Listas de Clusters usando Centros Espacialmente Dispersos para Búsquedas por Similaridad en espacios Métricos.

Roberto Uribe-Paredes[1,2], Roberto Solar[1], and Claudio Márquez[1]

[1] Depto. de Ingeniería en Computación*,
Universidad de Magallanes, Chile
[2] Grupo de Bases de Datos - UART,
Universidad Nacional de la Patagonia Austral, Río Turbio, Argentina
E-mail: ruribe@ona.fi.umag.cl

Abstract. *Sparse Spatial Selection* is a new pivot-based structure for similarity search in metric spaces. It is a array-type structure which has shown a good search performance when compared with other selection methods.
This work describes the building of a new metric structure. It is a tree-type structure and born as a recursive application from *Cluster List* and using *SSS* as a general method for center selection. Preliminaries experimental results show that it presents a better performance, in terms of the number of distance evaluations than original *Cluster List* and other well-known structures.
Keywords: databases, data estructures, algorithms, metric spaces, similarity queries.

Resumen El *Sparse Spatial Selection* es una nueva estructura basada en pivotes para búsqueda por similaridad en espacios métricos. Esta estructura es del tipo arreglo y ha demostrado buen rendimiento durante la búsqueda comparado con otros métodos de selección.
El presente trabajo describe la construcción de una nueva estructura métrica. Ésta es una estructura de tipo árbol y nace de la aplicación recursiva de *Listas de Clusters* usando *SSS* como un método general para la selección de centros. Resultados experimentales preliminares demuestran que tiene mejor desempeño, en términos de evaluaciones de distancia, que la estrutura *Lista de Clusters* original y otras estructuras conocidas.
Palabras claves: bases de datos, estructuras de datos, algoritmos, espacios métricos, consultas por similaridad.

1. Introducción

1.1. Antecedentes

Uno de los problemas de gran interés en ciencias de la computación es el de "búsqueda por similaridad", es decir, encontrar los elementos de un conjunto más similares a una muestra. Esta búsqueda es necesaria en múltiples aplicaciones, como ser en reconocimiento de voz e imagen, compresión de video, genética, minería de datos, recuperación de información, etc. En casi todas las aplicaciones la evaluación de la similaridad entre dos elementos es cara, por lo que usualmente se trata como medida del costo de la búsqueda la cantidad de similaridades que se evalúan.

Interesa el caso donde la similaridad describe un espacio métrico, es decir, está modelada por una función de distancia que respeta la desigualdad triangular. En este caso, el problema más común y difícil es en aquellos espacios de "alta dimensión" donde el histograma de distancias es concentrado, es decir, todos los objetos están más o menos a la misma distancia unos de otros.

El aumento de tamaño de las bases de datos y la aparición de nuevos tipos de datos sobre los cuales no interesa realizar búsquedas exactas, crean la necesidad de plantear nuevas estructuras para búsqueda por similaridad o búsqueda aproximada, esto en busca de superar los problemas de las

* Parcialmente financiado por los proyectos Fondecyt 1060776, Conicyt; programa de investigación PR-F1-002IC-06, Universidad de Magallanes, Chile.

estructuras existentes hasta ahora. Asimismo, se necesita que dichas estructuras sean dinámicas, es decir, que permitan agregar o eliminar elementos sin necesidad de crearlas nuevamente. Así también, las aplicaciones reales requieren que dichas estructuras permitan ser almacenadas en memoria secundaria eficientemente, como también que posean métodos optimizados para reducir los costos de accesos a disco. Adicionalmente, el método elegido en las estructuras para la elección de centros y pivotes resulta relevante al momento de descartar elementos durante la búsqueda.

1.2. Marco teórico

La similaridad se modeliza en muchos casos interesantes a través de un espacio métrico, y la búsqueda de objetos más similares a través de una búsqueda por rango o de vecinos más cercanos.

Definición 1 (*Espacios Métricos*): Un espacio métrico es un conjunto X con una función de distancia $d : X^2 \rightarrow R$, tal que $\forall x, y, z \in X$,

1. $d(x, y) \geq 0$ *and* $d(x, y) = 0$ *ssi* $x = y$. *(positividad)*
2. $d(x, y) = d(y, x)$. *(Simetría)*
3. $d(x, y) + d(y, z) \geq (d(x, z)$. *(Desigualdad Triangular)*

Definición 2 (*Consulta por Rango*): Sea un espacio métrico *(X,d)*, un conjunto de datos finito $Y \subseteq X$, una consulta $x \in X$, y un rango $r \in R$. La consulta de rango alrededor de x con rango r es el conjunto de puntos $y \in Y$, tal que $d(x, y) \leq r$.

Definición 3 (*Los k Vecinos más Cercanos*): Sea un espacio métrico *(X,d)*, un conjunto de datos finito $Y \subseteq X$, una consulta $x \in X$ y un entero k. Los k vecinos más cercanos a x son un subconjunto A de objetos de Y, donde la $|A| = k$ y no existe un objeto $y \in A$ tal que *d(y,x)* sea menor a la distancia de algún objeto de A a x.

El objetivo de los algoritmos de búsqueda es minimizar la cantidad de evaluaciones de distancia realizadas para resolver la consulta. Los métodos para buscar en espacios métricos se basan principalmente en dividir el espacio empleando la distancia a uno o más objetos seleccionados. El no trabajar con las características particulares de cada aplicación tiene la ventaja de ser más general, pues los algoritmos funcionan con cualquier tipo de objeto [9].

Existen distintas estructuras para buscar en espacios métricos, las cuales pueden ocupar funciones discretas o continuas de distancia. Algunos son BKTree [4], MetricTree [15], GNAT [2], VpTree [17], FQTree [1], MTree [10], SAT [12], Slim-Tree [14], EGNAT [16].

Algunas de las estructuras anteriores basan la búsqueda en pivotes y otras en clustering. En el primer caso se seleccionan pivotes del conjunto de datos y se precalculan las distancias entre los elementos y los pivotes. Cuando se realiza una consulta, se calcula la distancia de la consulta a los pivotes y se usa la desigualdad triangular para descartar candidatos.

Los algoritmos basados en clustering dividen el espacio en áreas, donde cada área tiene un *centro*. Se almacena alguna información sobre el área que permita descartar toda el área mediante sólo comparar la consulta con su centro. Los algoritmos de clustering son los mejores para espacios de alta dimensión, que es el problema más difícil en la práctica.

Existen dos criterios para delimitar las áreas en las estructuras basadas en clustering, *hiperplanos* y *radio cobertor* (*covering radius*). El primero divide el espacio en particiones de *Voronoi* y determina el hiperplano al cual pertenece la consulta según a qué centro corresponde. El criterio de radio cobertor divide el espacio en esferas que pueden intersectarse y una consulta puede pertenecer a más de una esfera.

Definición 4 (*Diagrama de Voronoi*): Considérese un conjunto de puntos $\{c_1, c_2, \ldots, c_n\}$(centros). Se define el diagrama de Voronoi como la subdivisión del plano en n áreas, una por cada c_i, tal que $q \in$ al área c_i sí y sólo sí la distancia euclidiana $d(q, c_i) < d(q, c_j)$ para cada c_j, con $j \neq i$.

Uno de los problemas que provoca que muchas veces buenas estructuras arrojen malos resultados es la elección poco afortunada de centros o pivotes. En este sentido, este trabajo propone el uso de un nuevo método denominado *Sparse Spatial Selection* (*SSS*) [3,13], como un método general para selección de centros o pivotes. Para la aplicación de este método, se eligió inicialmente la estructura denominada *Lista de Clusters* [8], que es de las estructuras basadas en clustering, del tipo arreglo y utiliza el radio cobertor para descartar centros durante la búsqueda.

Para este artículo se seleccionó, para la realización de las pruebas, un espacio métrico consistente en un diccionario de palabras en castellano de 86.061 objetos, donde la distancia utilizada es la *distancia de edición*, la cual entrega como resultado el número mínimo de inserciones, eliminaciones o reemplazos de caracteres, necesarios, para que una palabra sea igual a otra. El segundo es un espacio de vectores de coordenadas reales de dimensión 10 generados con distribución de *Gauss* con media 1 y varianza 0.1 cuya cantidad de objetos es de 100,000, para este espacio se utilizó la *distancia Euclidiana*. Para la búsqueda se creó la estructura con el 90 % de los datos y se reservó el 10 % como consultas.

2. Selección de Pivotes y Centros

En particular, la elección de pivotes o centros según sea la estructura resulta relevante para obtener un mayor rendimiento durante la búsqueda, lo que queda demostrado empíricamente en [5].

Distintas estrategias han sido propuestas como adecuadas para la elección de pivotes. En [11] se propone seleccionar como pivotes aquellos objetos que maximicen la suma de las distancias a los pivotes ya seleccionados. En [2,17] se siguen heurísticas que tratan de seleccionar pivotes que estén lejanos entre sí. En [5] se presenta un criterio de comparación de la eficiencia entre dos conjuntos de pivotes, así también se presentan varias estrategias de selección de conjuntos donde se usa el criterio de eficiencia anterior.

El método denominado *Sparse Spatial Selection* (*SSS*), es una nueva técnica propuesta para la selección de pivotes, ésta fue originalmente implementado sobre una estructura del tipo arreglo y usa la desigualdad triangular para discriminar objetos durante la búsqueda. Dicho método se comportó igual o mejor en términos de eficiencia que los propuestos en [5], con la ventaja adicional que no se requiere determinar a priori el número de pivotes necesarios.

2.1. *Sparse Spatial Selection (SSS)*

SSS es un método de selección de pivotes en el que escoge un conjunto dinámico de pivotes bien distribuidos en el espacio, lo que permite descartar más objetos al momento de realizar una búsqueda.

Sea $(\mathbb{X}, d)$ un espacio métrico, $\mathbb{U} \subset \mathbb{X}$ una colección de objetos y M la distancia entre los dos objetos más alejados. En un principio el conjunto de pivotes está formado por el primer objeto de la colección. Luego, para cada elemento de la colección se verifica si está a una distancia mayor o igual a $M * \alpha$ de los pivotes seleccionados, si es así, se agrega al conjunto de pivotes, siendo α una constante cuyos valores están cercanos a 0,4[13]. La figura 1 muestra la obtención de los pivotes en un espacio cualquiera.

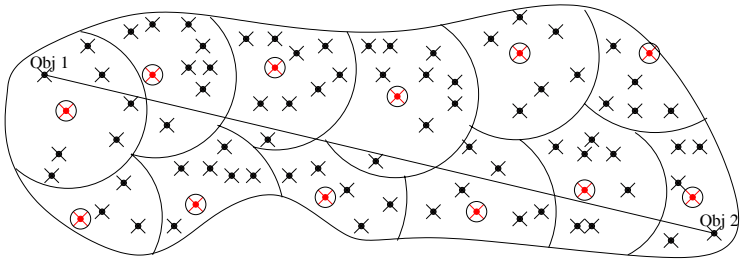


Figura 1. Los objetos encerrados en círculos representan los centros seleccionados como pivotes y M se define como la distancia entre Obj 1 y Obj 2, los más alejados en el espacio.

En [13] la estructura original es basada en pivotes y del tipo arreglo. Su construcción es similar a las estructuras *FQA*[7] y *Spaghettis*[6], pero se diferencia en la forma de elegir los pivotes y en la manera de buscar. Básicamente se tiene un arreglo donde la cantidad de filas es la cantidad total de objetos en la Base de Datos y la cantidad de columnas es el número de pivotes.

En el presente trabajo se considera que el *SSS* es básicamente un método de selección de pivotes, por lo que puede ser aplicado a cualquier estructura, independiente del tipo y de los criterios para delimitar áreas. Se considera también, que es posible construir una estructura plenamente basada en el *SSS*, es decir, una estructura que puede ajustarse completamente al espacio métrico sobre el cual es implementada.

3. *Lista de Clusters*

La ***Lista de Clusters*** [8]es una estructura basada en Clustering o particiones compactas, la cual es muy similar a una Lista Enlazada. Diseñada para tener un buen desempeño en espacios de altas dimensiones.

En la *Lista de Clusters* se selecciona un centro c perteneciente a la base de datos X y un radio r el cual determina la fracción del espacio que abarca la esfera (c, r) definida como el subconjunto de elementos de X los cuales están a una distancia no mayor a r del centro c. Luego se define como I a los elementos que están dentro de la esfera de centro c también llamado *Bucket*, y E definido como el resto de los elementos externos a la esfera de centro c. Este proceso se repite recursivamente. En consecuencia se obtiene una lista compuesta por un centro, un radio y un Bucket (c,r,I) denominado Cluster.

Comparado con otros algoritmos de Clustering, la *Lista de Clusters* solamente usa el criterio de radio cobertor y no áreas como en el Voronoi-Tree. También es posible ver la *Lista de Clusters* como un caso particular de Voronio-tree o un M-tree, considerando I y E como los sub-árboles izquierda y derecha de raíz c, con las diferencias de que las estructuras recién mencionadas tratan de construir un árbol balanceado y que además poseen estructuras internas, en cambio la *Lista de Clusters* es extremadamente desbalanceada y no poseen estructura interna alguna.

3.1. Construcción

La estructura *Lista de Clusters* se construye según el algoritmo 1, donde U que corresponde a los datos no insertados en la lista:

Algoritmo 1 $ContruirLC(U)$

if $U = \phi$ **then**
  retorna una lista vacia
end if
Se Selecciona $c \in U$
Se Selecciona un radio r
$I \longleftarrow \{\ u \in U - c\ ,\ d(c,u) \leq r\ \}$
$E \longleftarrow U - I$
retorna (c, r, I)

La estructura de datos construida debería ser simétrica, pero no lo es. El primer centro escogido tiene preferencia sobre los centros subsecuentes por lo que se provoca solapamiento entre clusters. La figura 2 lo ilustra. Todos los elementos que están dentro del cluster del primer centro (c1 en la figura) se guardan en su Bucket I, a pesar de eso ellos pueden quedar también dentro de los Buckets I de centros subsecuentes (c2, c3, etc. figura 2).

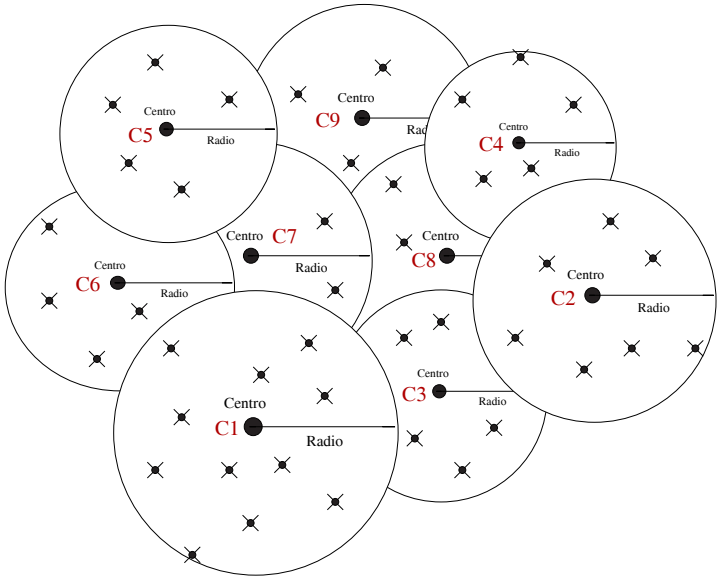


Figura 2. Zonas de influencia de 9 Clusters construidos según el orden : C1, C2, C3, C4, C5, C6, C7, C8 y C9.

3.2. Búsqueda

Dado la asimetría de esta estructura, también la búsqueda se puede reducir si la consulta esta totalmente contenida en el Cluster, por lo cual no se necesita considerar E porque por construcción todos los elementos que están dentro de la esfera de consulta han sido insertados en I. La búsqueda se muestra en el algoritmo 2, en el cual se aplica una consulta q y un radio de búsqueda r sobre la Lista de Cluster L :

Algoritmo 2 *BusquedaLC*(L,q,r)

if L *es vacia* **then**
  Se retorna
end if
$L = (c, r_c, I)$
Se Calcula $d(c, q)$
if $d(c, q) \leq r$ **then**
  c *es un resultado*
end if
if $d(c, q) \leq r_c + r$ **then**
  Se debe buscar exhaustivamente en I
end if
if $d(c, q) > r_c - r$ **then**
  Se debe seguir buscando en (E, q, r)
end if

Esta es una característica esencial ausente en otros algoritmos de Clustering, donde la búsqueda necesita entrar en todos los Clusters que son intersectados por la esfera de consulta. En esta estructura la búsqueda sobre los Cluster restante puede ser cancelada en cuanto la esfera de consulta este totalmente contenida en un Cluster. En la figura 3 se puede apreciar tres casos de consultas sobre un Cluster, en el caso de $q1$ se debe considerar el Bucket actual y el resto de los Clusters, para el caso de $q2$ se debe hacer la búsqueda sólo en este Bucket y para $q3$ evitamos la búsqueda en el Bucket actual.

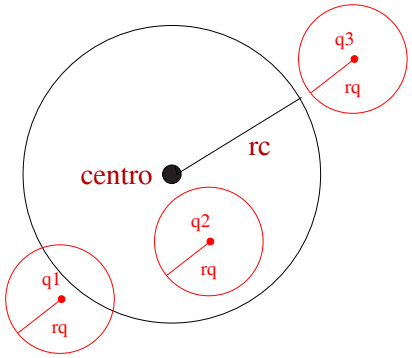


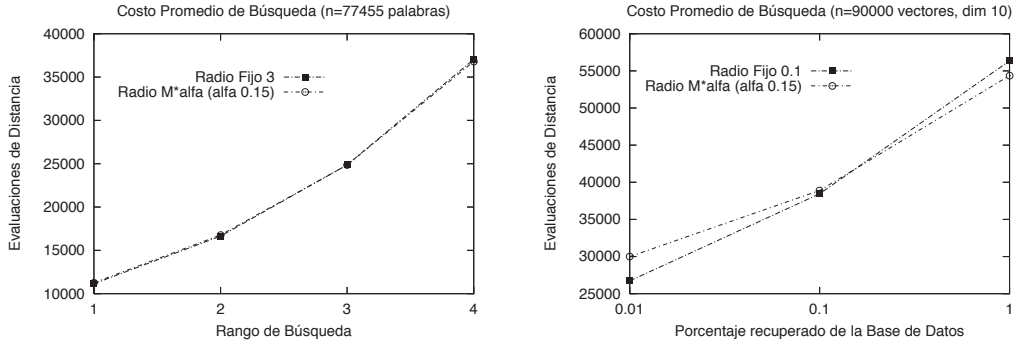
Figura 3. Tres casos de consultas sobre un centro de Cluster.

En las características generales de esta estructura no se especifica cómo se seleccionan los centros y radios en cada punto del algoritmo de construcción, ya que esto se relaciona con la eficacia y no a la exactitud de la estructura de datos. Una buena selección de centros podría ser más bien costosa, además debemos tomar en cuenta que se debe seleccionar un buen radio para cada centro de cluster en cada punto de la construcción debido a que el espacio va cambiando en la medida que los objetos son insertando en la Lista. En este sentido resulta interesante aplicar el método SSS para la selección de centros versus la opción de selección aleatoria. La estructura posee dos alternativas de construcción, la primera es seleccionar un radio fijo para todos los Clusters de la lista y la segunda es seleccionar un tamaño fijo para todos los Clusters de la lista, *Cluster de Radio Fijo* y *Cluster de Tamaño Fijo* respectivamente.

4. *Lista de Clusters* y *Sparse Spatial Selection*

Durante la construcción, la *Lista de Cluster* selecciona inicialmente un centro y un radio, por lo que resulta natural elegir como radio $M * \alpha$ y cada centro usando el algoritmo SSS, es decir, los centros son elegidos si están ubicados a una distancia $M * \alpha$ de todos los centros anteriores. SSS puede ser utilizado para radio fijo, como para tamaño fijo, sin embargo, pruebas preliminares determinaron un comportamiento superior en listas de cluster con radio fijo. El cálculo del valor de M se realiza sobre todos los objetos de la base de datos, sin embargo, esto es un proceso off-line y en este trabajo no se lo consideró como costo de construcción.

La figura 4 muestra el comportamiento de la estructura durante la búsqueda. Los gráficos son para la estructura Lista de Clusters en su versión de radio fijo versus la alternativa de selección de centros usando SSS y radio $M * \alpha$. En ambos casos, los gráficos corresponden a los mejores valores encontrados para radio fijo y a los mejores valores para α. Se puede observar que la diferencia es ínfima, sobre todo en el caso del espacio de palabras, ello es debido a que la función de distancia es discreta, por lo que el mejor radio, es muy parecido al mejor α. En el caso del espacio de vectores, existe una pequeña mejora al aumentar los rangos de búsqueda, en este experimento, el mejor radio es aquel que recupera el 0,1 % de los datos.



(a) Diccionario Español (b) Gauss Dimensión 10

Figura 4. Costos promedios de Búsqueda Lista de Cluster de Radio Fijo vs Lista de Cluster con SSS

4.1. *Lista de Cluster* Recursiva

En los gráficos mostrados anteriormente no se logra ver una mejora importante en el desempeño de la estructura. Esto resulta así, dado que se seleccionaron los mejores métodos para ambos experimentos. El mejor radio fijo y α fueron obtenidos experimentalmente.

Considerando que el espacio es dividido una sola vez en N partes, es posible aplicar esa división en los subespacios generados, es decir, cada cluster de la estructura puede ser a su vez una *Lista de Clusters*. Entonces, una segunda alternativa de construcción es aplicar recursivamente el proceso de construcción sobre cada cluster formado en la estructura original. Para la construcción de esta estructura se utiliza el método SSS.

Finalmente, lo que se obtiene es una estructura de tipo árbol donde se seleccionan los centros espacialmente dispersos, usando un radio de $M * \alpha$.

Cada cluster de la estructura original representa un subespacio con características distintas al original, de hecho los rangos de dicho espacio son mucho menores. Esto puede ser considerado una ventaja, ya que al aplicar recursivamente el método SSS usando el M original, el espacio quedaría sobredimensionado, provocando baja en la eficiencia al interior de la estructura. Ahora, es posible calcular nuevamente el M para el nuevo subespacio, pero implicaría un costo demasiado elevado durante la construcción. Sin embargo, es posible utilizar el mismo radio cobertor del subespacio para calcular un M aproximado, sin pagar costos adicionales.

El radio cobertor es la distancia desde el centro a su elemento más alejado, por lo que se garantiza que M siempre será menor o igual a $2 * rc$ (dos veces el radio cobertor). Esto puede ser utilizado cada vez que se realiza el proceso de construcción recursivamente.

La utilización recursiva del método SSS sobre cada cluster, modificando el valor de M, provoca que la estructura se vaya adaptando siempre a la nueva forma del espacio. Lo anterior implica que la cantidad de nodos en el árbol será dinámica, es decir, los nodos usualmente no tendrán la misma cantidad de objetos.

Finalmente, el proceso termina cuando el cluster tiene una cantidad de datos inferior a una cota determinada, por ejemplo a una página de disco.

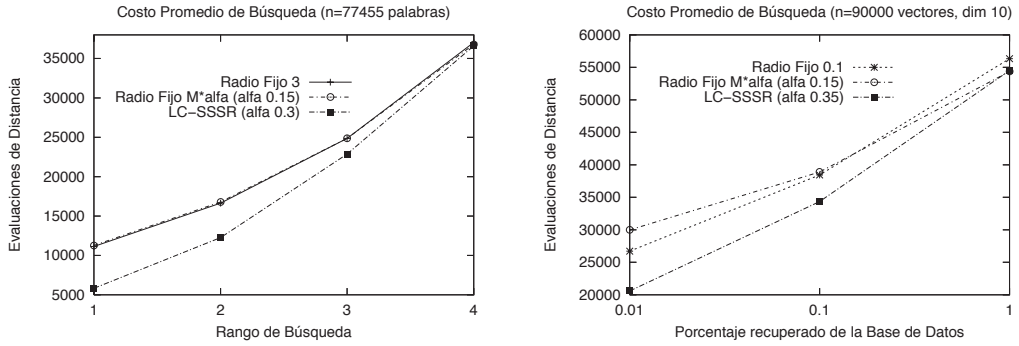
4.2. Resultados Experimentales

En las figuras 5 y 6 se pueden observar resultados interesantes sobre el comportamiento de la nueva estructura (identificada como *LC-SSSR)*. En el espacio de Gauss, los valores graficados corresponden a los rangos que permiten recuperar el 0,01, el 0,1 y el 1 %. La figura 5 corresponde a mejores versiones de Lista de Cluster de radio fijo y Lista de Cluster con SSS versus Lista de Cluster con SSS y Recursiva

LC-SSSR). En este experimento se puede ver claramente que la nueva estructura tiene una notable ventaja sobre las dos versiones anteriores, siendo dicha ventaja muy superior en los casos de bajos rangos de búsqueda.

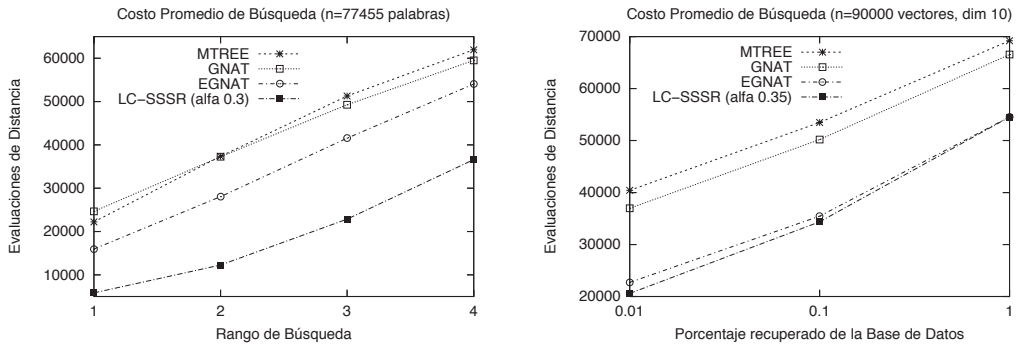
La figura 6 muestra los resultados para los mismos dos espacios de la nueva estructura versus tres estructuras conocidas de buen desempeño en espacios de alta dimensión. De los gráficos de búsqueda se puede ver claramente que el nuevo método aventaja a las estructuras *MTree*, *GNAT* y *EGNAT* muy notoriamente en el espacio de palabras. En el espacio de Gauss el *LC-SSSR* tiene un desempeño levemente mejor al *EGNAT*.

La figura 6 corresponde a un gráfico comparativo entre LC-SSSR y el EGNAT, la mejor de las estructuras comparadas. El experimento fue realizado sobre una nueva colección de datos, la que corresponde a un conjunto de 47.000 imágenes extraídas de archivos de la NASA y convertidas a vectores de 20 componentes. En esta colección, se puede notar que la estructura LC-SSSR aún mantiene una ventaja importante sobre la estructura EGNAT.



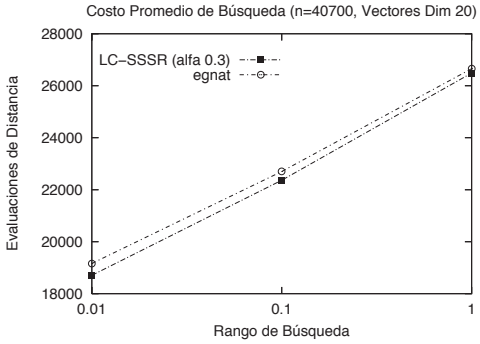
(a) Diccionario Español (b) Gauss Dimensión 10

Figura 5. Costos promedios de Búsqueda. Mejores versiones de Lista de Cluster versus la nueva estructura.



(a) Diccionario Español (b) Gauss Dimensión 10

Figura 6. Gráficos comparativos para la búsqueda (MTree v/s GNAT v/s EGNAT v/s LC-SSSR).



(a) Espacio de vectores de dimensión 20

Figura 7. Gráficos comparativos para la búsqueda (EGNAT v/s LC-SSSR).

5. Conclusiones

5.1. Aspectos Relevantes y Aportes

Una buena elección de pivotes y centros durante la construcción de estructuras métricas siempre será relevante para los procesos de búsqueda. Considerando que los mejores centros serán dependientes del espacio, es ideal contar con mecanismos que permitan recolectar, independiente de la forma del espacio, la mejores alternativas de centros.

En este sentido, los autores consideran que el método *SSS* permite, efectivamente, obtener un conjunto adecuado de centros, lo que queda demostrado claramente en el presente artículo.

Se considera que el principal aporte del presente trabajo es desarrollar la propuesta de una nueva estructura inicialmente denominada LC-SSSR, basada en la estructura *Lista de Clusters* con aplicación resursiva de la selección de centros espacialmente esparcidos (*SSS*). Esta estructura utiliza en forma natural el método *SSS* durante la construcción. Es basada en clustering y del tipo árbol y es competitiva en espacios de alta dimensión.

Los resultados experimentales demuestran lo anterior y proporcionan una visión de las enormes ventajas frente a otras estructuras que son prometedoras.

Es importante mencionar que la nueva estructura se va adaptando a la forma del espacio, dado el uso de SSS. Esto también es posible debido al recálculo del valor de M durante la construcción, lo que no tiene costos adicionales.

5.2. Trabajos Futuros

Los autores consideran que el desarrollo de esta estructura está aún en proceso de investigación y experimentación, por lo que se considera entre los trabajos futuros, la comparación con otras estructuras, la utilización de nuevas colecciones de objetos, la búsqueda de los mejores α's según cada subespacio o nivel interno de la estructura. Según los autores, una de las tareas pendientes es disminuir aún más los cálculos de distancia utilizando técnicas de descarte distintas al radio cobertor. Finalmente, determinar las capacidades dinámicas de la estructura y su comportamiento en memoria secundaria.

Referencias

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

2. Sergei Brin. Near neighbor search in large metric spaces. In *the 21st VLDB Conference*, pages 574–584. Morgan Kaufmann Publishers, 1995.
3. N. Brisaboa, A. Fariña, O. Pedreira, and N. Reyes. Selección espacial de pivotes dispersos para la búsqueda por similitud en espacios métricos. In *XII Congreso Argentino de Ciencias de la Computación*, Oct. 2006. San Luis, Argentina.
4. W. Burkhard and R. Keller. Some approaches to best-match file searching. *Communication of ACM*, 16(4):230–236, 1973.
5. B. Bustos, G. Navarro, and E. Chávez. Pivot selection techniques for proximity search in metric spaces. In *XXI Conference of the Chilean Computer Science Society*, pages 33–40. SCCC, IEEE Computer Science Press, 2001.
6. E. Chavéz, J. Marroquín, and R. Baeza-Yates. Spaghettis: An array based algorithm for similarity queries in metric spaces. In *6th International Symposium on String Processing and Information Retrieval (SPIRE'99)*, pages 38–46. IEEE CS Press, 1999.
7. E. Chavéz, J. Marroquín, and G. Navarro. Fixed queries array: A fast and economical data structure for proximity searching. *Multimedia Tools and Applications*, 14(2):113–135, 2001.
8. E. Chavéz and G. Navarro. An effective clustering algorithm to index high dimensional metric spaces. In *The 7th International Symposium on String Processing and Information Retrieval (SPIRE'2000)*, pages 75–86. IEEE CS Press, 2000.
9. Edgar Chávez, Gonzalo Navarro, Ricardo Baeza-Yates, and José L. Marroquín. Searching in metric spaces. In *ACM Computing Surveys*, pages 33(3):273–321, September 2001.
10. P. Ciaccia, M. Patella, and P. Zezula. M-tree : An efficient access method for similarity search in metric spaces. In *the 23st International Conference on VLDB*, pages 426–435, 1997.
11. 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.
12. Gonzalo Navarro. Searching in metric spaces by spatial approximation. *The Very Large Databases Journal (VLDBJ)*, 11(1):28–46, 2002.
13. Oscar Pedreira and Nieves R. Brisaboa. Spatial selection of sparse pivots for similarity search in metric spaces. In *SOFSEM 2007: 33rd Conference on Current Trends in Theory and Practice of Computer Science*, volume 4362 of *Lecture Notes in Computer Science*, pages 434–445, Harrachov, Czech Republic, January, 20-26 2007. Springer.
14. Caetano Traina, Agma Traina, Bernhard Seeger, and Christos Faloutsos. Slim-trees: High performance metric trees minimizing overlap between nodes. In *VII International Conference on Extending Database Technology*, pages 51–61, 2000.
15. J. Uhlmann. Satisfying general proximity/similarity queries with metric trees. In *Information Processing Letters*, pages 40:175–179, 1991.
16. Roberto Uribe-Paredes. Manipulación de estructuras métricas en memoria secundaria. Master's thesis, Facultad de Ciencias Físicas y Matemáticas, Universidad de Chile, Santiago, Chile, Abril 2005.
17. P. Yianilos. Data structures and algoritms for nearest neighbor search in general metric spaces. In *4th ACM-SIAM Symposium on Discrete Algorithms (SODA'93)*, pages 311–321, 1993.