# Técnicas de Indexación para Bases de Datos Avanzadas

**Norma Herrera, Darío Ruano, Paola Azar, Daniel Welch**
**Libertad Speranza, María de los Ángeles de la Torre**
Dpto de Informática, Universidad Nacional de San Luis, Argentina
{nherrera, dmruano, epazar, dwelch}@unsl.edu.ar
{libertadsperanza, angelesdlt}@gmail.com

**Anabella De Battista, Andrés Pascal**
Departamento Ingeniería en Sistemas de Información
FRCU, Universidad Tecnológica Nacional, Entre Ríos, Argentina
{anadebattista, andrespascal22}@gmail.com

## Resumen

Las bases de datos han incluido la capacidad de almacenar datos no estructurados tales como imágenes, sonido, texto, video, etc. La problemática de almacenamiento y búsqueda en estos tipos de base de datos difiere de las bases de datos clásicas, dado que no es posible organizarlos en registros y campos, y aun cuando pudiera hacerse, la búsqueda exacta carece de interés. Es en este contexto donde surgen nuevos modelos de bases de datos capaces de cubrir las necesidades de almacenamiento y búsqueda de estas aplicaciones. Nuestro interés se basa en el diseño de índices eficientes para estas nuevas bases de datos.

## Contexto

Este trabajo se desarrolla en el ámbito de la línea Técnicas de Indexación para Datos no Estructurados del Proyecto Tecnologías Avanzadas de Bases de Datos (22/F814), cuyo objetivo es realizar investigación básica sobre manejo y recuperación eficiente de información no tradicional.

## 1 Introducción

Gran parte de la información disponible en formato digital involucra el uso de datos no estructurados tales como imágenes, sonido, texto, video, etc. Debido a que no es posible organizar estos tipos de datos en registros y campos, las tecnologías tradicionales de bases de datos para almacenamiento y búsqueda de información no son adecuadas en este ámbito.

En este contexto surgieron nuevos modelos de bases de datos capaces de cubrir las necesidades de almacenamiento y búsqueda de estas aplicaciones. Nuestro interés se basa en el diseño de técnicas de indexación eficientes para estas nuevas bases de datos. Describimos a continuación los modelos de bases de datos en los que actualmente estamos trabajando.

**Bases de Datos de Texto.** Una base de datos de texto es un sistema que mantiene una colección grande de texto y que provee acceso rápido y seguro al mismo. Sin pérdida de generalidad, asumiremos que la base de datos de texto es un único texto $T$ que posiblemente se encuentra almacenado en varios archivos.

Una de las búsquedas más comunes en bases de datos de texto es la *búsqueda de un patrón*: el usuario ingresa un string $P$ (*patrón de búsqueda*) y el sistema retorna las ocurrencias

del patrón $P$ en el texto $T$. Para resolver eficientemente estas búsquedas se hace necesario construir un índice que permita acelerar el proceso de búsqueda.

Mientras que en bases de datos tradicionales los índices ocupan menos espacio que el conjunto de datos indexado, en las bases de datos de texto el índice ocupa más espacio que el texto, pudiendo necesitar de 4 a 20 veces el tamaño del mismo [7, 11]. Una alternativa para reducir el espacio ocupado por el índice es buscar una representación compacta del mismo, manteniendo las facilidades de navegación sobre la estructura [5, 6, 10, 15, 8, 9, 12, 13]. Pero en grandes colecciones de texto, el índice aún comprimido suele ser demasiado grande como para residir en memoria principal. Por esta razón, el estudio de índices comprimidos y en memoria secundaria para texto es un tema de interés

Un ***trie de sufijos***[7] es un full-text índice [14]. Básicamente es un *trie* construido sobre el conjunto de todos los sufijos del texto, en el cual cada hoja mantiene el índice del sufijo que esa hoja representa.

**Espacios Métricos.** El modelo de espacios métricos permite formalizar el concepto de búsqueda por similitud en bases de datos no tradicionales. [2].

Un espacio métrico está formado por un conjunto de objetos $\mathcal{X}$ y una función de distancia $d$ definida entre ellos que mide cuan diferentes son. La base de datos será un subconjunto finito $\mathcal{U} \subseteq \mathcal{X}$.

Una de las consultas más comunes en este modelo de bases de datos es la *búsqueda por rango*. En esta búsqueda dado un elemento $q \in \mathcal{X}$, al que llamaremos *query*, y un radio de tolerancia $r$, la búsqueda por rango consiste en recuperar los objetos de la base de datos cuya distancia a $q$ no sea mayor que $r$. Para evitar examinar exhaustivamente la base de datos, se preprocesa la misma por medio de un *algoritmo de indexación* con el objetivo de construir un *índice*, diseñado para ahorrar cálculos en el momento de la búsqueda. En [2] se presenta un desarrollo unificador de las soluciones existentes en la temática.

**Bases de Datos Métrico-Temporales.** Este modelo permite almacenar objetos no estructurados con tiempos de vigencia asociados y realizar consultas por similitud y por tiempo en forma simultánea. Formalmente un *Espacio Métrico-Temporal* es un par $(U, d)$, donde $U = O \times N \times N$, y la función $d$ es de la forma $d : O \times O \rightarrow R^+$. Cada elemento $u \in U$ es una triupla $(obj, t_i, t_f)$, donde $obj$ es un objeto (por ejemplo, una imagen, sonido, cadena, etc) y $[t_i, t_f]$ es el intervalo de vigencia de $obj$. La función de distancia $d$, que mide la similitud entre dos objetos, cumple con las propiedades de una métrica (positividad, simetría y desigualdad triangular).

Una consulta métrico-temporales se definen formalmente en símbolos como: $(q, r, t_{iq}, t_{fq})_d = \{o/(o, t_{io}, t_{fo}) \in X \wedge d(q, o) \leq r \wedge (t_{io} \leq t_{fq}) \wedge (t_{iq} \leq t_{fo})\}$ Esta consulta implica buscar todos los objetos $o$ de la base de datos que estén a una distancia a lo más $r$ de $q$, y cuyo tiempo asociado $t$ se solapa con el tiempo de la consulta.

Varios índices métrico-temporales se han propuesto en este ámbito, todos estos índices fueron desarrollados para ser eficientes en memoria principal. El *Historical-FHQT*(H-FHQT) [4] es un índice métrico-temporal que utiliza tanto la componente métrica como la temporal para resolver eficientemente búsquedas métrico-temporales. Este índice consiste en una lista de instantes válidos donde cada uno contiene un FHQT que indexa a todos los objetos vigentes en dicho instante.

# 2 Líneas de Investigación

## 2.1 Espacios Métricos

En este ámbito, avanzamos en el diseño de un prototipo de aplicación de índices métricos al comercio electrónico. El objetivo de este trabajo es agilizar las búsquedas por similitud sobre un conjunto de productos, con el fin de identificar eficiente y eficazmente aquellos productos con una descripción similar a la dada por un usuario.

En este caso, hemos usado como universo de datos un conjunto de productos disponibles en la plataforma Mercado Libre; la función que provee la medida de distancia, es la distancia de edición (o Levenshtein) aplicada al título principal de los productos. Este trabajo es la primera etapa de un proyecto mayor cuyo objetivo final es construir un sistema de recomendación basado en el modelo de espacios métricos.

En esta etapa, hemos usado el enfoque de indexación basado en pivotes para resolver la búsqueda de productos similares. Estudiamos el comportamiento de las búsquedas por rango utilizando las técnicas de selección de pivotes incremental y aleatoria. Los resultados obtenidos hasta el momento se pueden resumir en los siguientes puntos:

- A medida que aumentamos la cantidad de pivotes mejora el comportamiento del índice, pero no en la misma proporción que ocurre en los espacios de prueba generalmente utilizados en esta tématica.
- Se pudo observar que la selección incremental no siempre mejora a la selección aleatoria.
- El rango de búsqueda apropiado fue establecido experimentalmente dando como valor adecuado $r = 23$. Este valor difiere de los valores usados con los datasets habitualmente utilizados en esta temática, donde el rango de búsqueda mas apropiado es $r = 1, 2, 3, 4$.
- El universo de datos es bastante más singular, ya que se trata de títulos de productos reales, cuya redacción está a cargo del usuario que publica el producto para su venta y donde la única limitante es el tamaño de ese título (60 caracteres). Esta particularidad tiene como consecuencia un universo de datos variado y heterogéneo. Pudimos establecer experimentalmente que los espacios métricos obtenidos en este ámbito son de alta dimensionalidad.

Como trabajo futuro nos proponemos estudiar el comportamiento de índices basados en particiones compactas y diseñar además un índice en memoria secundaria, para luego implementar un sistema de recomendación basado en espacios métricos.

## 2.2 Bases de Datos de Texto

En este ámbito, usando como base el trie de sufijos, nos encontramos abocados al diseño de un índice en memoria secundaria, que además sea eficiente en espacio.

En [16] se propuso y evaluó una representación secuencial del trie de sufijos ( que denotaremos con $TS_s$ ) que es adecuada para un posterior proceso de paginación. En [17] se presentó una mejora en espacio del trie de sufijos (denotada con $TS_{dac}$ ), que consiste en el uso de códigos DAC (Directly Addressable Codes [1]) en algunas componentes de ($TS_s$ ). En [3] se propuso una mejora en tiempo del mismo índice (que denotaremos con $TS_{dacn}$ ) que implica agregar una componente más a $TS_s$ para mejorar las operaciones que se necesitan realizar durante la búsqueda de un patrón.

De la evaluación experimental de estas tres versiones se concluyó que:

- En tiempo de búsqueda $TS_{dacn}$ es la de mejor desempeño, logrando mejoras en tiempo de un $90\%$ aproximadamente.
- En cuanto al espacio ocupado, $TS_{dac}$ es la mejor opción, usando un $50\%$ menos de espacio que $TS_s$ . Sin embargo $TS_{dacn}$ , con muy poco espacio adicional ($0.01\%$) aproximadamente) logra mejorar substancialmente los tiempos de búsqueda.
- En cuanto a tiempos de construcción $TS_s$ es la de mejor desempeño, pero no es significativa la diferencia de tiempo de construcción con las otras dos versiones.

Por lo expuesto, $TS_{dac}$ es la versión seleccionada para continuar nuestro desarrollo. Nos encontramos trabajando en la implementación de un proceso de paginado de la misma.

## 2.3 Modelo Métrico-Temporal

En este ámbito nuestro objetivo es el diseño de índices en memoria secundaria. Específicamente estamos trabajando con el índice H-FHQT [4].

En este índice la búsqueda implica primero buscar por igualdad en la lista de instantes de tiempo y luego buscar por similitud sobre el FHQT asociado a cada instante seleccionado. Entonces, para paginar procedemos de la siguiente manera:

- Se pagina la lista de instantes válidos usando un Árbol $B^+$.
- Cada FHQT de cada instante de tiempo se pagina de manera separada.

Notar que en el H-FHQT la aridad de los nodos de los FHQT involucrados queda determinada por la cantidad de elementos que hay a cierta distancia de cada uno de los pivotes, por lo que la aridad no se puede modificar para adecuarla a una página de disco, como se hace en un Árbol $B^+$. Por esta misma razón el árbol tampoco se puede balancear.

Para paginar cada FHQT procedemos en forma bottom-up tratando de condensar en una única parte un nodo con uno o más de los subárboles que dependen de él. En este proceso de particionado las decisiones se toman en base a la *profundidad* de cada nodo involucrado, donde la profundidad indica la *cantidad de accesos a disco* que deberá realizar el proceso de búsqueda para llegar desde esa parte a una hoja del árbol. Comenzamos asignando cada hoja a una parte con profundidad 1 y luego, en forma bottom-up, procesamos cada uno de los nodos de este árbol r-ario.

Sea $x$ el nodo corriente a procesar, los hijos de $x$ se ordenan de mayor a menor según su profundidad; para aquellos hijos de igual profundidad se ordenan de menor a mayor según su tamaño. Este ordenamiento es solo a los efectos de la paginación, no implica modificar la topología del árbol. Para paginar el subárbol de raíz $x$ , se consideran los siguientes casos:

**Caso 1: $x$ y su primer hijo de mayor profundidad $d$ entran en una página de disco:**

• Se coloca en una misma parte $x$ y tantos hijos de $x$ como entren en una página, teniendo en cuenta en este proceso el orden ya establecido.

• Se cierran las partes de aquellos hijos que no conforman la nueva parte creada.

• Si todos los hijos de mayor profundidad $d$ se han agregado a la nueva parte creada, se establece que esta nueva parte tiene profundidad $d$. Si algún hijo de mayor profundidad $d$ es cerrado, la profundidad de la nueva parte se establece en $d + 1$.

**Caso 2: $x$ y su primer hijo de mayor profundidad $d$ no entran en una página de disco.**

• Se cierran todas las partes hijas y se crea una nueva parte para el nodo corriente.

• La profundidad de la nueva parte creada se establece en $d + 1$, donde $d$ es el máximo de las profundidades de los hijos.

En este proceso, cerrar una parte significa grabarla en disco y reemplazarla en el árbol original por una hoja que contiene el número de página donde esa parte se grabó. Actualmente estamos trabajando en la implementación de este proceso de paginado.

# 3 Resultados Esperados

Se espera obtener índices eficientes, tanto en espacio como en tiempo, para el procesamiento de consultas en bases de datos textuales, en bases de datos métricas y en bases de datos métrico temporales. Los mismos serán evaluados tanto analíticamente como empíricamente.

# 4 Recursos Humanos

El trabajo desarrollado en esta línea forma parte del desarrollo de dos Trabajos Finales de Licenciatura, dos Tesis de Maestría y una Tesis de Doctorado, todas ellas en el ámbito de la Universidad Nacional de San Luis.

# Bibliografia


[1] N. Brisaboa, S. Ladra, and G. Navarro. Directly addressable variable-length codes. In *Proc. 16th International Symposium on String Processing and Information Retrieval (SPIRE)*, LNCS 5721, pages 122–130. Springer, 2009.

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

[3] J. Cornejo, D. Ruano, and N. Herrera. Una mejora en tiempo del trie de sufijos. In *Congreso Argentino de Ciencias de la Computación*, Rio Cuarto, Córdoba, Argentina, 2019.

[4] A. De Battista, A. Pascal, G. Gutierrez, and N. Herrera. Un nuevo índice métrico-temporal: el historical-fhqt. In *Actas del XIII Congreso Argentino de Ciencias de la Computación*, Corrientes, Agentina, 2007.

[5] P. Ferragina and G. Manzini. Indexing compressed text. *J. ACM*, 52(4):552–581, 2005.

[6] T. Gagie and G. Navarro. *Compressed Indexes for Repetitive Textual Datasets*. Springer, 2019.

[7] G. H. Gonnet, R. Baeza-Yates, and T. Snider. *New indices for text: PAT trees and PAT arrays*. Prentice Hall, New Jersey, 1992.

[8] R. González and G. Navarro. Compressed text indexes with fast locate. In *Proc. 18th Annual Symposium on Combinatorial Pattern Matching (CPM)*, LNCS 4580, pages 216–227, 2007.

[9] R. González, G. Navarro, and H. Ferrada. Locally compressed suffix arrays. *ACM Journal of Experimental Algorithmics*, 19(1):article 1, 2014.

[10] Roberto GROSSI and Jeffrey Scott VITTER. Compressed suffix arrays and suffix trees with applications to text indexing and string matching. *SIAM journal on computing*, 35(2):378–407, 2006.

[11] U. Manber and G. Myers. Suffix arrays: A new method for on-line string searches. *SIAM Journal of Computing*, 22(5):935–948, 1993.

[12] E. Moura, G. Navarro, N. Ziviani, and R. Baeza-Yates. Fast and flexible word searching on compressed text. *ACM TrFansactions on Information Systems (TOIS)*, 18(2):113–139, 2000.

[13] J. I. Munro, G. Navarro, and Y. Nekrich. Fast compressed self-indexes with deterministic linear-time construction. *Algorithmica*, 82(2):316–337, 2020.

[14] G. Navarro and V. Mäkinen. Compressed full-text indexes. *ACM Computing Surveys*, 39(1):article 2, 2007.

[15] G. Navarro and K. Sadakane. *Compressed Tree Representations*. Springer, 2nd edition, 2015.

[16] D. Ruano and N. Herrera. Representación secuencial de un trie de sufijos. In *XX Congreso Argentino de Ciencias de la Computación*, Buenos Aires, Argentina, 2014.

[17] D. Ruano and N. Herrera. Indexando bases de datos textuales: Una representación compacta del trie de sufijos. In *Congreso Nacional de Ingeniería Informática / Sistemas de Información*, Buenos Aires, Argentina, 2015.