Políticas de Selección de Pivotes para Índices Métricos

Norma Edith Herrera
Departamento de Informática
Universidad Nacional de San Luis
nherrera@unsl.edu.ar

Anabella C. De Battista, Andrés J. Pascal
Departamento de Sistemas de Información
Universidad Tecnológica Nacional
Regional Concepción del Uruguay
{debattistaa, pascalj}@frcu.utn.edu.ar

Resumen

El modelo de Espacios Métricos permite formalizar el concepto de búsqueda por similitud en bases de datos no tradicionales. El objetivo es construir *estructuras de datos o índices* que permitan reducir el tiempo necesario para resolver una búsqueda por similitud. Uno de los enfoques para la construcción de índices es el usado por los algoritmos basados en pivotes. En este trabajo abordamos el estudio de este grupo de algoritmos, enfocándonos en políticas de selección de pivotes. Presentamos tres nuevas técnicas para la selección de pivotes, las que se encuentran en la etapa de codificación para su evaluación experimental.

1. Introducción

Con la evolución de los sistemas y las tecnologías de información, las bases de datos actuales han incluido la capacidad de almacenar datos no estructurados tales como imágenes, sonido, video, huellas digitales, entre otros. Las búsquedas que son de interés en ese tipo de bases de datos, son aquellas que recuperan objetos similares a un elemento dado. Este tipo de búsqueda se conoce con el nombre de *búsqueda por similitud* y surge en áreas tales como reconocimiento de voz, reconocimiento de imágenes, compresión de texto, recuperación de texto, biología computacional, por nombrar algunas.

El conjunto X de todos los objetos sobre los cuales se puede realizar la búsqueda, junto con una función de distancia d que mide la similitud entre los elementos de X, se denomina *espacio métrico*, y se denota (X, d). La base de datos será un conjunto $U \subseteq X$. La función de distancia $d : X \times X \rightarrow R^+$ debe satisfacer las propiedades que caracterizan a una métrica, a saber: ***positividad*** ($d(x, y) \geq 0$), ***simetría*** ($d(x, y) = d(y, x)$) y ***desigualdad triangular*** ($d(x, y) \leq d(x, z) + d(z, y)$).

Si bien existen distintos tipos de búsquedas por similitud, una de las más comunes es la ***búsqueda por rango***. Esta búsqueda, que denotaremos con $(q, r)_d$, consiste en recuperar todos los elementos de U cuya distancia a un elemento q dado no supere un rango de tolerancia r; en símbolos $(q, r)_d = \{u \in \mathcal{U} : d(q, u) \leq r\}$. Nos referiremos a q como elemento de consulta o query.

Hay tres factores que afectan el tiempo necesario para resolver una búsqueda por similitud; por un lado tenemos la cantidad de evaluaciones de la función de distancia d que se realizaron durante el proceso de búsqueda; por otro lado tenemos una cierta cantidad de operaciones adicionales que implican un tiempo extra de CPU; finalmente, tenemos un tiempo de I/O determinado por la cantidad de accesos a memoria secundaria, si es que fuera necesario.

En muchas aplicaciones el cálculo de la función de distancia d es tan costoso que las demás componentes que afectan el tiempo pueden ser despreciadas. Éste es el modelo usado en este trabajo;

en consecuencia, nuestra medida de complejidad será la cantidad de evaluaciones de la función de distancia d.

Una búsqueda por similitud puede ser resuelta con $O(n)$ evaluaciones de distancias examinando exhaustivamente la base de datos. Para evitar esta situación, se preprocesa la base de datos por medio de un *algoritmo de indexación* con el objetivo de construir una *estructura de datos o índice*, diseñada para ahorrar cálculos en el momento de resolver una búsqueda. Un algoritmo de indexación se considera eficiente si puede responder una búsqueda por similitud haciendo una cantidad pequeña de cálculos de distancia, sublineal en la cantidad de elementos de la base de datos.

En [7] se presenta un desarrollo unificador de las soluciones existentes en la temática. En dicho trabajo se muestra que todos los enfoques para la construcción de índices en espacios métricos consisten en:

- particionar el espacio en clases de equivalencia.
- indexar las clases de equivalencia.
- durante la búsqueda, usando el índice y la desigualdad triangular, descartar algunas clases y buscar exhaustivamente en las restantes.

La diferencia entre los distintos algoritmos radica en cómo construyen estas clases de equivalencia. Básicamente se pueden distinguir dos enfoques: *algoritmos basados en pivotes* y *algoritmos basados en particiones compactas*. En este trabajo nos centraremos sobre algoritmos basados en pivotes [1, 2, 3, 5, 6, 7].

Comenzaremos dando una pequeña introducción a algoritmos basados en pivotes, luego analizaremos la problemática de selección de un buen grupo de pivotes y finalizaremos dando el trabajo futuro.

2. Algoritmos Basados en Pivotes

Este grupo de algoritmos definen una relación de equivalencia basados en la distancia de los elementos a un conjunto de elementos preseleccionados que llamaremos *pivotes*. Sea $\{p_1, p_2, \ldots, p_k\}$ el conjunto de pivotes, dos elementos son equivalentes si y solo si están a la misma distancia de todos los pivotes:

$$x \sim_{\{p_i\}} y \Leftrightarrow\ d(x, p_i) = d(y, p_i),\ \forall i = 1 \ldots k$$

Gráficamente, cada clase de equivalencia está definida por la intersección de varias capas de esferas centradas en los puntos p_i (ver figura 1, izquierda).

Durante la indexación, se seleccionan k pivotes $\{p_1, p_2, \ldots, p_k\}$, y se le asigna a cada elemento a de la base de datos, el vector o firma:

$$\Phi(a) = (d(a, p_1), d(a, p_2), \ldots, d(a, p_k))$$

Durante la búsqueda se usa la desigualdad triangular junto con la firma de cada elemento para filtrar objetos de la base de datos sin medir su distancia a la query q. Dada $(q, r)_d$, se computa la firma de la query q, $\Phi(q) = (d(q, p_1), d(q, p_2), \ldots, d(q, p_k))$, y luego se descartan todos aquellos elementos a, tales que para algún pivote p_i se cumple que $\mid d(q, p_i) - d(a, p_i) \mid > r$, es decir:

$$max_{1 \leq i \leq k}\{\mid d(a, p_i) - d(q, p_i) \mid\} = L_\infty(\Phi(a), \Phi(q)) \leq r$$

Los elementos no descartados forman parte de una lista de candidatos, que posteriormente se comparan directamente con la query q. Esto significa que la cantidad total de cálculos de la función de

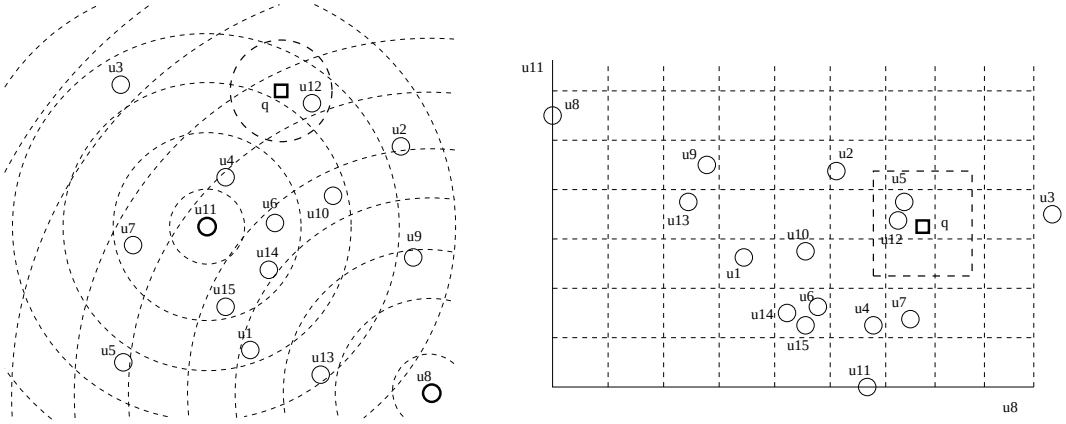

Figura 1: *Un ejemplo de la relación de equivalencia inducida por la intersección de anillos centrados en dos pivotes, u_8 y u_{11} (izquierda); y su correspondiente transformación en un espacio vectorial de dimensión 2 (derecha). También se ilustra la transformación de una búsqueda $(q, r)_d$.*

distancia d queda determinada por la cantidad de pivotes k más la cardinalidad de la lista de candidatos.

Notar que, la relación de equivalencia definida por un conjunto de k pivotes también puede verse como una proyección al espacio vectorial $\mathbb{R}^k$. La i-ésima coordenada de un elemento es la distancia al i-ésimo pivote. Esto significa que hemos proyectado el espacio métrico original $(\mathcal{X}, d)$ en el espacio vectorial $\mathbb{R}^k$ con la función de distancia L_∞ (ver figura 1).

3. Políticas de Selección de Pivotes

Se sabe que la política usada en la selección de pivotes afecta notablemente la performance de la búsqueda [4, 7, 9, 10]. Esto significa que si tenemos dos conjuntos de pivotes del mismo tamaño elegir el mejor de los dos puede reducir la cardinalidad de la lista de candidatos y, en consecuencia, reducir el tiempo de búsqueda. Por otro lado, un grupo pequeño de pivotes bien elegidos puede resultar tan eficiente como un grupo de mayor cantidad de pivotes pero elegidos aleatoriamente. Por lo tanto, el tema de selección de un buen grupo de pivotes para indexar un determinado espacio métrico está siendo ampliamente estudiado.

En [4] se proponen tres técnicas para la selección de un buen grupo de pivotes. Dichas técnicas tratan de maximizar la media μ_D de la distribución de D, donde:

$$D([x],[y]) = max_{1 \leq i \leq k}\{ \mid d(x,p_i) - d(y,p_i) \mid \}$$

De las tres técnicas allí presentadas, la que muestra un mejor desempeño es *selección incremental*. Este método consiste en tomar una muestra de N elementos de la base de datos y seleccionar como primer pivote p_1 a aquel elemento que tenga el máximo valor para μ_D. El segundo pivote p_2 se elige de otra muestra de N elementos de forma tal que $\{p_1, p_2\}$ tenga el máximo valor para μ_D. Este proceso se repite hasta terminar de elegir los k pivotes necesitados.

En dicho trabajo se muestra que un buen grupo de pivotes tiene dos características básicas:

- los pivotes están alejados unos de otros, es decir, la distancia media entre pivotes es mayor que la distancia media entre elementos tomados al azar del espacio métrico.
- los pivotes están alejados del resto de los elementos del espacio métrico.

Los elementos que tienen estas dos propiedades se denominan *outliers*. También se observa que, si bien buenos pivotes tienen la propiedad de ser outliers, no todos los outliers son eficientes como pivotes. A partir de los resultados experimentales se concluye que los conjuntos de outliers tienen un buen desempeño en espacios vectoriales uniformemente distribuidos, pero tienen una baja performance en espacios métricos generales, aún peor que una selección aleatoria.

Por otro lado, en [8] se presenta un enfoque alternativo que consiste en una selección dinámica del conjunto de pivotes, y por consiguiente del índice sobre el que se resolverá la consulta. En lugar de seleccionar durante la construcción del índice un grupo de pivotes que sea efectivo para todo el espacio métrico, se *selecciona durante la búsqueda un grupo de pivotes que sea efectivo para la query* q. Para ello, se construyen varios índices sobre el espacio con distintos grupos de pivotes (elegidos aleatoriamente); luego, durante una búsqueda $(q, r)_d$ se selecciona aquel índice que sea más adecuado a q de acuerdo al conjunto de pivotes con el que fue construido.

En este trabajo se presentan varias heurísticas para la selección del índice adecuado, ellas son: selección por votos, pivote más cercano, pivote más lejano, menor masa total, pivote de menor masa y votación global. A partir de la evaluación experimental de las mismas se muestra que las de mejor desempeño son las heurísticas *pivote de menor masa y votación global*. También se analizan las características que presentan los pivotes de los índices más competitivos, siendo la observación más importante que *los mejores índices son aquellos cuyo grupo de pivotes tienen una mayor varianza*.

4. Trabajo Futuro

Nos proponemos diseñar y evaluar experimentalmente nuevas políticas de selección de pivotes, usando como base los resultados de [4] y [8].

Hasta el momento, hemos diseñado tres nuevas políticas:

- **Selección incremental con máxima varianza**
  Esta técnica consiste en usar la selección incremental presentada en [4] pero, en lugar de maximizar la media μ_D, la selección se realiza de manera tal de maximizar la varianza.

- **Selección a partir de votación global**
  Básicamente, la idea es adaptar las mejores técnicas de optimización local para seleccionar de manera incremental un buen grupo de pivotes con el objetivo de construir un único índice para todo el espacio métrico.
  Para ello, construimos un grupo de índices auxiliares de k pivotes cada uno. Sobre ese grupo de índices realizamos una determinada cantidad de queries bajo la idea de optimización local, es decir, seleccionando el mejor índice de acuerdo a la técnica de votación global. Los k pivotes del índice que resulte elegido mayor cantidad de veces formarán parte del grupo de pivotes para el índice final. Este proceso se repite sobre otro lote de índices auxiliares hasta completar la cantidad total de pivotes requerida para el índice final.

- **Selección a partir de pivote de menor masa**
  Idem a la anterior, pero usando pivote de menor masa en lugar de votación global.

La evaluación de estas políticas se realizará experimentalmente, evaluando su desempeño sobre distintos tipos de espacios métricos: diccionarios de palabras, espacio de documentos e imágenes. Se compararán las tres políticas diseñadas con la selección aleatoria y con la selección incremental de [4], midiendo para cada una de ellas la cantidad media de evaluaciones de distancias requeridas para una búsqueda $(q, r)_d$.

Referencias

[1] R. Baeza-Yates. Searching: an algorithmic tour. In A. Kent and J. Williams, editors, *Encyclopedia of Computer Science and Technology*, volume 37, pages 331–359. Marcel Dekker Inc., 1997.

[2] 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.

[3] W. Burkhard and R. Keller. Some approaches to best-match file searching. *Comm. of the ACM*, 16(4):230–236, 1973.

[4] B. Bustos, G. Navarro, and E. Chávez. Pivot selection techniques for proximity searching in metric spaces. In *Proc. of the XXI Conference of the Chilean Computer Science Society (SCCC'01)*, pages 33–40. IEEE CS Press, 2001.

[5] E. Chávez and K. Figueroa. Faster proximity searching in metric data. In *Proceedings of MICAI 2004. LNCS 2972, Springer*, Cd. de México, México, 2004.

[6] E. Chávez, J. Marroquín, and G. Navarro. Fixed queries array: A fast and economical data structure for proximity searching. *Multimedia Tools and Applications (MTAP)*, 14(2):113–135, 2001.

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

[8] E. Chávez and N. Herrera. Selección dinámica de Índices métricos para consultas de proximidad. In *Actas del X Congreso Argentino de Ciencias de la Computación (CACIC'04)*, pages 389–400, Buenos Aires, Argentina, 2004.

[9] A. Faragó, T. Linder, and G. Lugosi. Fast nearest-neighbor search in dissimilarity spaces. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15(9):957–962, 1993.

[10] L. Micó, J. Oncina, and E. Vidal. A new version of the nearest-neighbor approximating and eliminating search (AESA) with linear preprocessing-time and memory requirements. *Pattern Recognition Letters*, 15:9–17, 1994.