Análisis de la Separabilidad de Clusters y Filtrado de Pivotes para el D-Index

Norma Beatriz Perez, Veronica Gil-Costa, Nora Reyes

Departamento de Informática, Universidad Nacional de San Luis,
+54(2652)424027 — Fax: +54(2652)430224
Ejército de los Andes 950
5700 — San Luis, Argentina.
e-mail:{nbperez,gvcosta,nreyes}@unsl.edu.ar

Resumen En este trabajo implementamos y evaluamos la estructura denominada D-Index, que combina técnicas de clustering con estrategias de búsqueda basada en pivotes para acelerar la ejecución de consultas por similitud en rango y de vecinos más cercanos. Primero mostramos experimentalmente las ventajas de la estructura D-Index. En particular, nos enfocaremos en el problema de la separabilidad entre los objetos, que es un problema abierto y no está tan explorado como el problema de selección de pivotes. Segundo mostramos experimentalmente como la técnica de filtrado de pivotes permite reducir los costos de los algoritmos comparada con la estructura D-Index sin filtrado de pivotes ya que la resolución de consultas sobre este tipo de índice tiende a ser muy costosa por la dificultad que implica le ejecución de la función de similitud.

1. Introducción

El concepto de búsqueda por similitud basado en distancias relativas entre una consulta y un conjunto de objetos de una base de datos se ha vuelto esencial para numerosas y variadas áreas de aplicación, incluyendo recuperación de información multimedia, minería de datos, reconocimiento de patrones, aprendizaje de máquina, compresión de datos y análisis estadístico de los datos, biología computacional (procesamiento de secuencias biológicas, como ADN y proteínas), entre otros. En la Figura 1 se muestra un ejemplo de búsqueda por similitud en un sistema de reconocimiento de huellas dactilares. Como consecuencia, la performance se ha convertido en un punto crítico para el éxito del diseño. Se sabe que la performance es una notable limitación de los programas informáticos y la falta de ella es la principal causa de fracaso, en aplicaciones tales como el almacenamiento de datos, con enormes repositorios de datos heterogéneos. Como en toda aplicación que realiza búsquedas, surge la necesidad de tener una respuesta rápida y adecuada, y un uso eficiente de memoria, lo que hace necesaria la existencia de estructuras de datos especializadas que incluyan estos aspectos. Las búsquedas por similitud (o proximidad) trabajan sobre tipos de datos como por ejemplo conjuntos, cadenas, vectores o estructuras complejas como documentos XML. Las mismas intuitivamente consisten en recuperar objetos similares con

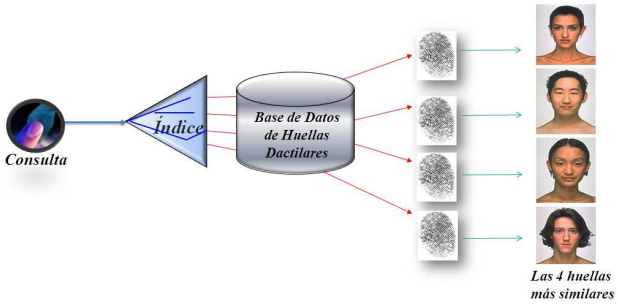

Figura 1. Ejemplo de consulta por similitud sobre huellas dactilares

respecto a un objeto de consulta, de acuerdo a una medida de distancia específica del dominio. Las búsquedas por similitud pueden ser especificadas formalmente por la noción matemática de *espacios métricos*.

1.1. Modelo Formal de Espacios Métricos

Sea $\mathcal{D}$ un universo de objetos, y d una función de distancia total definida como $d : \mathcal{D} \times \mathcal{D} \rightarrow \mathbb{R}^+$. Si la función d cumple con las propiedades:

1. $(\forall x, y \in \mathcal{D})\ d(x, x) = 0 \wedge x \neq y \Leftrightarrow d(x, y) > 0$ *(No Negativa)*
2. $(\forall x, y \in \mathcal{D})\ d(x, y) = d(y, x)$ *(Simétrica)*
3. $(\forall x, y \in \mathcal{D})\ d(x, z) \leqslant d(x, y) + d(y, z)$ *(Desigualdad Triangular)*

entonces el par $\mathcal{M} = (\mathcal{D}, d)$ define un *espacio métrico*. Luego, dado $S \subseteq \mathcal{D}$ existen básicamente dos tipos de búquedas para un objeto de consulta $q \in \mathcal{D}$:

> **Búsqueda por Rango:** Recupera todos los elementos que están a distancia menor o igual a r de un objeto de consulta q, esto es, $(q, r)_d = \{x \in S / \ d(q, x) \leqslant r\}$.
> **Búsqueda de los k Vecinos más Cercanos:** Recupera los k objetos de S más cercanos a un objeto de consulta q dado. Formalmente, $A \subseteq S$ tal que $|A| = k$ y $\forall x \in A,\ \forall y \in (S - A),\ \ d(q, x) \leqslant d(q, y)$.

La *función de distancia* d se considera costosa de calcular (por ejemplo, comparar dos huellas dactilares). Así, es usual definir la complejidad de la búsqueda como el número de evaluaciones de distancia realizadas. Es importante notar que existen otros métodos para buscar sobre espacios D-dimensionales, tales como Kd-tree [1] o R-trees [3]. Sin embargo, estas estructuras no trabajan bien para dimensiones altas. En este trabajo trabajamos sobre espacios métricos generales, aunque las soluciones son también adecuadas para espacios D-dimensionales.

2. Clustering a través de Particionamientos Separables

Las estructuras de datos para espacios métricos se pueden clasificar en técnicas basadas en pivotes o basadas en clustering [3]. El clustering de objetos agrupa

recursivamente los datos en particiones separables de bloques de datos y se puede combinar con estrategias basadas en pivotes a fin de disminuir los costos de E/S. El particionamiento separable de S se hace en base a una función llamada ρ-split, donde ρ, conocido como el *parámetro de separabilidad*, es un número real en el rango $0 \leqslant \rho < d^+$. Definimos formalmente una función ρ-split:

Definición 1. *Dado un espacio métrico $\mathcal{M} = (\mathcal{D}, d)$, una función ρ-split de primer orden $s^{1,\rho}$ es el mapeo $s^{1,\rho} : \mathcal{D} \to \{0, 1, -\}$, tal que para objetos arbitrarios diferentes $x, y \in \mathcal{D}$,*

- $s^{1,\rho}(x) = 0 \wedge s^{1,\rho}(y) = 1 \;\Rightarrow\; d(x, y) > 2\rho$ *(propiedad Separable) y*
- $\rho_2 \geqslant \rho_1 \wedge (s^{1,\rho_2}(x) \neq - \wedge s^{1,\rho_1}(y)) = - \Rightarrow\; d(x, y) > (\rho_2 - \rho_1)$ *(propiedad Simétrica).*

Generalizamos una función ρ-split concatenando t funciones ρ-split de primer orden. Esto se refleja en la siguiente definición.

Definición 2. *Dadas t funciones ρ-split de primer orden $s_1^{1,\rho}, \ldots, s_t^{1,\rho}$ definidas en el espacio métrico $\mathcal{M} = (\mathcal{D}, d)$, una función ρ-split de orden t, $s^{t,\rho} = (s_1^{1,\rho},\ s_2^{1,\rho}, \ldots,\ s_t^{1,\rho}) : \mathcal{D} \to \{0, 1, -\}^t$ es el mapeo, tal que para objetos arbitrarios diferentes $x, y \in \mathcal{D}$,*

- $\forall i\ s_i^{1,\rho}(x) \neq - \wedge \forall j\ s_j^{1,\rho}(y) \neq - \wedge s^{t,\rho} \neq s^{t,\rho}(y) \Rightarrow d(x, y) > 2\rho$ *(propiedad Separable) y*
- $\rho_2 \geqslant \rho_1 \wedge (\forall i\ s_i^{1,\rho_2}(x) \neq - \wedge \exists j\ s_j^{1,\rho_1}(y)) = - \Rightarrow d(x, y) > (\rho_2 - \rho_1)$ *(propiedad Simétrica).*

La siguiente definición transforma en enteros las cadenas obtenidas mediante una función ρ-split, cuyo objetivo es el de obtener un esquema de direccionamiento:

Definición 3. *Dada una cadena $b = (b_1, \ldots, b_t)$ de t elementos $0, 1,$ o $-$, la función $\langle\cdot\rangle : \{0, 1, -\}^t \to [0..\ 2^t]$ es especificada como:*

$$\langle b \rangle = \begin{cases} [b_1, b_2, \ldots, b_t]_2 = \sum_{j=1}^{t} 2^{(j-1)} b_j, & \text{si } \forall j b_j \neq - \\ 2^t, & \text{en otro caso} \end{cases}$$

La idea intuitiva de la Definición 3 consiste en traducir la cadena b en un entero, interpretándolo como un número binario (siempre es menor que 2^t) para todos los elementos que sean distintos de '$-$', en otro caso la función devuelve 2^t. Haciendo uso de las Definiciones 2 y 3, podemos asignar un número i que se encuentre en el rango $0 \leqslant i \leqslant 2^t$ a cada objeto $x \in D$, es decir, la función ρ-split puede agrupar objetos de $S \subset \mathcal{D}$ en $2^t + 1$ subconjuntos disjuntos.

Dada una función ρ-split $s^{t,\rho}$ y un conjunto de objetos S, definimos:

$$S_{[i]}^{t,\rho}(x) = \{\ x \in S \mid \langle s^{t,\rho}(x) \rangle = i \}$$

Llamaremos *conjunto de exclusión* al conjunto de objetos para el cual la función $\langle s^{t,\rho}(x) \rangle$ produce el valor 2^t, mientras que a los primeros 2^t conjuntos (para los cuáles la función $\langle b \rangle$ retorna un valor entre 0 y $(2^t - 1)$) los llamaremos *conjuntos separables.*

Funciones de Particionamiento por Bolas Existen varios tipos diferentes de funciones ρ-split de primer orden, las cuáles son propuestas y analizadas en [5]. La técnica *ball partitioning split* (*bps*) propuesta originalmente en [6], ha demostrado que produce conjuntos de exclusión más pequeños que si usaramos otras técnicas de particionamiento, por eso la empleamos en nuestro trabajo.

Definición 4. *La función ρ-split de particionamiento por bolas de grado 1 se define:*

$$bps^{1,\rho}(o) = \begin{cases} 0 \;\; si \quad d(o,p) \leqslant (d_m - \rho) \\ 1 \;\; si \quad d(o,p) > (d_m + \rho) \\ - \;\; en \; otro \; caso \end{cases}$$

La Figura 2 muestra un ejemplo de una función ρ-split que utiliza la técnica de particionamiento por bolas. La función $bps^{\rho}(o, o_v)$ hace uso de un objeto $o_v \in \mathcal{D}$ y de la distancia mediana d_m para particionar el archivo de datos en tres subconjuntos: $bps^{1,\rho}_{[0]}$, $bps^{1,\rho}_{[1]}$, $bps^{1,\rho}_{[2]}$. En la figura se muestra el resultado de la función $\langle bps^{1,\rho}(x)\rangle$ dando el índice del conjunto al cual pertenece el objeto "o". Más precisamente el resultado de la función *bps* para este ejemplo retorna:

Valor:
- ▶ 0 $\Rightarrow$ para el objeto o_v,
- ▶ 1 $\Rightarrow$ para el objeto o_j,
- ▶ $-$ $\Rightarrow$ para el objeto o_i.

Donde:
- ▶ Conjuntos separables: $bps^{1,\rho}_{[0]}(\mathcal{D})$, $bps^{1,\rho}_{[1]}(\mathcal{D})$,
- ▶ Conjunto exclusión: $bps^{1,\rho}_{[-]}(\mathcal{D})$

Debido al hecho de que esta función produce dos conjuntos separables, la llamaremos *función bsp binaria*. Los objetos que se encuentran en el conjunto de exclusión son retenidos para un procesamiento futuro.

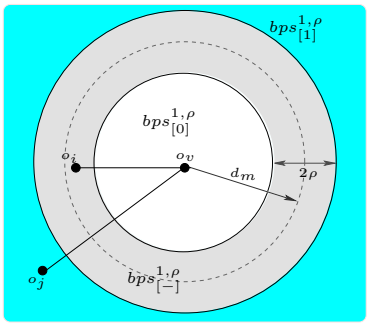

Figura 2. Función de particionamiento *bps*.

3. D-Index

El D-Index [4] es una estructura que combina novedosas técnicas de clustering con técnicas basadas en pivotes, para acelerar la ejecución de consultas por rango o por K-NN para grandes colecciones de datos.

La arquitectura de almacenamiento del D-Index está basado en un arreglo 2-dimensional de buckets usados para almacenar objetos de datos. Usa una función ρ-split en cada nivel para separar los objetos en distintas particiones. Cada nivel se divide a su vez en *buckets* y uno de estos buckets, denominado *bucket de exclusión*, mantiene aquellos objetos que no pertenecen a ninguno de los buckets del nivel que está siendo analizado. Es importante notar que las funciones ρ-split pueden tener diferente orden, típicamente decreciendo con el nivel, permitiendo que la estructura D-Index tenga niveles con un número diferente de buckets. Más precisamente, la estructura del D-Index puede ser definida como sigue.

Definición 5. *Dado un conjunto de objetos $S \in \mathcal{D}$, el índice de búsqueda por similitud de* nivel *h $DI^{\rho}(S, m_1, m_2, \cdots, m_h)$ con los buckets en cada nivel separables hasta 2ρ, es determinado por h funciones ρ-split independientes $s_i^{m_i,\rho}$, $(i = 1, 2, \cdots, h)$ la cual genera:*

$$\text{Bucket de Exclusión} \quad E_i = \begin{cases} S^{m_1,\rho}_{1[2^{m_1}]}(S) & \text{si} \quad i = 1 \\ S^{m_i,\rho}_{i[2^{m_i}]}(E_i - 1) & \text{si} \quad i > 1 \end{cases}$$

$$\text{Bucket Separables } \{B_{i,0}, B_{i,1}, \cdots, B_{i,2^{m_i}-1}\} = \begin{cases} \{S^{m_1,\rho}_{1[\cdot]}(S)\} & \text{si} \quad i = 1 \\ \{S^{m_i,\rho}_{i[\cdots]}(E_i - 1)\} & \text{si} \quad i > 1 \end{cases}$$

Desde el punto de vista de la estructura, se puede ver a los buckets organizados como el siguiente arreglo 2-dimensional que consiste de $1 + \sum_{i=1}^{h} 2^{m_i}$ elementos.

$$\begin{array}{|l|} \hline B_{1,0}, B_{1,1}, \cdots, B_{1,2^{m_1}-1} \\ B_{2,0}, B_{2,1}, \cdots, B_{2,2^{m_2}-1} \\ \vdots \\ B_{h,0}, B_{h,1}, \cdots, B_{h,2^{m_h}-1}, E_h \\ \hline \end{array}$$

Todos los buckets separables son incluidos, pero solo el bucket de exclusión E_h está presente en la estructura. Los buckets de exclusión $E_{i<h}$ son re-particionados recursivamente en el nivel $i + 1$. Entonces, para cada fila i (es decir, nivel D-Index), 2^{m_i} buckets son separables hasta 2ρ. Así se asegura que no existen dos buckets en el mismo nivel i y que ambos contengan objetos relevantes para alguna consulta por similitud con radio $r_x \leqslant \rho$.

Es importante observar que hay un *trade-off* entre el número de buckets separables y el número de objetos en el bucket de exclusión. Mientras más buckets separables hay, más grande es el número de objetos en el bucket de exclusión.

4. Técnica de Filtrado por Pivotes Aplicada al D-Index

Con el próposito de optimizar nuestros tiempos de respuesta ante las consultas, la idea es aplicar la técnica de filtrado por pivotes, que fue explicada en

[7]. Partimos de un conjunto de objetos $S = \{o_1, o_2, \ldots, o_n\}$, donde $S \in \mathcal{D}$ y sea el conjunto de pivotes $P = \{p_1, p_2, \ldots p_t\}$. Definimos una estructura de pivotes como una matriz $\mathcal{T}$ de $n \times t$, esto es $\mathcal{T}[1 \ldots n, 1 \ldots t]$. En $\mathcal{T}$ se almacenan las distancias $d(o_i, p_j)$, es decir, en cada fila almacena las distancias entre el objeto o_i y cada uno de los pivotes p_j.

A continuación se define formalmente la distancia máxima por pivotes como:

$$\mathbf{D}(x, q) = \max_{1 \leq i \leq t} |d(x, p_i) - d(q, p_i)| \quad (1)$$

La función $\mathbf{D}$ está definida por el pivote que maximice la diferencia. Se puede demostrar que $\mathbf{D}(x, q) \leq d(x, q), \forall x, q \in \mathcal{D}$. En tiempo de búsqueda, se calcula $\mathbf{D}(o_i, q)$ para todos los objetos de S. Sin embargo, y dado que $\mathbf{D}(x, q) \leq d(x, q)$, si $r < \mathbf{D}(o_i, q)$, entonces sabemos que $r < d(o_i, q)$ sin necesidad de computar d. Luego, se aplica d sobre los objetos que no pudieron descartarse con el primer paso. Es importante notar que usando más pivotes, se incrementa la probabilidad de excluir un objeto, sin realmente computar su distancia con respecto a q.

5. Evaluación y Comparación de la Performance

Realizamos nuestra evaluación experimental con dos bases de datos: la primera es un conjunto de imágenes representadas como histogramas de colores, que son vectores de dimensión 112, y que consta con 111.682 objetos. La segunda es un conjunto de 40.150 imágenes de la NASA, representadas como vectores de dimensión 20. Las consultas fueron realizadas con 5.000 objetos no pertenecientes a la base de datos y se utilizaron radios que recuperan el 0,01 %, 0,02 %, ..., 0,1 % de la base de datos para las consultas por rango.

5.1. Variando el parámetro de Separabilidad ρ

Los experimentos fueron realizados para distintos valores de ρ como se puede observar en la Figura 3 (para histogramas de color) y en la Figura 4 (para imágenes de la NASA) para distintas organizaciones del D-Index. El Cuadro 1 muestra los distintos parámetros que se consideraron al momento de llevar a cabo los experimentos, para el caso de histogramas de color. La primera columna indica la cantidad de niveles generados en la estructura. La segunda columna muestra el valor de ρ utilizado en cada nivel y la tercera columna indica la cantidad de elementos en el bucket de exclusión del último nivel. El tamaño del bucket de exclusión aumenta a medida que ρ crece, porque el anillo de separabilidad (de grosor 2ρ) de la partición contiene más elementos a medida que es más grueso. Sin embargo, un anillo delgado puede provocar que una mayor cantidad de buckets separables sean examinados. Así que la elección del ρ óptimo no es trivial.

La Figura 3 muestra el número de evaluaciones de distancias obtenidos para diferentes valores del parámetro ρ a medida que varía el radio de búsqueda. Los resultados se presentan para diferentes diseños de la estructura, modificando el

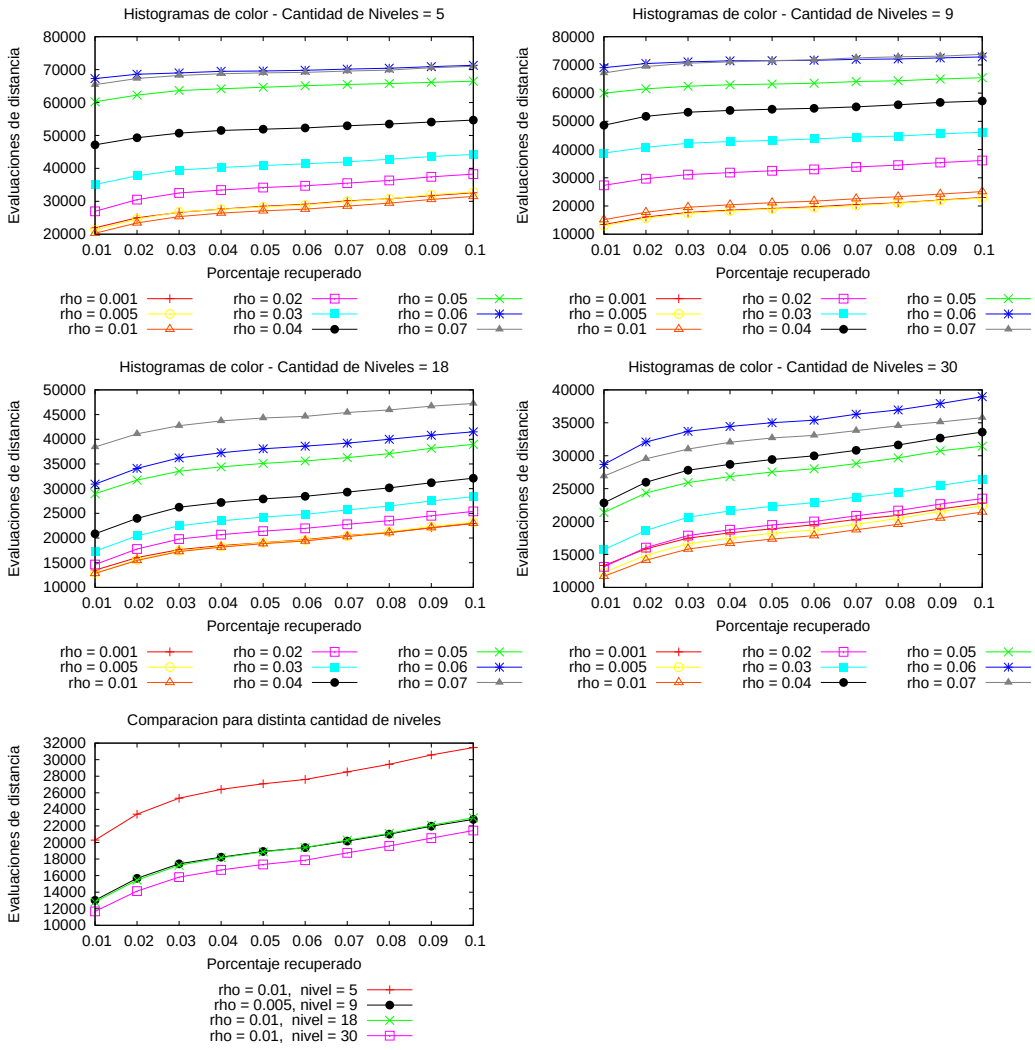

Figura 3. Performance del D-Index de la base de dato histogramas de color para consultas por rango.

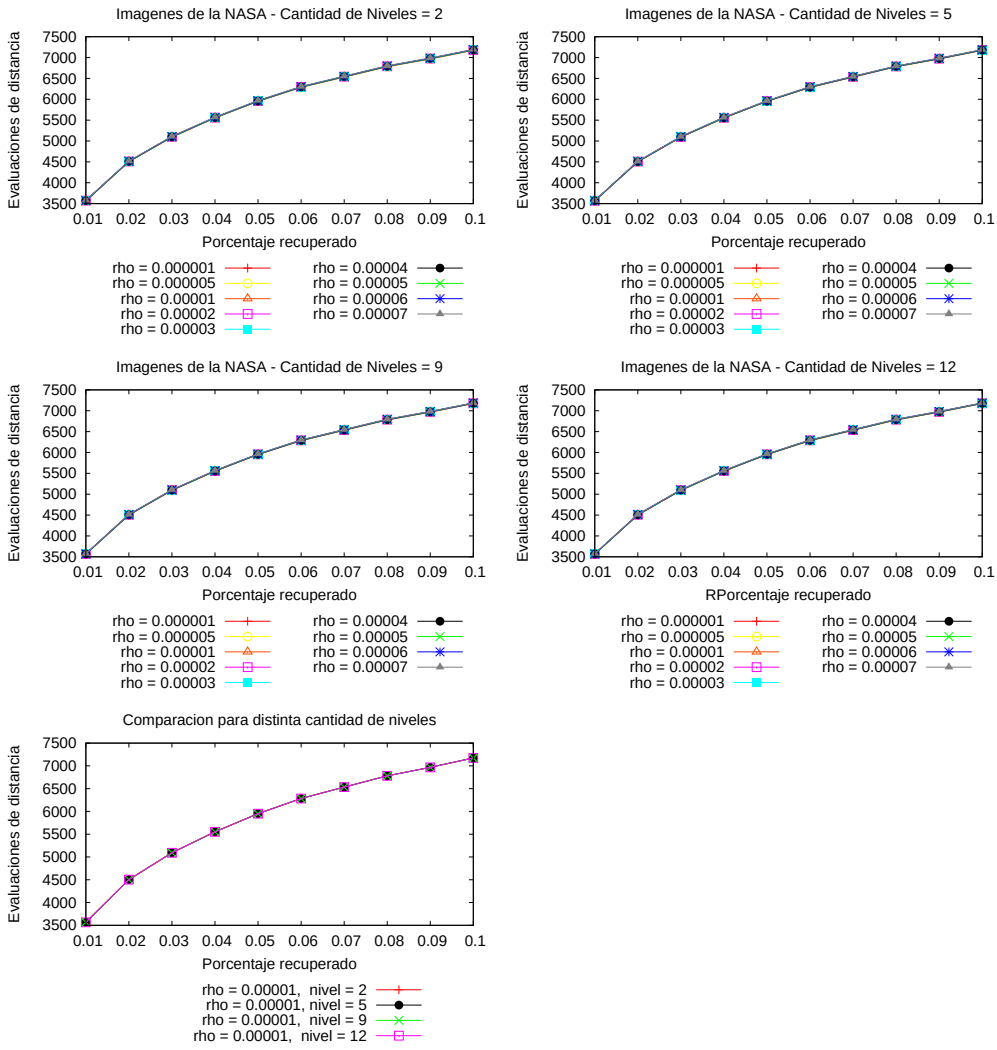

Figura 4. Performance del D-Index de la base de imágenes de la NASA para consultas por rango.

Cuadro 1. Parámetros con los que se realizaron los experimentos.

Niveles	Grado por Nivel	Tamaño bucket exclusión, variando ρ								
		0,001	0,005	0,01	0,02	0,03	0,04	0,05	0,06	0,07
5	8,7,6,5,4	2	236	2.848	16.172	32.840	50.417	67.191	76.625	78.094
9	14,13,12,11,10,9,8,7,6	2	211	3.492	24.287	43.509	58.323	62.283	78.419	78.957
18	16,15,14,13,12,11,10,9, 8,7,6,5,4,4,3,2,2,1	1	2	8	863	4.594	11.088	29.607	32.142	43.714
30	16,16,16,15,15,15,14,14, 13,13,12,12,11,11,10,10,9, 9,8,8,7,7,6,5,4,4,3,2,2,1	1	1	11	754	5.816	18.909	15.043	30.172	23.256

número de niveles. Como se puede observar, el mejor valor de ρ obtenido en las primeras cuatro gráficas (búsqueda por rango) fue $0,01$ excepto cuando la cantidad de niveles es 9, en donde el mejor valor de ρ fue $0,005$. La quinta gráfica compara el desempeño del D-Index con diferentes niveles y el mejor ρ reportado. Esta estructura presenta un mejor rendimiento con 30 niveles y un $\rho = 0{,}01$.

Los resultados para imágenes de la NASA se muestran en la Figura 4. Allí puede observarse que el mejor valor de ρ es $0,00001$. No existe una gran diferencia entre los resultados obtenidos con los valores de ρ probados. Para obtener una idea más precisa del valor óptimo de ρ, tomamos una muestra aleatoria de pares de objetos de la base de datos para calcular la distancia promedio entre elementos. El resultado es que la distancia promedio para la base de datos de histogramas de color es $0,42$, lo cual es aproximadamente 40 veces más grande que el mejor valor $\rho = 0,01$. Para las imágenes de la NASA, la distancia promedio es $1,41$. Esto parece indicar que los valores óptimos de ρ deben ser significativamente más pequeños que la distancia promedio entre elementos de la base de datos, lo que puede ayudar a elegir un buen valor de ρ sin tener que construir el índice múltiples veces, lo cual es costoso.

5.2. Experimentos con Filtrado de Pivotes

Para los experimentos con filtrado de pivotes, nuestros experimentos indican que la estructura se vuelve menos sensible, a la distribución de niveles. Por eso decidimos mostrar los resultados para 5 niveles. La cantidad de pivotes por nivel es: $8, 7, 6, 5$ y 4. En el Cuadro 2 se muestra los resultados que se obtuvieron para los distintos radios de búsqueda. En el cuadro, como puede verse, la cantidad de evaluaciones de distancia al usar el filtrado de pivotes es mucho menor que la cantidad de evaluaciones obtenidas en los experimentos de la Figura 3. Luego se puede decir que el mejor ρ ahora es $0,001$, a diferencia de antes que era $0,01$. Notar que tener buckets de exclusión de tamaño grande implica que se realice una mayor cantidad de evaluaciones de distancia, haciendo que la estructura sea muy sensible a la elección del parámetro ρ, que es quien determina el tamaño del bucket de exclusión.

A continuación analicemos qué pasa en las búsquedas. De los resultados vemos que el mejor rendimiento se logra con un $\rho = 0,001$, lo que parece indicar

Cuadro 2. Resultados experimentales del índice D-Index secuencial con el filtrado de pivotes

ρ	*Tamaño Bucket Exclusión*	*Cantidad de Niveles*	*Rango de Evaluaciones de Distancia*
0.001	2	5	[1.331 - 5.742]
0,005	236	5	[1.571 - 6.274]
0,01	2.848	5	[2.976 - 7.748]
0,02	16.172	5	[12.576 - 18.287]
0,03	32.840	5	[24.277 - 29.191]
0,04	50.417	5	[38.042 - 42.693]
0,05	67.191	5	[55.235 - 59.227]
0,06	76.625	5	[64.139 - 67.121]
0,07	78.094	5	[61.814 - 66.171]

que, con el filtrado de pivotes, la separabilidad entre buckets no es tan importante para mejorar la eficiencia de búsqueda.

6. Conclusiones y Trabajos Futuros

En este trabajo implementamos y evaluamos la estructura D-Index [4], enfocándonos principalmente en el análisis de separabilidad y el efecto producido por el filtrado de pivotes. Como se pudo observar a través de los experimentos (para distintas colecciones), la estructura muestra ser efectiva, aunque es muy sensible a los parámetros y a las distintas combinaciones de los mismos. También es importante notar que aplicar una técnica de selección de pivotes [2] mejoraría el comportamiento de nuestra estructura. Todo esto ha sido tenido en cuenta para una implementación futura del D-Index, siendo fácilmente adaptable a nuestro código.

Actuamente estamos estudiando diferentes técnicas de selección de pivotes que pueden afectar el rendimiento del índice. También estamos estudiando implementaciones paralelas que pueden ser fácilmete aplicadas al D-Index, por su estructura de hashing multinivel.

Referencias

1. J. Bentley. Multidimensional binary search trees in database applications. pages 333–340. IEEE Trans. on Software Engineering, 1979.
2. B. Bustos, G. Navarro, and E. Chávez. Pivot selection techniques for proximity searching in metric spaces. In *Proc. of SCCC'01*. IEEE Computer Society, 2001.
3. E. Chávez, G. Navarro, R. Baeza-Yates, and Marroquín J. Searching in metric spaces. *ACM Computing Surveys*, 33(3):273–321, 2001.
4. V. Dohnal. *Indexing structure for searching in metric spaces.* PhD thesis, Masaryk University, 2004.
5. V. Dohnal, C. Gennaro, P. Savino, and P. Zezula. Separable split in metric data sets. In *Proc. of 9-th ISADS*, pages 45–62. LCM Selecta Group—Milano, 2001.
6. P. N. Yianilos. Excluded middle vantage point forests for nearest neighhor search in general metric spaces. In *DIMACS Implementation Challenge, ALENEX'99*, 1999.
7. P. Zezula, G. Amato, V. Dohnal, and M. Batko. *Similarity Search: The Metric Space Approach.* Springer, 2006.