Definición de un Recuperador de Imágenes basado en Contenidos sobre Espacios Métricos

Eduardo Peña Jaramillo[1] and Roberto Uribe Paredes[1,2]
[1] Depto. de Ingeniería en Computación
Universidad de Magallanes, Chile
[2] Grupo de Base de Datos – UART
Universidad Nacional de la Patagonia Austral, Rio Turbio, Argentina
E-mail: {edopena, ruribe}@ona.fi.umag.cl

Resumen

La recuperación de imágenes desde una base de datos ha una tarea estudiada desde hace varios años . La producción de imágenes crece más rápido que las metodologías que administran y procesan visualmente la información, dejando como gran reto la recuperación, almacenamiento y representación. Actualmente existen muy pocas máquinas recuperadoras de imágenes con enfoques basados en contenidos. Este enfoque ha ganado popularidad en la comunidad científica debido a que sus resultados han sido satisfactorios y de mayor efectividad que los sistemas basados en enfoques tradicionales.

Una de las líneas de investigación en los recuperadores de imágenes está dirigida hacia el manejo eficiente de los espacios de almacenamiento, construcción de índices y manejo de estructuras de búsquedas.

El presente artículo muestra la definición de un recuperador de imágenes basado en contenidos, con un enfoque sobre imágenes multidimensionales complejas de distinta clase bajo un índice común.

La Recuperador de Imágenes Basado en Contenidos propuesto fue implementado mediante estructuras métricas SAT y EGNAT, bajo modelos paralelos BSP.

Keywords: Recuperadores de imágenes basados en contenidos, estructuras de datos, espacios métricos, consultas por similaridad, paralelismo, BSP.

1. Introducción

1.1. Antecedentes

Las bases de datos de imágenes pueden ser enormes en tamaño, conteniendo centenares, miles o incluso millones de objetos. En los sistemas recuperadores de imágenes, la búsqueda es uno de los problemas que ha concentrado gran parte de los esfuerzos y se realiza comúnmente como: dada una consulta sobre un conjunto de datos, se recupera el valor que es igual a la consulta o query. Ésta es conocida como búsqueda exacta y se aplica en bases de datos tradicionales sobre datos estructurados.

Sobre estas bases de datos también se pueden realizar búsquedas más sofisticadas como consultas por rango sobre claves numéricas o prefijos sobre claves alfabéticas, sin embargo, éstas también se basan en el concepto de que dos claves son o no iguales o que existe un orden sobre las claves. A pesar de que en la actualidad las bases de datos permiten almacenar tipos de datos más complejos, como por ejemplo, imágenes, audio o video, las búsquedas se realizan de igual manera sobre los atributos de tipo alfanumérico.

Con la evolución de las tecnologías de la información y comunicación, han surgido depósitos no estructurados de datos. No sólo se hace necesario almacenar datos, sino también realizar búsquedas sobre ellos. Tal estructuración es muy dificultosa, (tanto manual como computacionalmente), y restringe de antemano los tipos de consultas que luego se pueden efectuar. Realizar búsquedas exactas sobre estos datos sería poco útil. Por ejemplo, si se consultase por un elemento sobre una base de datos de imágenes, la consulta sólo podría encontrar su copia digital exacta en la base de datos. El verdadero interés reside, por ejemplo, en consultar sobre una base de datos de fotografías una imagen que contiene un rostro, donde no necesariamente existe una copia exacta de la misma fotografía; identificación de individuos a

través de dispositivos biométricos, donde el dato consulta (voz, retina, huella digital, rostro, etc.) podría verse afectado por factores externos; etc.

Para lo anteriormente mencionado se requiere de algoritmos de búsqueda y modelos más generales. Un concepto clásico para estos algoritmos es el de "búsqueda por similitud" o "búsqueda en proximidad", es decir buscar elementos de la base de datos que sean similares o próximos a un elemento de consulta dado, en este sentido es posible que la consulta no sea parte de la base de datos.

La necesidad de procesar grandes cantidades de datos obliga a aumentar la capacidad de procesamiento. Para ello es necesaria la paralelización de los algoritmos y la distribución de las bases de datos.

En este trabajo se propone un recuperador de imágenes complejas de distintas clases bajos un índice común utilizando estructuras métricas SAT y EGNAT implementando algunas estrategias de distribución de carga en procesadores paralelos para hacer más eficiente la búsqueda de elementos en la base de datos. El método de computación paralela utilizado es el denominado modelo BSP [10] qué proporciona independencia de la arquitectura de la computadora.

1.2. Marco Teórico

Recuperación de Imágenes Basadas en Contenido: Un sistema ***CBIR*** (Content Based Image Retrieval), sinónimo de (feature-based retrieval), nace del estudio de la percepción humana y recupera información por su contenido. Tienen como base una serie de tecnologías que permiten al usuario almacenar, indexar, clasificar y posteriormente recuperar información de imágenes por su contenido. Usa un análisis y procesamiento digital para generar descriptores a partir de los datos.

Los pasos que se deben tener en cuenta para su construcción y funcionamiento son: extraer características de imágenes, almacenar índices, construir solicitudes de búsqueda y retornar resultados del proceso de búsqueda. Estos pasos se efectúan independientemente de la arquitectura del sistema, ya sea centralizada, (los diferentes pasos de funcionamiento los ejecuta un solo sistema modular); distribuida, (cada operación la realiza un elemento distribuido); o Cliente/Servidor.

Espacios métricos: La similaridad en muchos casos es modelada a través de un espacio métrico y la búsqueda de objetos más similares bajo una función conveniente de similaridad, a través de una búsqueda por rango o vecinos más cercanos.

Definición 1 (Espacios métricos): Un espacio métrico es un conjunto X con una fución de distancia d: $X^2 \longrightarrow R$, tal que para todo $x,y,z \in X$

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 $d(x,y) \leq r$.

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.

Existen dos métodos para la construcción de estructuras métricas, los basados en Clustering y los basados en Pivotes. El primero divide el área en particiones Voronoi, donde existe un centro por cada área y los demás objetos se almacenan en el centro más cercano.

En el caso de los algoritmos basados en Pivotes, un pivote es un objeto preseleccionado y que no necesariamente pertenece a la base de datos. Su objetivo es filtrar objetos en una consulta a través de la utilización de la desigualdad triangular, sin medir realmente la distancia entre el objeto consulta y los objetos descartados.

Existen distintas estructuras para buscar en espacios métricos, las cuales pueden ocupar funciones discretas o continuas de distancia. Algunos son BKTree [3], MetricTree [8], GNAT [2], VpTree [11], FQTree [1], MTree [4], SAT [5,7], EGNAT [11].

Sin embargo son pocas las estructuras que tienen buen desempeño en espacios de alta dimensión, que es el caso de estudio. Además, son mucho menos las estructuras que permiten dinamismo (eliminación y reinserción de datos).

SAT (Árbol de Aproximación Espacial): Es una estructura propuesta en [12], que se basa en un concepto diferente respecto de la mayoría de las estructuras métricas. La idea general, más que dividir el espacio de búsqueda, es aproximarse espacialmente a la consulta. Para ello se utilizan las búsquedas del vecino más cercano como una forma natural de aproximarse a la consulta. En este modelo, dado un punto $q \epsilon X$ (conjunto de objetos válidos del espacio métrico) y estando posicionado en algún elemento $a \epsilon Y$ (base de datos) con $Y \subseteq X$, el objetivo es moverse a otro elemento de Y que esté más cerca espacialmente de q que a .

EGNAT (Evolutionary gnat): El gnat evolutivo o egnat presentado en [11] es una estructura de datos completamente dinámica, competitiva para búsquedas por similaridad en espacios métricos de alta dimensión y con buen desempeño en memoria secundaria. El egnat pertenece al grupo de algoritmos basados en particiones compactas y es una optimización en memoria secundaria para el gnat en términos de espacio, accesos a disco y evaluaciones de distancia. El egnat es un árbol que posee dos tipos de nodos, un nodo bucket y un nodo gnat. Los nodos nacen como bolsas o buckets y cuya única información es sólo la distancia al padre, al estilo de las estructuras basadas en pivotes (un solo pivote). Este mecanismo es el que permite disminuir el espacio requerido en disco para almacenar la estructura y logra mantener un buen desempeño en términos de evaluaciones de distancia. Si un nodo se completa, éste evoluciona de un nodo bolsa a un nodo gnat, reinsertando los objetos de la bolsa al nuevo nodo gnat.

Modelo de programación paralela BSP: El modelo BSP de computación paralela fue propuesto en 1990 con el objetivo de permitir el desarrollo de software sea portable y tenga desempeño eficiente y escalable [10]. BSP propone alcanzar este objetivo mediante la estructuración de la computación en una secuencia de pasos llamados supersteps y el empleo de técnicas aleatorias para el ruteo de mensajes entre procesadores. El computador paralelo, independiente de su arquitectura, es visto como un conjunto de pares procesadores-memoria, los cuales son conectados mediante una red de comunicación cuya topología es transparente al programador. Los supersteps son delimitados mediante la sincronización de procesadores. Los procesadores proceden al siguiente superstep una vez que todos ellos han alcanzado el final del superstep, los cuales son agrupados en bloques para optimizar la eficiencia de la comunicación. Durante un superstep, los procesadores trabajan asincrónicamente con datos almacenados en sus memorias locales. Cualquier mensaje enviado por un procesador está disponible para procesamiento en el procesador destino sólo al comienzo del siguiente superstep. Dada la estructura particular del modelo de computación, el costo de los programas BSP puede ser obtenido utilizando técnicas similares a las empleadas en el análisis de algoritmos secuenciales. En BSP, el costo de cada superstep esta dado por la suma de el costo en computación (el máximo entre los procesadores), el costo de sincronización entre procesadores, y el costo de comunicación entre procesadores (el máximo enviado/recibido entre procesadores). El costo total del programa BSP es la suma del costo de cada superstep.

Para el presente artículo se seleccionó la estructura EGNAT, la cual ha desarrollado buen desempeño en espacios de alta dimensión. Esta basada en clustring y son del tipo árbol. Sobre dichas estructura se implementará el recuperador de imágenes basadas en contenido y se mostrará algunas estrategias de distribución de carga sobre procesadores paralelos permitiendo una búsqueda eficiente dada una consulta por rango.

2. CBIR Propuesto

Recuperar información desde una imagen basada en contenido corresponde a una metodología de recuperación con respecto al dominio de aplicación del proceso de recuperación en si. Usa un análisis y procesamiento digital para generar descriptores a partir de los datos. Los méritos principales de sistemas basados en el contenido son: soporta el procesamiento de consultas visuales, la consulta es intuitiva y amistosa al usuario, la generación de los descriptores es automática, siendo objetiva y consistente.

2.1. Extracción de Características

El sistema recuperador implementado en este trabajo, propuesto en [6], mide las propiedades de todas las imágenes que comprenden cada una de las clases de nuestra base de datos. A partir de estas características medidas se puede decidir si los objetos imagen corresponden a una u otra clase y así poder discriminar clasificar y recuperar desde el subconjunto adecuado. Se trabaja con este tipo de características para lograr una dimensión alta de los vectores representativos de cada imagen. Si las características extraídas representan un valor global y la imagen es compleja, (más de cuatro regiones localizadas), estas características no servirían para describir la imagen debido a que se necesita información adicional relacionada con los tipos de regiones como por ejemplo su descripción geométrica y su importancia dentro de la imagen en estudio. Una forma de disminuir esta imprecisión frente a este tipo de descriptores globales es aplicar la técnica *"layout"* [6], que consiste en subdividir la matriz imagen en la mayor cantidad posible de subcuadros y extraer de cada uno de éstos la misma información que se pretendía extraer en un comienzo en forma global.

Como las imágenes utilizadas para el presente trabajo, son de dimensión 256x256 píxeles, al aplicar la técnica "*layout*", la imagen original queda subdividida en 256 cuadros de 16x16 píxeles cada uno. Realizar esta técnica simula el tener 256 regiones segmentadas de una imagen original con el mismo peso y a las cuales se les extraen sus características. Las características extraídas de cada una de las imágenes de la base de datos, del presente trabajo, corresponden a características globales de intensidad en 9 espacios de color, histogramas en espacio RGB, histograma en niveles de gris, 20 niveles de energía bajo técnicas Wavelet Standard, textura de Segundo momento angular (E), Contraste (I), Entropía (H**)** y Homogeneidad **(**Z**),** obtenidas de la matriz de co-ocurrencia en niveles de grises con distancias entre píxeles de d=1, d=2, y d=3 y con orientaciones de 0°, 45°, 90° y 135°. La cantidad de elementos por cada característica junto a su descripción y posición dentro del vector formado esta presentado en la Tabla I.

La unión de todas estas características extraídas nos da como resultado un gran vector de 27 características con dimensión de 14.431 elementos. Este vector será el que represente a la imagen dentro de la base de datos. Cabe destacar que cada característica fue normalizada antes de incorporarla al vector. Esta normalización consiste en transformar linealmente cada característica de tal forma que se tenga un valor medio igual a cero y una varianza igual a uno.

Una vez extraídas las características de las imágenes se procedió a preseleccionar cuál de éstas cumplen con estar por sobre el 20% de la función discriminante maximizada, (Fisher), de tal manera de asegurar una buena separabilidad entre las clases, esto se realiza para cada una de los conjuntos de imágenes.

Luego de la preselección de características se procedió a seleccionar mediante el método Sequential Forward Selection (SFS) **(Jain et al, 2000)** las características más relevantes de las preseleccionadas. Para esto se trabajó con el discriminante Fisher para evaluar el desempeño de la clasificación.

Esta clasificación de imágenes sobre un mismo espacio es la que permite finalmente poder agruparlas a todas bajo una misma estructura y no tener una para cada clase. Si apareciese una imagen cuyo patrón difiere de los que lideran la estructura de índice, éste pasa a ser un nuevo patrón o clase de búsqueda e indexación.

2.2. Clasificación y búsqueda de patrones

Cuando ya se tiene las características que describen a cada una de las 12 clases de imágenes propuestas para este trabajo se procede a definir los patrones que caracterizan a cada clase.

Para diferenciar los falsos positivos en la recuperación de la base de datos de imágenes se realizó una clasificación para cada grupo de imágenes. La clasificación analiza los rasgos de imagen y las clasifica en una de las clases siguientes: imagen falsa que no corresponde al contexto del grupo seleccionado e imagen positiva perteneciente al grupo de búsqueda. La dimensión de los vectores extraídos de cada uno de las 12 clases de imágenes varían de una clase a otra dependiendo de la cantidad de características que la definan. El sistema de recuperación propuesto debe contener a todas las imágenes en su índice el cual apunta a la base de datos. El problema de comparar imágenes cuyos vectores de características no concuerdan en dimensión y posición, se solucionó creando un patrón por cada clase de imagen. Esto permite poder almacenar bajo un mismo índice a todas las clases de imágenes, (ver tablas II y III).

Tabla I: Características Extraídas a cada imagen.

N° característica	**Característica**	**N° elementos**	**N° elemento vector**	**Descripción**
1	BlueLay	256	1-256	Promedio de color blue layout
2	CMYk	3	257-259	Promedio espacio colores CMYk global
3	CMY	3	260-262	Promedio espacio colores CMY global
4	CY	3	263-265	Promedio espacio colores CY global
5	GrayLay	256	266-521	Promedio de color gray layout
6	Gray	1	522	Promedio de color gray global
7	GreenLay	256	523-778	Promedio de color green layout
8	HistBlue	256	779-1034	Histograma de color blue global
9	HistGray	256	1035-1290	Histograma de color gray global
10	HistGreen	256	1291-1546	Histograma de color green global
11	HistRed	256	1547-1802	Histograma de color red global
12	HSI	3	1803-1805	Promedio espacio colores HSI global
13	HSV	3	1806-1808	Promedio espacio colores HSV global
14	RedLay	256	1809-2064	Promedio de color red layout
15	RGB	3	2065-2067	Promedio espacio colores RGB global
16	TextureE_gray	12	2068-2079	Promedio y rango de característica homogeneidad de textura global para d=1,d=2,d=3
17	TextureH_gray	12	2080-2091	Promedio y rango de característica entropia de textura global para d=1,d=2,d=3
18	TextureI_gray	12	2092-2103	Promedio y rango de característica contraste de textura global para d=1,d=2,d=3
19	TextureZ_gray	12	2104-2115	Promedio y rango de característica 2° momento angular de textura global para d=1,d=2,d=3
20	Text_LayoutEgray	3072	2116-5187	Promedio y rango de característica E de textura layout para d=1,d=2,d=3
21	Text_LayoutHgray	3072	5188-8258	Promedio y rango de característica H de textura layout para d=1,d=2,d=3
22	Text_LayoutIgray	3072	8259-11330	Promedio y rango de característica I de textura layout para d=1,d=2,d=3
23	Text_LayoutZgray	3072	11331-14402	Promedio y rango de característica Z de textura layout para d=1,d=2,d=3
24	Wavelets_Energy_gray	20	14403-14422	20 primeros niveles de energía wavelets
25	YCrCb	3	14423-14425	Promedio espacio colores YCrCb global
26	YIQ	3	14426-14428	Promedio espacio colores YIQ global
27	YUV	3	14429-14431	Promedio espacio colores YUV global

El vector que describe en forma general a cada una de las imágenes de la base de datos, incluyendo todas sus características, se expresa de la siguiente manera:

Vim=[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27]

Cada uno de los elementos descritos con los número 1 al 27 en el vector Vim, corresponden a las características extraídas de cada imagen.

Las formas de indexación bajo espacios métricos, consiste en medir distancias entre todos los elementos de la base de datos y cumplir con ciertas reglas para poder armar un árbol eficiente de búsqueda. Para poder realizar estas medidas, con las métricas evaluadas en este trabajo, es necesario que los elementos cumplan con tener las mismas dimensiones y sus características deben estar en la misma posición dentro del vector. La solución al problema de indexar vectores de distintas clases sería mantener un patrón de posiciones de características en cada vector. Esto significa que cada característica tiene una posición específica dentro del vector y en caso de no existencia de esta característica se reemplaza por

elementos de la misma dimensión cuyo valor sea nulo. Los vectores que representan a cada una de las clases propuestas bajo el enfoque de una indexación en espacios métricos quedarían según la Tabla II.

Tabla II: Vectores que representan a cada clase dentro del índice en espacio métrico. Descripción de la clase y dimensión de cada característica del vector se puede apreciar en la tabla I.

CLASE	VECTOR
1	[**1** 0 0 0 0 0 0 0 0 **9** 0 0 **12** 0 0 **15** 0 0 0 **19** 0 0 0 0 0 0 **26** **0**]
2	[0 0 0 0 **5** 0 0 0 0 0 0 **11** 0 0 0 0 0 0 0 0 0 0 0 **22** 0 **24** 0 0 0]
3	[0 0 0 0 0 0 0 0 0 0 0 0 **13** 0 0 0 0 0 0 0 0 **21** 0 0 0 **25** 0 0]
4	[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 **18** 0 0 0 0 0 0 0 **26** 0]
5	[0 **2** 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 **19** 0 0 0 0 **24** 0 0 0]
6	[0 0 0 0 0 0 0 0 0 **9** 0 0 0 0 0 0 0 **17** **18** 0 0 0 0 0 0 0 0 0]
7	[0 0 0 0 0 0 0 0 0 **9** 0 0 0 0 0 0 0 0 **18** 0 **20** 0 0 0 0 0 0 0]
8	[0 0 0 0 0 0 0 0 0 0 0 0 **12** 0 0 0 0 0 0 **18** 0 **20** 0 0 0 0 0 0 0]
9	[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 **18** 0 0 0 **22** 0 0 0 0 0]
10	[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 **18** **19** 0 0 0 **23** 0 0 0 0]
11	[0 0 0 0 0 0 0 0 0 **9** **10** 0 0 0 0 **15** 0 0 0 0 0 0 0 0 0 0 0 0]
12	[0 0 0 0 0 0 0 0 0 **9** **10** 0 0 0 0 0 0 0 0 0 0 0 **22** 0 0 0 0 0]

2.3 Máscaras

Para comparar una imagen con la estructura de índices que abarca la base de datos es necesario un patrón cuya finalidad sea filtrar las características que serían necesarias para hallar la similitud entre dos objetos al momento de compararlos. La obtención de un patrón consiste en tomar todos los vectores de una clase y hallar la media, (centro de gravedad), de este grupo. La clasificación utilizada para el reconocimiento de patrones en este trabajo es el basado en el concepto de similitud. Sea ***Xi*** un vector de características que representa el centro de gravedad de la clase de imagen **(*i*),** con **i=1,…,12**. Dada la imagen consulta cuyo vector es ***Yo*** se busca la menor distancia a ***Xi*** y el vector obtenido como menor distancia indicará la clase a la cual se asemeja y por ende el tipo de máscara que se le debe aplicar según tabla III, al vector ***Yo*** y así buscar dentro del índice que agrupa a todas las clases de la base de datos de imágenes.

Tabla III: Máscaras que representan a cada clase para aplicar a imágenes consultas antes de entrar en el índice de la estructura métrica espacial. Descripción de la clase y dimensión de cada característica del vector se puede apreciar en la tabla I

CLASE	MASCARA
1	[**1** 0 0 0 0 0 0 0 0 **1** 0 0 **1** 0 0 **1** 0 0 0 **1** 0 0 0 0 0 0 0 **1** 0]
2	[0 0 0 0 **1** 0 0 0 0 0 0 **1** 0 0 0 0 0 0 0 0 0 0 **1** 0 **1** 0 0 0]
3	[0 0 0 0 0 0 0 0 0 0 0 0 0 **1** 0 0 0 0 0 0 0 0 **1** 0 0 0 **1** 0 0]
4	[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 **1** 0 0 0 0 0 0 0 0 **1** 0]
5	[0 **1** 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 **1** 0 0 0 0 **1** 0 0 0]
6	[0 0 0 0 0 0 0 0 0 **1** 0 0 0 0 0 0 0 **1** **1** 0 0 0 0 0 0 0 0 0 0]
7	[0 0 0 0 0 0 0 0 0 **1** 0 0 0 0 0 0 0 0 **1** 0 **1** 0 0 0 0 0 0 0 0]
8	[0 0 0 0 0 0 0 0 0 0 0 0 **1** 0 0 0 0 0 **1** 0 **1** 0 0 0 0 0 0 0 0]
9	[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 **1** 0 0 0 **1** 0 0 0 0 0]
10	[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 **1** **1** 0 0 0 **1** 0 0 0 0]
11	[0 0 0 0 0 0 0 0 0 **1** **1** 0 0 0 0 **1** 0 0 0 0 0 0 0 0 0 0 0 0]
12	[0 0 0 0 0 0 0 0 0 **1** **1** 0 0 0 0 0 0 0 0 0 0 0 **1** 0 0 0 0 0]

2. Estrategias de Paralelización de Estructuras

2.1. Estructuras Métricas

Las estructuras seleccionadas para implementar el recuperador de imágenes fueron árboles: SAT [5,7] y EGNAT [11]. Éstas fueron seleccionadas por su buen desempeño en espacios de alta dimensión, la posibilidad de dinamismo y por la documentación existente. Para discriminar áreas durante la búsqueda, el SAT usa el criterio de radio cobertor y el EGANT una mezcla entre radio cobertor y el criterio de hiperplanos. El EGNAT almacena tablas de rangos que le permite determinar si las áreas de búsqueda se intersectan con las tablas de rango almacenadas, reduciendo así los subárboles donde buscar.

2.2. Estrategias

Pese a la efectividad y robustez de la estructuras evaluadas [6,11], existe problemas de búsquedas para conjuntos de imágenes sobre 100.000 elementos, para lo cual se proponen estrategias de paralelización basadas sobre modelos BSP (Bulk Synchronous Parallel). En esta sección se describen estrategias de distribución de la estructura SAT y EGNAT en múltiples procesadores y la paralelización de su algoritmo de búsqueda. La paralelización se realizó utilizando modelo BSP, (Bulk Synchronous Parallel), para ver y evaluar el comportamiento y rendimiento de los algoritmos bajo las estrategias de distribución. Se midió el cálculo observado por el número de evaluaciones de distancias entre objetos. El número de evaluaciones de distancia ***(ED)***, es la suma de todos los $\boldsymbol{maxED_i}$ (número máximo de ED de un determinado procesador en el superstep i).

$$ED = \sum_{i=1}^{superstep} \max ED_i$$

La principal métrica que se utiliza para cómputos efectuados por cada procesador es el balance de carga. Ésta es medida por el radio promedio hacia el máximo número de cómputos observados en cada procesador. Esto es llamado eficiencia ***Ef*** y un valor 1 indica el óptimo.

$$E_f = \frac{\left\lfloor \frac{\sum_{i=1}^{N°Proc} \max ED_i}{\text{N° Procesadores}} \right\rfloor}{\text{Máximo valor de ED entre los procesadores}}$$

Para todas las estrategias que se proponen, existe un broker que distribuye las consultas de forma circular entre todos los procesadores.

Por cada superstep cada una de los procesadores reciben $\boldsymbol{Q}$ consultas, (desde el broker), y realiza el proceso de búsqueda con esa información. Posteriormente se procede a distribuir las $\boldsymbol{Q}$ consultas, (procesadas anteriormente), a los demás procesadores y a la vez se envían resultados de las consultas a los procesadores que corresponda.

Estrategia 1 (Estructura multiplexada en cada procesador)

La siguiente estrategia que se propone consiste en distribuir en forma multiplexada entre los procesadores disponibles los subárboles enlazados a los nodos hijos de la estructura. No se requiere duplicación de datos de la estructura de árbol, salvo la raíz y sus hijos.

Para recuperar el 0.01% y 0.1% de los datos se puede observar, los cálculos de evaluaciones de distancias hechas por cada procesador para recuperar estos porcentajes.

Esta estrategia resultó tener una mejor eficiencia en la recuperación llegando a un promedio de Ef=0.61 para estructura SAT y Ef=0.68 para estructura EGNAT.

A pesar de haber aumentado la eficiencia en la recuperación paralela la estrategia presentada no es buena debido a que existen desbalances significativos de carga entre los procesadores, (estructura SAT en mayor proporción que EGNAT).

Estrategia 2 (Estructura multiplexada balanceada en cada procesador)

Esta estrategia consiste en crear varios árboles a partir de la estructura original, (un árbol por procesador), con la característica que cada árbol generado sólo tendrá algunos subárboles de la estructura original. Al momento de asignar un subárbol se consulta sobre quién es el procesador con menos nodos para que a éste se le asigne dicho subárbol. Los nodos raiz e hijos de la raíz están duplicados en cada procesador. Esta estrategia mejora la media Ef=0.74 para estructura SAT y Ef=0.8 para estructura EGNAT para las consultas exigidas en el caso de de la Estrategia 2. La mejora es significante con un pequeño aumento en el costo de la comunicación, (menos del 1% con respecto al número de evaluaciones de distancias). La comunicación consiste en el envío de consultas entre procesadores durante el proceso de búsqueda, esto se hace necesario debido a que los subárboles se localizan en diferentes procesadores sin duplicaciones.

Estrategia 3 (Estructura multiplexada y balanceada recursivamente en cada procesador)

Consiste en controlar la distribución de subárboles o nodos, según sea el caso, en los procesadores que se tengan disponibles. Esta estrategia toma la estructura original y cuenta la cantidad total de nodos que la conforman. Se obtiene un promedio de los nodos que deberían ir distribuidos en total por cada procesador y ese valor será el patrón de control para ir incorporando nodos a los procesadores. Para incorporar un subárbol completo en un procesador se debe tener en cuenta que la cantidad de nodos que posee el subárbol debe ser menor igual al promedio aceptado por el procesador. En caso contrario se copia el nodo que encabeza ese subárbol en cada uno de los procesadores y se procede a avanzar dentro del subárbol. Una vez avanzado al siguiente nodo del subárbol se consulta nuevamente por la cantidad de nodos en el. Esta recursividad se repite tantas veces sea necesario hasta que el promedio de nodos sea menor igual al aceptado por los procesadores y que se haya recorrido todos los subárboles del nodo por donde se comenzó. A medida que se avanza en los nodos y subárboles se van copiando en el procesador que tenga menos nodos de carga, (distribución balanceada). Los nodos raíz e hijos de la raíz están duplicados en cada procesador. Esta estrategia mejora la media Ef=0.82 para estructura SAT y Ef=0.9 para estructura EGNAT para las mismas consultas exigidas en los casos anteriores. La mejora es significante pero aumenta nuevamente el costo de la comunicación pero a pesar de ello sigue siendo menos del 1% con respecto al número de evaluaciones de distancias. Los costos de comunicación aumentan en los casos en que existen más nodos que ramas en un subárbol, (caso estructura SAT), y/o más elementos en el split que el promedio aceptado por cada procesador, (caso estructura EGNAT). La figura 1 muestra los resultados de estas tres estrategias.

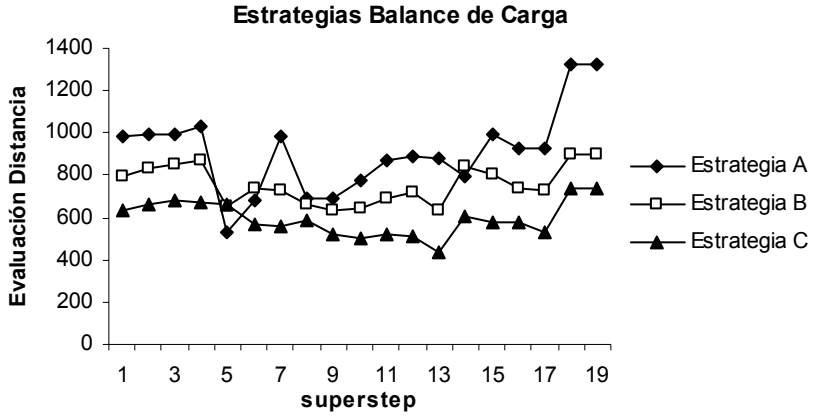

Figura 1: Comparación de estrategias de balance de carga (A) sin balance multiplexado, (B) con balance multiplexado y (C) multiplexado con balance recursivo. Consultas con distribución circular con radio de recuperación 0.1% sobre 4 procesadores.

Se llevó a cabo la evaluación de la estructura formada con la estrategia de balance de carga multiplexado y recursivo con pruebas en 2 y hasta 10 procesadores para evaluar mediciones de distancias, tiempos, comunicación y eficiencia de la paralelización.

Las figuras 3 y 4 nos muestra los resultados en términos de eficiencia para las tres estrategias, comparando los resultados para las dos estructuras. Los experimentos graficados en estas figuras corresponden a un set de 1000 imágenes con dimensión 1431, la ejecución fue realizada sobre 2 y 4 procesadores.

Como se puede observar la estructura EGNAT entrega mejores resultados de eficiencia que la estructura SAT en espacios de alta dimensión. Este comportamiento ocurrió en las tres estrategias. Para pruebas sobre 1000 imágenes con la misma dimensión se descartó la estructura SAT debido a que el tiempo de construcción es exponencial a medida que aumenta la cantidad de elementos de la base de datos. Otro problema de la estructura SAT, es que al no estar optimizada para memoria secundaria requería de muchos recursos al trabajar directamente en memoria principal. EGNAT, en cambio, no da problemas al trabajar con memoria secundaria.

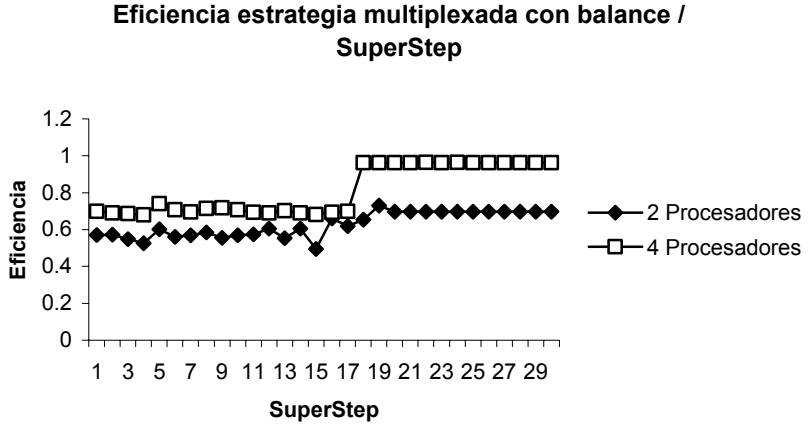

Figura 2: Eficiencia de estrategia Árbol multiplexado con balance de carga y distribución circular de consultas para SAT y EGNAT.

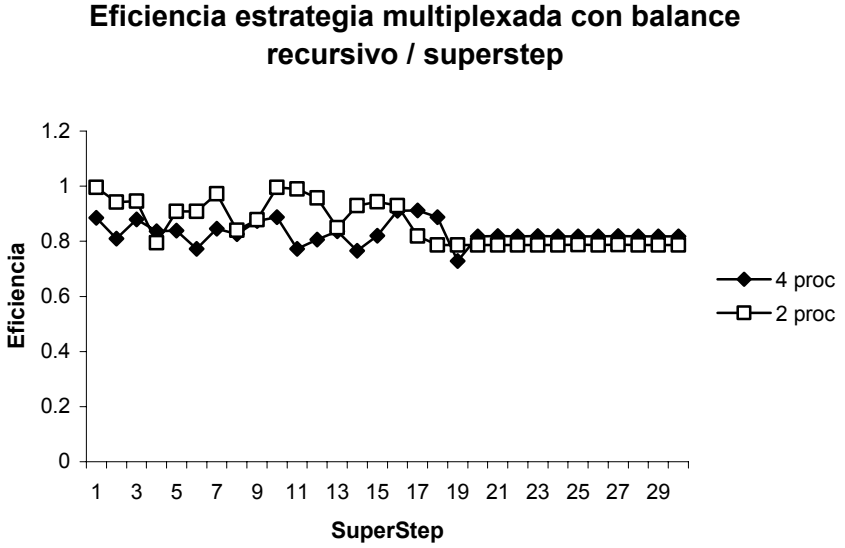

Figura 3: Eficiencia de estrategia Árbol multiplexado con balance de carga recursivo y distribución circular de consultas, 2 y 4 procesadores para estructuras SAT y EGNAT.

Una forma gráfica de evaluación del sistema recuperador es mediante una curva de "Precisión–Recall". Esta técnica consiste en medir progresivamente cada uno de los objetos recuperados versus los que deberían haberse recuperado. Después del proceso de consulta y recuperación de imágenes por parte del sistema se tiene un conjunto de elementos que se recuperaron. A este conjunto lo llamaremos ***(A)***, del cual se extrae los resultados relevantes para esa consulta, (los Verdaderos Positivos), y se denomina a este subconjunto (***B***).

La precisión es definida como: $$\frac{|\mathbf{A} \cap \mathbf{B}|}{|\mathbf{B}|}$$

y Recall es definido como: $$\frac{|\mathbf{A} \cap \mathbf{B}|}{|\mathbf{A}|}$$

Este proceso se va graficando elemento a elemento a medida que se van recuperando y permite visualizar el comportamiento del sistema frente a una consulta de una clase x. La figura 4 nos muestra esta precisión en la recuperación desde la base de datos dada una query.

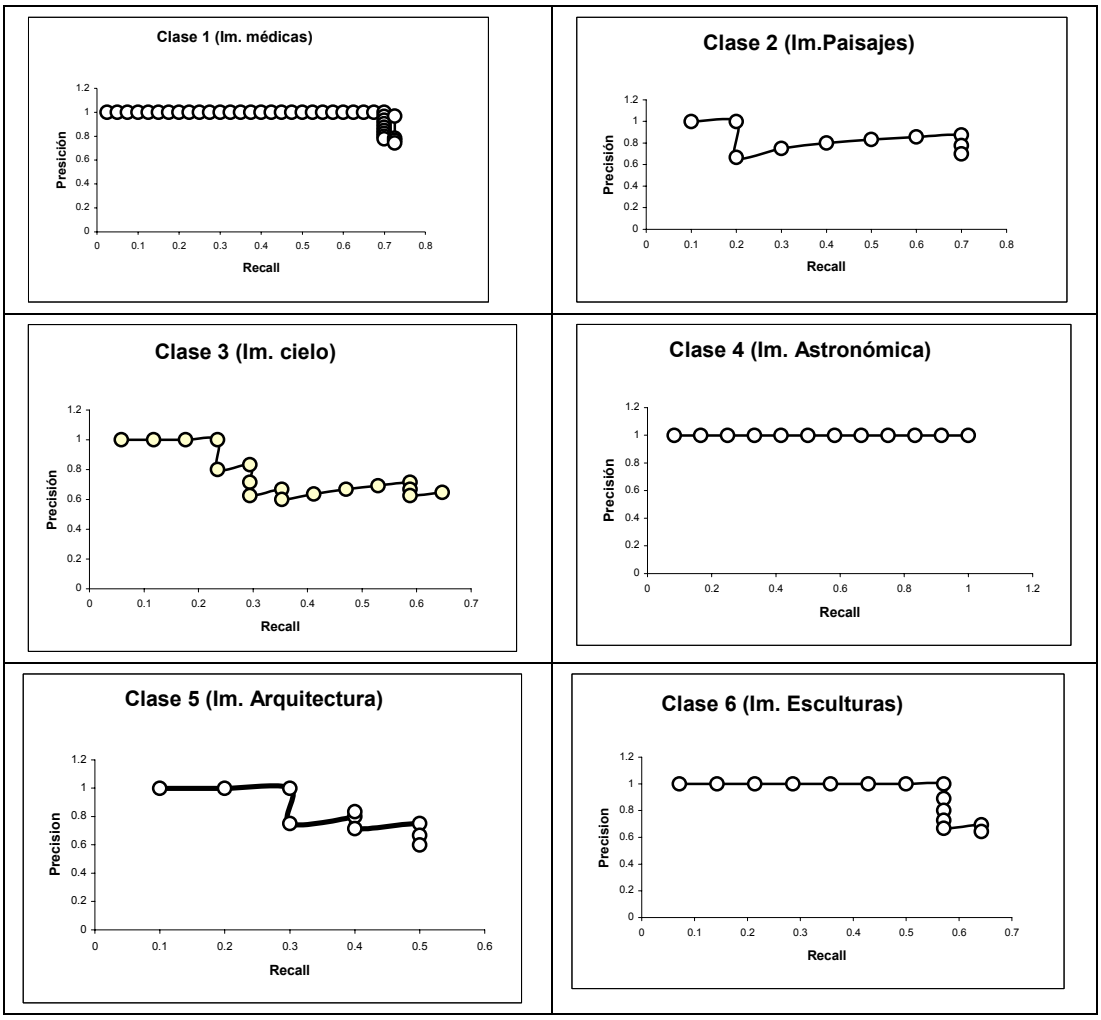

FIGURA 4**:** Curvas de Precisión – Recall para clases 1 a la 6

4. Resultados Experimentales

La base de datos utilizada en nuestros experimentos consiste en 10000 vectores de dimensión 14431 que representan 27 características correspondientes a intensidades de color y textura de regiones extraídas de 10000 imágenes. La Interfaz integradora de herramientas de búsqueda sobre estructuras espaciales de objetos cuyas características están basadas en contenido se presenta en la figura 5. Esta herramienta esta basada en PHP.

Figura 5: Interfaz integradora con selección de queries, métricas, tipos de indexación y objetos recuperados.

4.1 Comparación Con Otros Sistemas Existentes

A continuación se presentan algunas comparaciones en rendimiento del sistema propuesto con respecto a dos sistemas de similares características de recuperación y que corresponden a uno del tipo comercial, **[13]** y uno académico, **[14]**. La comparación se realizó en base a 3 clases comunes en sus bases de datos. El promedio de la precisión de las tres clases para cada sistema recuperador se puede apreciar en la Figura 6. Esta figura nos presenta la curva de precisión promedio de los sistemas comparados lo que nos indica que hasta el 20% de los elementos recuperados en tiempo online son verdaderos positivos, (imágenes que pertenecen a la misma clase que la consulta). A partir del 20% y hasta el 90% de los datos recuperados están entre 80% y 90% que corresponden a verdaderos positivos. Si tomamos en cuenta todos los verdaderos positivos que debería entregar el sistema, (100% de los datos), nos encontramos con que la precisión promedio es del 76%.

Los tiempos no fueron comparables debido a que los tres sistemas están montados sobre plataformas multiusuarios y cuyas respuestas a una consulta por parte del usuario dependen de la velocidad de conexión y del tráfico al momento de la consulta.

En la figura 7 se puede apreciar los elementos recuperados por el CBIR propuesto para seis clases de imágenes, donde la primera columna corresponde a las consultas y las restantes a los objetos recuperados.

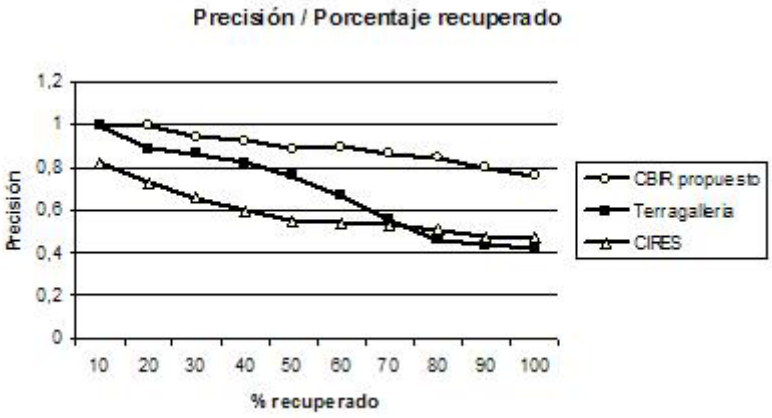

Figura 6: Comparación del CBIR propuesto con un recuperador comercial y otro del tipo académico.

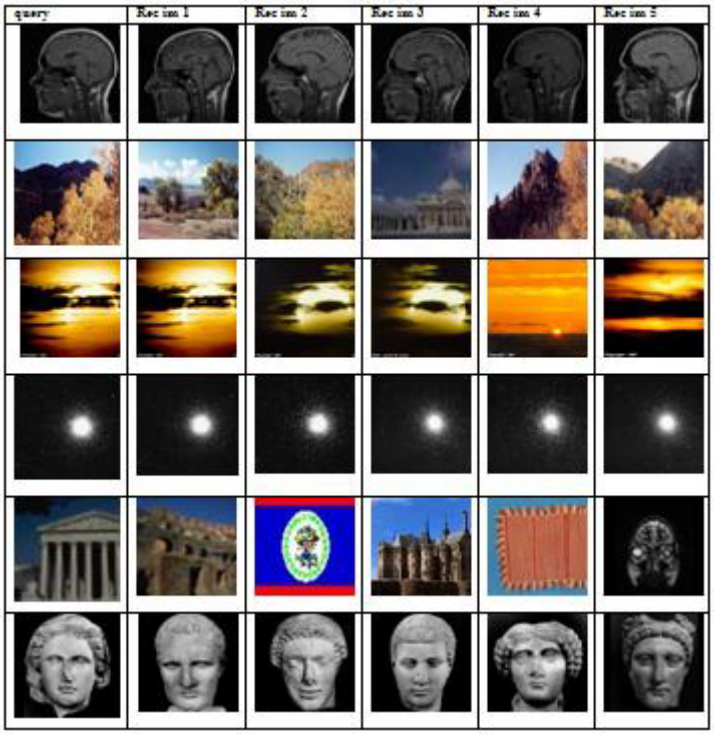

Figura 7: Elementos recuperados por el CBIR para seis clases propuestas.

5. Conclusiones

Aunque desde los años 90's que se estudian métodos que permitan recuperar imágenes almacenadas en algún sistema de base de datos, las soluciones obtenidas hasta ahora no son del todo satisfactorias. Los sistemas sencillos de Bases de Datos de Imágenes o bien no tienen capacidades de consulta o éstas son muy limitadas. Esto ha ocasionado que el trabajo presentado se centre en los CBIR que, aunque sea de una forma parcial, permite recuperar imágenes a partir de una descripción de los objetos que en ellas aparecen.

La implementación fue realizada sobre dos estructuras que han demostrado buen desempeño en búsquedas por similaridad. La eficiencia de las estructuras fueron dependientes de las dimensiones, de la cantidad de vectores en la base de datos y de si la estructura era manejada en memoria principal o secundaria. En este sentido los experimentos determinaron que a baja dimensionalidad del espacio, la estructura SAT se comportab a mejor que el EGNAT, pero en el caso contrario el mejor rendimiento lo ofrece las estructuras EGNAT.

Se considera, como aporte del presente trabajo el desarrollo de una versión paralela eficiente de un nuevo CBIR y su implantación sobre estructuras métricas, permitiendo de esta manera acercar más estas estructuras a problemas verdaderamente reales, como lo es el caso de grandes volúmenes de imágenes representados en espacios de muy alta dimensión.

Se considera, también, como parte de los aportes, el hecho de realizar análisis comparativos entre las distintas estructuras y entre distintas estrategias de paralelización de éstas.

Las estrategias propuestas consisten en:

A. Distribuir los nodos en forma multiplexada por cada uno de los procesadores que se tenga disponibles sin balancear.
B. Distribuir los nodos en forma multiplexada pero balanceada por los procesadores disponibles.
C. Distribuir los nodos en forma multiplexada pero balanceada recursiva (subdividiendo cada uno de las ramas del árbol a distribuir en caso de superar el subárbol, el promedio de nodos permitido por los procesadores).

La estrategia que mejor resultados arrojó fue la estrategia (C) seguida de la (B) con eficiencias de 0.9 y 0.8 respectivamente y utilización de base de datos de 10000 elementos.

Por otro lado, se ha observado que la cantidad de comunicación y la sincronización es muy pequeña con respecto al costo de cálculos de distancia. El número de envíos de mensajes está por debajo del 1% con respecto al número de distancia

No se ha considerado el costo de enviar los objetos de la solución. Este costo tiene que ser pagado por cualquier estrategia. Por oto lado los experimentos llegaron a término en menos de 500 supersteps, lo cual es una cantidad muy modesta de sincronización para procesar 1.000 consultas. En las estrategias propuestas de multiplexión o distribución de subárboles de la estructura se debió considerar los subárboles pertenecientes a los hijos de la raíz. Puede darse el caso que se dispongan más procesadores que hijos. En este caso no podría paralelizarse la estrategia considerando la distribución de subárboles con uno o más niveles hacia abajo. Esos subárboles generan muy pocas comparaciones de distancia y en tal caso simplemente deben tratarse usando menos procesadores que los disponibles, (puede suceder en este caso que la estructura no admita más paralelismo), o acudiendo a la duplicación de algunos subárboles en los procesadores.

El desempeño del recuperador de imágenes para estas dos estrategias propuestas se mide de acuerdo a los valores de sensibilidad y especificidad y para ambos casos los valores son 0.78 y 0.993 respectivamente.

Aunque computacionalmente el modelo propuesto sea caro para bases de datos pequeñas, (comparados con una búsqueda secuencial), los resultados serían menos costosos al trabajar con bases de datos grandes en las cuales los tiempos de recuperación serían extremadamente altos para buscadores secuenciales. La exactitud viene de la mano con el costo computacional. Este disminuye con la utilización de algoritmos eficientes como los presentados y el aumento del poder computacional que en la actualidad lo hace más económico.

El estudio que se realizó para recuperar imágenes a través del CBIR planteado entregó resultados aceptables para la poca variedad de características que se extrajeron del conjunto de imágenes.

La curva de "Precisión–Recall", nos permite medir progresivamente cada uno de los objetos recuperados versus los que deberían haberse recuperado. Mediante esta herramienta podemos observar que los gráficos correspondientes a la clase 8, no fue el adecuado para poder ser comparados con otras clases y recuperar sus similares.

El sistema recupera en promedio el 78% de las imágenes que son verdaderos positivos de las distintas clases con una sensibilidad de 78%. La excepción son las clases 5 y 7 correspondientes a arquitectura y fauna y que bajan el promedio de recuperación del sistema. La razón se debe al poco detalle en cuanto a tipo de características extraídas.

Si sólo revisamos el 50% de las imágenes recuperadas la sensibilidad corresponde al 89%, y frente a la comparación de dos recuperadores comerciales el comportamiento del sistema propuesto es aceptable (diferencias de hasta un 34% a favor del recuperador propuesto). La comparación de tiempos de los sistemas recuperadores de imágenes no pueden ser del todo comparables debido que al estar en plataformas multiusuarios dependen mucho de la comunicación cliente servidor. El recuperador propuesto fue montado sobre una red local multiusuario y los tiempos de comunicación no fueron altos frente a los recuperadores comerciales [13] y [14], con los cuales se comparó el CBIR propuesto.

La eficiencia del sistema de recuperación de datos propuesto bajo una estructura EGNAT y con estrategia de paralelización de distribución de los nodos en forma multiplexada balanceada y recursiva nos presenta un 90% .

Los tiempos promedios de recuperación de un conjunto de 20 datos similares a una consulta corresponden a aproximadamente 2 seg. para una máquina secuencial cuya base de datos comprende 1.000 imágenes y de 16 seg. para un conjunto de 10.000 imágenes.

6. Trabajo Futuro

- Incorporar al CBIR propuesto la introducción de consultas online por parte del cliente. Esta consulta puede ser un vector de rasgos o bien una imagen. Al ser una imagen se requiere extraer características de acuerdo a petición del usuario.
- Durante la selección y recuperación de imágenes, se pretende realizar una nueva consulta en base a la imagen seleccionada por el usuario del conjunto recuperado por el sistema.
- Se contempla que los módulos de extracción de características y normalización, que actualmente son parte del los módulos del servidor, sean a futuro online, con clasificadores dinámicos y módulos de enlace a estructuras distribuidas.
- Las estrategias propuestas fueron ejecutadas sobre una red Fast Ethernet 10/100 por lo que habría que considerar y hacer pruebas futuras sobre un clúster donde la comunicación entre procesadores sea más eficiente.

Referencias

[1] R. Baeza-Yates and W. Cunto and U. Manber and S. Wu, Proximity Matching using fixedqueries trees, 5th Combinatorial Pattern Matching (CPM'94), 1994.

[2] Sergei Brin, Near Neighbor Search in Large Metric Spaces. The 21st VLDB Conference, 1995.

[3] W. Burkhard and R. Keller, Some Approaches to best-match File Searching, Communication of ACM, 1973.

[4] P. Ciaccia and M. Patella and P. Zezula, M-tree : An eficient access method for similarity search in metric spaces. The 23st International Conference on VLDB, 1997.

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

[6] E. Peña, Estructuras métricas paralelas en la recuperación de Imágenes. Master's thesis Escuela de Ingeniería, Departamento de Ciencias de la Computación, Pontificia Universiadad Católica de Chile, Santiago, 2006.

[7] N. Reyes, Índices dinámicos para espacios métricos de alta dimensionalidad. Master´s thesis, Universidad Nacional de San Luis, Argentina, 2002.

[8] D.B. Skillicorn, J.M.D. Hill, and W.F. McColl. Questions and answers about BSP.
Technical Report PRG-TR-15-96, Computing Laboratory, Oxford University, 1996. Also in Journal of Scienti_c Programming, V.6 N.3, 1997.

[9] J. Uhlmann, Satisfying general proximity/similarity queries with Metric Trees. Information Processing Letters, 1991.
[10] L.G. Valiant. A bridging model for parallel computation. Comm. ACM, 33:103{111, Aug. 1990.

[11] R. Uribe, Manipulación de Estructuras Métricas en Memoria Secundaria, Master Thesis, Universidad de Chile, Chile, 2005.

[12] G. NAVARRO, **Searching in metric spaces by spatial approximation**. In Proc. String Processing and Information Retrieval (SPIRE'99), pages 141–148. IEEE CS Press. 1999.

[13] LUANG, Q-T, **SIMPLIcity, TerraGalleria Photograpy**, Terra Galleria, Recuperado en 2006, <http://www.terragalleria.com>.

[14] IQBAL, Q. and AGGARWAL, J.K. (2002) **CIRES: Content Based Image REtrieval System**, Techniques and Applications International Conference on Control, Automation, Robotics and Vision (ICARCV), Singapore, pp. 205-210, December 2-5, 2002.
<http://amazon.ece.utexas.edu/~qasim/research.htm>.