Consultas sobre Bases de Datos no Convencionales

Jorge Arroyuelo, Susana Esquivel, Alejandro Grosso, Verónica Ludueña, Nora Reyes
Departamento de Informática, Universidad Nacional de San Luis
{bjarroyu, esquivel, agrosso, vlud, nreyes}@unsl.edu.ar

Gonzalo Navarro
Departamento de Ciencias de la Computación, Universidad de Chile.
gnavarro@dcc.uchile.cl

Resumen

Gracias a los avances en el ámbito computacional la cantidad de información que nuestra sociedad produce y almacena crece exponencialmente día a día. Los diferentes tipos y tamaños de datos provenientes de diversas fuentes como transacciones financieras, revistas, fotografías, etc., han dado lugar a la aparición de depósitos no estructurados de información, *Bases de Datos no Convencionales*, donde se consultan nuevos tipos de datos (texto libre, imágenes, audio, vídeo, etc.). Estos grandes volúmenes de información representan un desafío y exigen dispositivos de almacenamiento capaces de mantenerlos y de proveer un acceso eficiente y efectivo a los mismos. Este escenario requiere un modelo más general tal como las *Bases de Datos Métricas*, con un nivel de madurez similar al de las bases de datos tradicionales. La amplia brecha entre los tiempos de CPU y los de I/O hace necesario considerar memorias con mayor capacidad y más rápidas, lo que ha promovido la aparición estructuras de datos especializadas que tienen en cuenta estas arquitecturas como las *estructuras de datos compactas* y las *estructuras de datos con I/O eficiente*. Nuestra investigación apunta a contribuir a la madurez de estas nuevas bases de datos.

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

Contexto

La línea de investigación *Base de Datos no Convencionales* integra el Proyecto Consolidado 330303 "Tecnologías Avanzadas de Bases de Datos"; este proyecto pertenece a la Universidad Nacional de San Luis y se encuentra dentro del Programa de Incentivos a la Investigación (Código 22/F014).

En el ámbito de este proyecto se han desarrollado actividades vinculadas al tratamiento de objetos de diversos tipos, útiles en variados campos de aplicación, por ejemplo, robótica, visión artificial, computación gráfica, sistemas de información geográfica, computación móvil, diseño asistido por computadora, motores de búsqueda en internet, entre otras, y que se relacionan en tales bases de datos. Tanto la investigación de aspectos empíricos, teóricos y aplicativos del problema general de administrar una base de datos capaz de operar con tipos de datos no convencionales, como el análisis de distintos tipos de bases de datos, la expresividad de los lenguajes de consulta, los operadores necesarios para responder consultas de interés, como así también las estructuras y operaciones necesarias para responderlas eficientemente son actividades centrales de esta línea.

También es importante mencionar que nuestras investigaciones se han desarrollado además en el marco de un proyecto dentro del Programa de Promoción de la Universidad Argentina para el Fortalecimiento de Redes Interuniversitarias III en el que participaba nuestra universidad junto con las Universidades de Chile y de La Coruña (España) (Proyecto 1-10-269, finalizado a fines de 2010).

1. Introducción y Motivación

Debido a la diversidad de datos presentes en el marco computacional, nos centraremos en aquellas estructuras capaces de manejar datos tales como: secuencias, textos, espacios métricos, entre otros. Es claro que las búsquedas exactas sobre estos datos carecen de sentido, por lo que se hace necesario un modelo como el de *espacios métricos*, más general y donde las *búsquedas por similitud*, más naturales sobre estos tipos de datos, son posibles.

Mientras han aparecido nuevos niveles en la jerarquía de memoria (caches de tamaño cada vez más considerable), la brecha entre los tiempos de CPU y los de I/O se ha mantenido creciente; esto ha hecho cada vez más atractivo el uso de estructuras de da-

tos que ocupen poco espacio, lo que puede lograrse comprimiendo la información sobre la que actúan. Si bien trabajar sobre esta información compacta es más laborioso, el hecho de poder mantenerla en una memoria de órdenes de magnitud más rápida la convierte en una alternativa ventajosa frente a las implementaciones clásicas. Este contexto ha originado líneas de investigación que consideran estas diferencias de costos de operaciones, y diseñan estructuras de datos más eficientes (en espacio o en términos de la I/O u otras medidas de eficiencia) para memorias jerárquicas, utilizando la compacticidad o la I/O eficiente. Además de diseñar estructuras de datos estáticas, planeamos investigar otros aspectos tales como operaciones de búsqueda complejas (más allá de las básicas soportadas por las estructuras de datos clásicas), el dinamismo (es decir actualizaciones eficientes), tratar de obtener una mayor expresividad en los lenguajes que expresan consultas y caracterizar la clase de consultas computables. La "*maldición de la dimensionalidad*" describe el fenómeno por el cual el desempeño de todos los índices existentes se deteriora exponencialmente con la dimensión. Este fenómeno, presente en los espacios de vectores (representación más común para datos multimedia), aún no está completamente determinado en los espacios métricos, aunque los espacios vectoriales son un caso particular de espacios métricos, no está totalmente definido cómo afecta la dimensión a los índices para espacios métricos. Entonces, como de las numerosas estructuras que existen para búsquedas por similitud en espacios métricos, sólo unas pocas trabajan eficientemente en espacios de alta o mediana dimensión, y la mayoría no admiten dinamismo, ni están diseñadas para trabajar sobre grandes volúmenes de datos, es decir, en memoria secundaria, es que se estudian distintas maneras de optimizarlas.

2. Expresividad de Lenguajes de Consulta

Algunos de los lenguajes de consulta que permiten obtener información de una base de datos son equivalentes, en su poder expresivo, a la Lógica de Primer Orden (FO). Sin embargo, puede suceder que la expresividad de FO no alcance para reflejar algunas consultas, esto ha llevado a la búsqueda de un mayor poder expresivo por medio de extensiones a la misma. Se logró incrementar la expresividad de esta lógica pero todavía aparece como incompleta, entonces se enfatizó el estudio de la expresividad de la Lógica de Segundo Orden (SO). En los últimos años se han realizado importantes investigaciones sobre relación cercana entre la *complejidad computacional*, es decir la cantidad de recursos necesarios para resolver un problema sobre algún modelo de máquina computacional, y la *complejidad descriptiva*, o sea el orden de la lógica que se necesita para describir el problema. La consecuencia más importante de esta relación es el resultado de Fagin [6]. Allí se establece que las propiedades de las estructuras finitas que son definidas por sentencias existenciales de segundo orden coinciden con las propiedades que pertenecen a la clase de complejidad NP, lo cual fue extendido por Stockmeyer [12] estableciendo una relación cercana entre la lógica de segundo orden y la jerarquía polinomial. Hay muchos resultados igualando la expresividad lógica a la complejidad computacional, pero ellos requieren estructuras ordenadas (ver [7], [8]).

En nuestra investigación se ha introducido la definición de una restricción de SO que consiste en restringir las relaciones que pueden tomar los cuantificadores de SO, considerando a la lógica como uno de los lenguajes de consulta a base de datos. El tipo de relaciones a los que estos cuantificadores pueden referirse son relaciones cerradas bajo *FO type*. Esta lógica (SOF) es un intento de lograr una lógica de mayor poder expresivo que la lógica definida por Dawar(SO^{ω}) en la que los cuantificadores sólo pueden tomar relaciones cerradas bajo FO-k tipos. Se demostró que nuestra lógica incluye estrictamente la lógica definida por Dawar [4]. También se ha podido definir una nueva clase de complejidad descriptiva (NP^{F}) y se la ha caracterizado mediante una modificación de las máquinas relacionales. Esta nueva clase de complejidad caracteriza el fragmento existencial de nuestra lógica.

En estos temas están desarrollando sus tesis de doctorado dos investigadores de la línea.

3. Indexación de Texto

Entre los datos que aparecen en formato digital en cantidades creciente cada día, se encuentra el *texto*, el cual debe ser manejado en forma adecuada para permitir un acceso eficiente al mismo como también un ahorro de espacio de almacenamiento. Una herramienta clave para el manejo eficiente de grandes cantidades de texto son los índices textuales. Una variante de los índices clásicos son los llamados índi-

ces *comprimidos* que suelen aprovechar la existencia de la jerarquía de memorias y que no sólo responden a consultas en forma rápida sino que también almacenan sus datos y el texto en forma eficiente tratando de ocupar el menor espacio posible.

Por otro lado, entre las operaciones necesarias a realizar sobre el texto encontramos las búsquedas por similitud; este tipo de búsqueda tiene aplicaciones tales como la recuperación de errores (en reconocimiento óptico de caracteres, spelling), biología computacional, comunicaciones de datos, data mining, bases de datos textuales, entre otras. Aquí el problema general de la *búsqueda aproximada* puede verse como: sea $T = T[1, n]$ un *texto*, y $P = P[1, m]$ un *patrón* sobre el alfabeto Σ (con $m << n$) y un entero k, se desea encontrar y devolver todos los *substrings* en el texto T que sean una ocurrencia aproximada de P, con a lo más k diferencias. La diferencia entre dos strings α y β se obtiene con la *distancia de edición* o *distancia de Levenshtein* d; $d(\alpha, \beta)$ es el mínimo número de inserciones, eliminaciones y/o sustituciones de caracteres que se deben realizar para convertir β en α.

Una variante de los índices clásicos son los llamados índices *compactos* que suelen aprovechar la existencia de la jerarquía de memorias. La idea de los índices compactos se diferencia de la compresión pura en su capacidad de manipular los datos en forma comprimida, sin tener que descomprimirlos primero, por ejemplo son capaces de realizar una búsqueda sin la necesidad de descomprimir los datos. En la actualidad los índices compactos pueden manipular secuencias de bits o de símbolos generales, árboles en general, grafos, colecciones de texto, permutaciones y mapping, sumas parciales, búsqueda por rango en una y más dimensiones, etc. Entre la gran variedad de índices comprimidos existentes encontramos los basados en listas de ocurrencia de q-gramas.

Indexación con *q-gramas*. Un q-grama es una subsecuencia de tamaño q de un texto. Un índice de q-gramas es una estructura de datos que permite encontrar rápidamente en el texto todas las ocurrencias de un q-grama dado. Existen distintas implementaciones de índices para q-gramas. La básica es un arreglo de punteros de tamaño $|\Sigma|^q$. Cada posición del arreglo referencia a la lista de ocurrencias en el texto del q-grama correspondiente. Una mejora a este esquema de indexación es reducir el tamaño del arreglo de punteros por medio de un hashing eficiente, sin una demora significativa en las búsquedas. Otro enfoque usa una estructura de trie construido con los distintos q-gramas que aparecen en el texto. Cada hoja del trie contiene un puntero a la lista de ocurrencias del correspondiente q-grama en el texto.

Las implementaciones anteriores encuentran todas las ocurrencias de un q-grama dado en tiempo óptimo en función del número de ocurrencias encontradas. Pero todas sufren del mismo inconveniente: el tamaño del índice se vuelve impráctico al crecer la longitud del texto.

Existe un índice para q-gramas que comprime las listas de ocurrencias de q-gramas. En él, la estructura (hashing o trie) para los distintos q-gramas, ahora llamada *índice primario*, es aún necesaria para proveer un punto de comienzo para las búsquedas. Éstas pueden dividirse en dos etapas: en la primera se seleccionan las regiones del texto en las que ocurren todos los q-gramas del patrón, es decir que éste podría estar presente en esas porciones de texto, y en una segunda etapa se verifica la existencia del patrón buscado en las regiones de texto seleccionadas en la primer etapa. Durante la implementación de este índice se pueden analizar distintas técnicas de compresión de las listas de ocurrencias de q-gramas seleccionando la mejor.

Esta temática se investiga en el marco de la tesis de maestría de un investigador de la línea.

4. Bases de Datos Métricas

Tomando como modelo para las bases de datos no convencionales a los espacios métricos, surge la necesidad de responder consultas por similitud eficientemente haciendo uso de *métodos de acceso métricos* (MAMs). En espacios métricos generales la complejidad usualmente se mide como el número de cálculos de distancias realizados. Por ello, se analizan aquellos MAMs que han mostrado buen desempeño en las búsquedas, con el fin de optimizarlos aún más, teniendo en cuenta la jerarquía de memorias.

4.1. Dimensión Intrínseca

En los espacios vectoriales existe una clara relación entre la dimensión (*intrínseca*) del espacio y la dificultad de buscar. Se habla de "intrínseca", como opuesta a "representacional". Los algoritmos más ingeniosos se comportan más de acuerdo a la dimensión intrínseca. Hay varios intentos de medir la dimensión intrínseca en espacios de vectores, como la transformada de *Karhunen-Loève (KL)* y otras

medidas como *Fastmap* y, para espacios no uniformemente distribuidos, la *dimensión fractal* [2].

Existen sólo unas pocas propuestas diferentes sobre cómo estimar la dimensión intrínseca de un espacio métrico tales como el *exponente de la distancia* [13], y la medida de dimensión intrínseca como una medida cuantitativa basada en el histograma de distancias [3]. Aunque, también parece posible adaptar algunos de los estimadores de distancia en espacios de vectores para aplicarlos a espacios métricos generales, como por ejemplo *Fastmap* y *dimensión fractal*.

Muchos autores [3] han propuesto usar histogramas de distancia para caracterizar la dificultad de las búsquedas en espacios métricos arbitrarios. Existe al menos una medida cuantitativa [3], pero ella no refleja fielmente la facilidad o dificultad de buscar en un espacio métrico dado.

En aplicaciones reales de búsqueda en espacios métricos, sería muy importante contar con un buen estimador de la dimensión intrínseca porque nos permitiría decidir el índice adecuado a utilizar en función de la dimensión del espacio. Además, tener una buena estimación de la dimensión nos permitiría, en algunas ocasiones, elegir la función de distancia de manera tal que se obtenga una menor dimensión.

Este tema se está desarrollando como un trabajo final de la Lic. en Ciencias de la Computación.

4.2. Árbol de Aproximación Espacial Dinámico

El estudio del *Árbol de Aproximación Espacial* [9], uno de los MAMs que había mostrado un muy buen desempeño en espacios de mediana a alta dimensión, pero que era totalmente estático, nos permitió el desarrollo de un nuevo índice llamado *Árbol de Aproximación Espacial Dinámico (SATD)* [10] que permite realizar inserciones y eliminaciones, conservando el buen desempeño en las búsquedas, lo cual ha sido importante ya que pocos índices para espacios métricos son completamente dinámicos.

El *SATD* es una estructura que realiza una partición del espacio considerando la proximidad espacial; pero, si el árbol agrupara los elementos que se encuentran muy cercanos entre sí, lograría mejorar las búsquedas al evitar recorrerlo para alcanzarlos.

Podemos pensar entonces que construimos un *SATD*, en el que cada nodo representa un grupo de elementos muy cercanos ("clusters") y relacionamos los clusters por su proximidad en el espacio. La idea sería que en cada nodo se mantenga el centro del cluster correspondiente, y se almacenen los k elementos más cercanos a él; cualquier elemento a mayor distancia del centro que los k almacenados, pasa a formar parte de otro nodo en el árbol.

Obtuvimos así una estructura más eficiente en espacios donde de antemano se sabe que pueden existir "clusters" de elementos y que aprovechando la existencia de los mismos mejore las búsquedas [1].

Otro aspecto aún por analizar es cuán bueno es el agrupamiento o "clustering"que logra esta estructura, lo cual podría estudiarse haciendo uso de nuevas estrategias de optimización de funciones a través de heurísticas bioinspiradas, las cuales han mostrado ser útiles en detección de clusters.

Estos temas dieron lugar a un trabajo final de la Lic. en Ciencias de la Computación, finalizado a fines de 2010.

4.3. Join Métricos

El modelo de espacios métricos permite cubrir muchos problemas de búsqueda por similitud o proximidad, aunque en general se deja fuera de consideración al operador de ensamble o "join" por similitud, otra primitiva extremadamente importante [5]. De hecho, a pesar de la atención que esta primitiva ha recibido en las bases de datos tradicionales y aún en las multidimensionales, no han habido grandes avances para espacios métricos generales.

Nos hemos planteado resolver algunas variantes del problema de join por similitud: (1) *join por rango:* dadas dos bases de datos de un espacio métrico y un radio r, encontrar todos los pares de objetos (uno desde cada base de datos) a distancia a lo sumo r, (2) *k–pares más cercanos:* encontrar los k pares de objetos más cercanos entre sí (uno desde cada base de datos). Para resolver estas operaciones de manera eficiente hemos diseñado un nuevo índice métrico, llamado *Lista de Clusters Gemelos* (*LTC*) [11], éste se construye sobre ambas bases de datos conjuntamente, en lugar de indexar una o ambas bases de datos independientemente. Esta nueva estructura permite además resolver las consultas por similitud clásicas en espacios métricos sobre cada una de las bases de datos independientemente.

A pesar de que esta estructura ha mostrado ser competitiva y obtener buen desempeño en relación a las alternativas más comunes para resolver las operaciones de join, aún queda mucho por mejorar para que se vuelva una estructura práctica y mucho más eficiente para trabajar con grandes bases de datos métricas. A la fecha se está analizando la construc-

ción de otra clase de índice basada en "permutantes" para resolver el join aproximado de dos bases de datos métricas; es decir que permita rápida y eficientemente encontrar los pares de elementos más similares entre ambas bases de datos, aunque no los obtenga a todos. De esta manera sería posible pensar en extender apropiadamente el álgebra relacional como lenguaje de consulta y diseñar soluciones eficientes para nuevas operaciones, teniendo en cuenta aspectos no sólo de memoria secundaria, sino también de concurrencia, confiabilidad, etc. Algunos de estos problemas ya poseen solución en las bases de datos espaciales, pero no en el ámbito de los espacios métricos.

Este tema forma parte de la tesis doctoral de uno de los investigadores de la línea.

5. Conclusiones y Trabajos Futuros

Como trabajo futuro de esta línea de investigación se consideran varios aspectos relacionados al diseño de estructuras de datos que, consciente de que existe una jerarquía de memorias y de las características particulares de los datos a ser indexados, saquen el mejor partido haciendólas eficientes tanto en espacio como en tiempo.

Se trabajará en particular con estructuras de datos compactas para textos, implementando un índice basado en q-gramas y estudiando su comportamiento en comparación con los mejores auto-índices desarrollados del sitio *Pizza&Chili*.

Respecto de los lenguajes de consulta se continuará analizando la expresividad de distintas extensiones de FO y posibles restricciones de SO, para lograr caracterizar la clase de las consultas computables sobre bases de datos no convencionales.

En el caso de bases de datos métricas, se intentará que los índices se adapten mejor al espacio métrico particular considerado, gracias a la determinación de su dimensión intrínseca, y también al nivel de la jerarquía de memorias en que se deba almacenar. Es importante destacar que estos estudios sobre espacios métricos y sobre algunas estructuras de datos particulares permitirán no sólo mejorar el desempeño de las mismas sino también aplicar, eventualmente, muchos de los resultados que se obtengan a otros índices para bases de datos métricas.

Referencias

[1] M. Barroso, N. Reyes, and R. Paredes. Enlarging nodes to improve dynamic spatial approximation trees. In *Proc. of the 3rd International Conference on Similarity Search and Applications*, pages 41–48. ACM Press, 2010.

[2] F. Camastra. Data dimensionality estimation methods: a survey. *Pattern Recognition*, 36(12):2945–2954, 2003.

[3] E. Chávez and G. Navarro. Towards measuring the searching complexity of metric spaces. In *Proc. International Mexican Conference in Computer Science*, Vol. II, pages 969–978, 2001.

[4] A. Dawar. A restricted second order logic for finite structures. *Information and Computation*, 143:154–174, 1998.

[5] V. Dohnal, C. Gennaro, P. Savino, and P. Zezula. Similarity join in metric spaces. In *Proc. 25th European Conf. on IR Research*, LNCS 2633, pages 452–467, 2003.

[6] R. Fagin. Generalized first-order spectra and polynomial-time recognizable sets. *Complexity of Computation*, 7:43–73, 1974.

[7] N. Immerman. Descriptive and computational complexity. *Computational Complexity Theory*, 38:75–91, 1989.

[8] N. Immerman. Descriptive complexity. *Springer*, 1998.

[9] G. Navarro. Searching in metric spaces by spatial approximation. *The Very Large Databases Journal (VLDBJ)*, 11(1):28–46, 2002.

[10] G. Navarro and N. Reyes. Fully dynamic spatial approximation trees. In *Proc. of the 9th International Symposium on String Processing and Information Retrieval*, LNCS 2476, pages 254–270. Springer, 2002.

[11] R. Paredes and N. Reyes. Solving similarity joins and range queries in metric spaces with the list of twin clusters. *J. of Discrete Algorithms*, 7(1):18–35, 2009.

[12] L. Stockmeyer. The polynomial-time hierarchy. *Theoret. Comput. Sci.*, 3:1–22, 1976.

[13] C. Traina Jr., A. Traina, and C. Faloutsos. Distance exponent: A new concept for selectivity estimation in metric trees. In *ICDE*, page 195, 2000.