métricos

Como se dijo anteriormente, una de las optimzaciones necesarias a los MAM's es el dinamismo. Por ejemplo, considerando el *Árbol de Aproximación Espacial* (*SAT*), un índice con muy buen desempeño en espacios de mediana a alta dimensión, pero totalmente estático, se desarrolló el *Árbol de Aproximación Espacial Dinámico (DSAT)* [9] que permite realizar inserciones y eliminaciones, conservando muy buen desempeño en las búsquedas, pero que agrega un parámetro a sintonizar. El *Árbol de Aproximación Espacial Distal* (*DiSAT*) [5], una variante también estática del *SAT* y sin parámetros, logra optimizar las búsquedas respecto de ambos (SAT y DSAT). Por ello, se ha propuesto la *Foresta de Aproximación Espacial Distal* (*DiSAF*) [3], que es dinámica, para memoria principal y que para lograr mejorar al máximo su desempeño, aplica la técnica de dinamización de Bentley y Saxe al *DiSAT* y aprovecha el profundo conocimiento que se tiene sobre la aproximación espacial.

Sin embargo, muchas veces los índices no caben en memoria principal, ya sea porque administran una base de datos masiva, o porque los objetos de la misma son muy grandes. Entonces surge la necesidad de diseñar índices para memoria secundaria. Muchos de estos índices se basan en "agrupar elementos"; y para analizar cuán buenos son los agrupamientos que logran, se pueden utilizar estrategias de optimización basadas en heurísticas bioinspiradas. Teniendo esto en consideración, se han diseñado dos nuevos índices basados en la *Lista de Clusters*(*LC*) [6] que son totalmente dinámicos, es decir, admiten inserciones y eliminaciones de elementos y están especialmente diseñados para trabajar sobre grandes volúmenes de datos [9]. La *Lista de Clusters Dinámica* (*DLC*), tiene buen desempeño en espacios de alta dimensión, con buena ocupación de página y operaciones eficientes tanto en cálculos de distancia como en operaciones de I/O. Sin embargo, las búsquedas en ella deben recorrer completamente la lista de centros de los clusters, elevando los costos. El *Conjunto Dinámico de Clusters* (*DSC*), también mantiene los clusters en memoria secundaria, pero organiza los centros de clusters en un *DSAT* en memoria principal, permitiendo que las búsquedas realicen menos cálculos de distancia y accedan a menos páginas/clusters. La información de ese *DSAT* también se aprovecha en las inserciones, mejorando los costos de las operaciones en cálculos de distancia y manteniendo los bajos costos de acceso a disco. Ambos, *DLC* y *DSC*, han demostrado tener una razonable utilización de páginas de disco y son competitivas respecto a las alternativas representativas del estado del arte.

Otro aspecto a considerar en este caso es la calidad de los clusters generados. Por lo tanto, una variante que se está considerando para la *DSC* es que en lugar de insertar los elementos en el índice a medida que van llegando, se puede demorar la incorporación de cada nuevo elemento a un cluster hasta tener varios elementos y poder determinar así un mejor agrupamiento de los elementos. Esto permite además reducir el costo de construcción del índice, porque se realiza una escritura de un cluster en disco luego de varias inserciones y además implícitamente puede mejorar los costos de búsqueda al lograr clusters más compactos y que aseguran una total ocupación de la página del disco, achicando así el tamaño del archivo y reduciendo así los tiempos de acceso.

Algunas aplicaciones requieren que las respuestas sean aún más rápidas, aunque sea a costa de perder algunos elementos: se intercambia precisión (devolviendo sólo algunos objetos relevantes) por ve-

locidad en la respuesta. Este tipo de búsquedas se denominan *aproximadas*. Para conjuntos de datos masivos, las búsquedas por similitud aproximadas permiten obtener un buen balance entre el costo de las búsquedas y la calidad de la respuesta obtenida. El *Algoritmo Basado en Permutaciones* (*PBA*) [2]. es uno de los mejores representantes de este tipo de consultas, logrando una respuesta de alta calidad a un bajo costo. Por ello, se ha diseñado la *Lista Dinámica de Permutaciones Agrupadas* (*DLCP*) [8], que combina *LC* con *PBA*), es dinámica y para memoria secundaria. Este índice agrupa por distancia entre las permutaciones de los objetos, en lugar de por distancia entre objetos y se le puede indicar cuántos cálculos de distancia y/o operaciones de I/O utilizar, para obtener una respuesta rápida, aunque menos precisa. Además, se están considerando nuevas variantes para obtener mejores resultados.

Arquitecturas de Procesadores Orientadas a Bases de Datos

La arquitectura del procesador es la funcionalidad que se le provee al programador en lenguaje de máquina, modos de direccionamiento, operaciones, interrupciones y entrada-salida [1]. En ella se distinguen: la organización básica del flujo de datos y el control que se utilizan para alcanzar dicha funcionalidad (*implementación*) y la estructura física que se utiliza para materializar la implementación (*realización*). El lenguaje de máquina (LM) actual no es ni un lenguaje de aplicación ni un lenguaje de hardware, sino algo intermedio. Entonces, ¿por qué no interpretar directamente un lenguaje de alto nivel en lugar de compilar a un lenguaje intermedio? o ¿por qué no darle acceso directo al programador/compilador al hardware en lugar de restringirlo al LM? ¿En qué nivel debería estar el LM?

Se puede elegir la estrategia de "impulso hacia arriba"; es decir subir el nivel, para mejorar el desempeño de la máquina, además de facilitar el uso del lenguaje de máquina. Un aspecto a considerar en este caso es el tráfico de bits y la forma usual de reducirlo es tener una arquitectura que haga lo más posible con cada búsqueda de instrucción, abandonando la arquitectura de bajo nivel y yendo tan alto como el software lo permita. El otro aspecto es explotar la concurrencia, porque si una implementación conoce más sobre lo que debe ser hecho entonces es posible que a menudo realice varias acciones simultánemente. El implementador posee varias técnicas para aumentar la concurrencia: paralelizar, segmentar (pipelining), adelantar, poner a un lado (cache look-aside), adivinar y corregir (control and data prediction). La otra estrategia es considerar el "impulso hacia abajo". Aún si todas las aplicaciones fueran escritas en lenguaje de alto nivel, hay razones para definir una arquitectura de computadora de nivel más bajo, pues existe conflicto de intereses entre usuario e implementador: el usuario desea expresar en forma simple y breve, haciendo uso del contexto, y el implementador desea que cada instrucción sea interpretada independientemente del resto.

Por lo tanto, es importante definir una arquitectura cuando se construye una computadora. En la actualidad la investigación sobre arquitecturas de procesadores ha sido desplazada por la de implementación de procesadores. La mayoría de los trabajos de investigación se dedican a mejorar técnicas de predicción (tanto de control como de datos), técnicas para sincronizar y comunicar procesadores (núcleos) mediante mensajes y/o memoria compartida. Muchas de estas técnicas de implementación surgieron en los años 60 y hoy se han incorporado a los diseños de microprocesadores actuales. Sin embargo, estas técnicas de implementación se podrían aplicar a todo tipo de arquitectura, desde una arquitectura RISC, [1] que intenta acercar el lenguaje de máquina al hardware del procesador, a una arquitectura que se aleje del hardware e intente disminuir el tráfico de bits entre procesador y memoria. El objetivo en esta área es plantear nuevas arquitecturas que minimicen el tráfico de bits entre el procesador y la memoria. Se está construyendo un simulador del set de instrucciones AMD-64 o x86-64, como "benchmark", para evaluar el tráfico de bits, como Specint y Specfp para la arquitectura x86. Luego, se evaluará el tráfico de bits para la arquitectura propuesta sobre los mismos benchmarcks, lo que implica construir tanto el simulador de la arquitectura como el compilador C para la misma. Finalmente, se pretende aprovechar el conocimiento adquirido para, desde bajo nivel, mejorar el desempeño de los DBMSs.

Otra linea de investigación en esta materia, se refiere al diseño de sumadores/restadores y multiplicadores de punto flotante, con vistas a generar una especificación en un lenguaje de descripción de hardware, que concuerde en parte con el estandar IEEE para números de simple precisión. La descripción debe ser sintetizable sobre lógica programable como por ejemplo los FPGA (Field Programmable Gate Array). Se considerarán distintas implementaciones para evaluar su desempeño y el área ocupada de ca-

[1] Acrónimo del inglés "Reduced Instruction Set Computer".

da una. Para la evaluación de estos diseños se deberá disponer de varios sumadores y multiplicadores que exploten el paralelismo presente en la evaluación de redes neuronales para el reconocimiento de imágenes, cuyas estructuras contienen cientos de miles de neuronas. Estos resultados podrían servir para definir funciones de similitud entre imágenes, es decir, dos imágenes que son clasificadas con el mismo peso por la red, se consideran muy similares.

Resultados y Objetivos

Los estudios realizados sobre el modelo de espacios métricos permitirán mejorar el desempeño de los MAMs analizados y estudiar la aplicación de los resultados obtenidos a otros [3, 4, 5, 9, 8].

Se profundizará el estudio del diseño de estructuras de datos, mejorando su eficiencia y adaptándolas mejor al nivel de la jerarquía de memorias donde se almacenarán y a las características de los datos a ser indexados. Se espera brindar nuevas herramientas eficientes de administración para bases de datos métricas, que logren acercar su desarrollo al de los modelos tradicionales de base de datos.

Se continuará analizando la manera de resolver consultas eficientemente, sin la utilización de índices. Además, se espera mejorar el desempeño de las operaciones de bajo nivel en los DBMS, mediante una nueva arquitectura del procesador.

Actividades de Formación

Se están formando investigadores en:
Doctorado en Cs. de la Computación: una tesis sobre expresividad de lenguajes lógicos de consulta.
Maestría en Cs. de la Computación: una tesis sobre búsqueda por similitud aproximada y otra sobre un índice dinámico eficiente.
Maestría en Informática: una tesis, de la Universidad Nacional de San Juan, sobre un índice dinámico para búsquedas aproximadas en disco.
Ingeniería en Computación: un trabajo de fin de carrera sobre diseño de sumadores/restadores y multiplicadores de punto flotante.

Referencias

[1] G. Blaauw and F. Brooks, Jr. *Computer Architecture: Concepts and Evolution*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1997.

[2] E. Chávez, K. Figueroa, and G. Navarro. Effective proximity retrieval by ordering permutations. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(9):1647–1658, Sept 2008.

[3] E. Chávez, M. Di Genaro, N. Reyes, and P. Roggero. Decomposability of disat for index dynamization. *Computer Science & Technology*, pages 110–116, 2017.

[4] E. Chávez, V. Ludueña, N. Reyes, and F. Kasián. All near neighbor graph without searching. *Computer Science & Technology*, page 7, 2018. Por aparecer.

[5] E. Chávez, V. Ludueña, N. Reyes, and P. Roggero. Faster proximity searching with the distal SAT. *Inf. Systems*, 59:15 – 47, 2016.

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

[7] P. Ciaccia and M. Patella. Approximate and probabilistic methods. *SIGSPATIAL Special*, 2(2):16–19, 2010.

[8] K. Figueroa, C. Martínez, R. Paredes, N. Reyes, and P. Roggero. Dynamic list of clustered permutations on disk. In *Computer Science and Technology Series: XXI Argentine Congress of Computer Science Selected Papers*, pages 201–211. EDULP, 2016.

[9] G. Navarro and N. Reyes. New dynamic metric indices for secondary memory. *Inf. Systems*, 59:48 – 78, 2016.

[10] R. Paredes, E. Chávez, K. Figueroa, and G. Navarro. Practical construction of k-nearest neighbor graphs in metric spaces. In *Proc. 5th Workshop on Efficient and Experimental Algorithms (WEA)*, LNCS 4007, pages 85–97, 2006.

[11] H. Samet. *Foundations of Multidimensional and Metric Data Structures (The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2006.

[12] P. Zezula, G. Amato, V. Dohnal, and M. Batko. *Similarity Search: The Metric Space Approach (Advances in Database Systems)*. Springer-Verlag New York, 2005. XVIII, 220 p., Hardcover ISBN: 0-387-29146-6.