UNIVERSIDAD CENTRAL "MARTA ABREU" DE LAS VILLAS
FACULTAD DE MATEMÁTICA, FÍSICA Y COMPUTACIÓN
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

Trabajo de Diploma

Sistema para la gestión de información científico-técnica

Autor: Michel Artiles Egüe

Tutores: MSc. Leticia Arco García

Dr. Carlos Morell Pérez

Santa Clara, 2008

"Los sueños de hoy serán las realidades de mañana"

José Martí

A mi familia, por su apoyo incondicional.

A Leticia, por su guía y colaboración de todos estos años.

A mi abuelita del alma.

A mis padres queridos y mi hermanita linda.

A mi abuelo Pedro.

A mi tía Julita.

A mis abuelos maternos.

A mis primos y tíos.

A mi novia por su compresión.

A los colegas del Joven Club de Esperanza.

A todos los que me han apoyado y soportado.

Resumen

En este trabajo se diseñó e implementó el sistema GARLucene para la gestión de información científica, a partir de los resultados de la recuperación de artículos científicos usando LIUS y Lucene. Los documentos se agrupan utilizando las propiedades estructurales de sus representaciones gráficas. Además los resultados del agrupamiento se valoran mediante la validación y el etiquetamiento de los grupos. La implementación del sistema se basa en la metodología de análisis y diseño orientada a objetos, éste es extensible y reutilizable.

El sistema incluye métodos de agrupamientos jerárquicos divisivos que posibilitan organizar los resultados de procesos de recuperación de información, y por tanto, contribuir a una mejor gestión de los artículos científicos que los usuarios desean analizar. Además, se utiliza la teoría de los conjuntos aproximados para determinar los documentos más representativos por grupos y caracterizar los resultados de los agrupamientos lo que permite la validación de los grupos y agrupamientos en general.

GARLucene permite la indexación de múltiples tipos de ficheros y fue desarrollado completamente en JAVA, característica que lo convierte en un sistema multiplataforma. Además, los códigos fuente de LIUS y Lucene se encuentran totalmente disponibles, por lo que se pudo interactuar con facilidad para indexar, recuperar y procesar los documentos.

Abstract

This describes the design and implementation of the GARLucene system for management of scientific information, based upon the results of the gathering of scientific articles using LIUS and Lucene. Documents are clustered using the structural properties of its graphic representation. The results of the clustering are evaluated through the validation and labeling of the clustering groups. The system design was carried out following the guidelines of the analysis and design oriented to object methodology guaranteeing its extensibility and reusability.

The system also includes two implementations of the hierarchic divisive clustering that enables the system to organize the result of the information retrieval and, therefore, to contribute to a better management of the scientific articles that users wish to analyze. Besides, Rough Sets theory is used to determine the most representative documents on each group and to characterize the results of the clustering, making possible the validation of the formed groups and the clustering in general.

GARLucene allows the indexing of different file types. It was fully developed using Java, feature that makes this program multiplatform. Furthermore, the source codes are fully available, making it easy to interact with for indexing, retrieving and processing documents.

Tabla de contenidos

Introducción

El Siglo XX es reconocido por el enorme desarrollo tecnológico que lo caracterizó. También se le llamó "era de la información" o el de la aparición de la "sociedad de la información". Esta en sí misma no supone ninguna ventaja, su sistematización es la que aporta ese valor añadido. A diferencia de hace unos pocos años, hoy estamos "invadidos" de información, pues se estima que cada 20 meses aproximadamente la cantidad de información en el mundo se duplica, el problema es cómo procesarla y usarla en beneficio propio. Por tanto, la explotación del conocimiento requiere de alguna herramienta que lleve a cabo esa sistematización de la información.

Los sistemas de información deben ser gestionados de modo que superen ampliamente la mera acumulación, el ordenamiento y la facilitación de la búsqueda, estos sistemas deben generar un "plus" de información, producto de la sinergia de los contenidos.

Hoy se habla de la sustitución de "información" por "conocimiento" y de "sistemas que permiten procesar información" por sistemas que generan o entregan conocimientos, es decir, que guíen una toma de decisión óptima. Dando lugar a la Gestión del Conocimiento (GC) (Bueno 2001, Canals 2003, Grau 2007). La GC es el proceso sistemático de buscar, organizar, filtrar y presentar la información con el objetivo de mejorar la compresión de las personas en un área específica de interés (Davenport and Prusak 1997).

Las Tecnologías de la Información y la Comunicación (TIC) juegan un papel fundamental en la GC; en (Passoni 2005) se afirma que estas se vuelven el factor clave en el proceso de creación y aplicación del modelo de GC y que la Intranet de una institución es el recurso apropiado para la instalación del sistema de GC. La minería de datos se ha convertido en un aliado de la gestión del conocimiento a partir de la información almacenada en las bases de datos de la institución (Holsheimer and Siebes 1994, Fayyad 1996a, Fayyad 1996b) .

La minería de datos es una fase del "descubrimiento de conocimiento en bases de datos" (extracción no trivial de información implícita, desconocida, y potencialmente útil de los datos (Lezcano 2002)) que integra los métodos de aprendizaje y estadísticas para obtener hipótesis

de patrones y modelos. La limitante que existe es que las técnicas de minería de datos procesan información estructurada, y sin embargo, aproximadamente un 80% de la información está almacenada en forma textual no estructurada, de ahí que se desarrollen actualmente técnicas de "minería de textos" (Dürsteler 2001), que pretenden algo similar a la minería de datos, porque identifican relaciones y modelos, pero desde la información no cuantitativa. Es decir, proveen una visión selectiva y perfeccionada de la información contenida en documentos, sacan consecuencias para la acción y detectan patrones no triviales e información sobre el conocimiento almacenado en las mismas.

Muchas veces recuperamos información sobre un tema y no sabemos cómo enfrentarnos a ella. Una forma de facilitarle los resultados de sistemas de recuperación a los usuarios es mostrándole grupos homogéneos de documentos afines. Pero no tenemos que referirnos solamente al resultado de una búsqueda, en la información estática que tenemos almacenada en nuestra computadora o que se encuentra disponible en un servidor, sería útil obtener las interrelaciones antes de comenzar a estudiar un conjunto de artículos.

A veces es necesario enfrentarse a colecciones textuales ya sea para realizar un análisis de artículos científicos, para organizar materiales por equipos de estudiantes para la docencia, para organizar por temáticas los artículos que le han llegado al comité científico del programa de un evento, o sencillamente para tener una idea de las asociaciones que existen a partir de un resultado de un proceso de recuperación de información.

De las limitantes anteriormente expuestas se deriva el **problema científico a resolver.** Los sistemas de Gestión del Conocimiento existentes actualmente no han resuelto totalmente el procesamiento de la información científico-técnica, explotando tanto las facilidades de los sistemas de recuperación de información, como particularizando el análisis en las características de los artículos científicos ni su estructura. Por otra parte, los métodos aplicados para agrupar los documentos usualmente requieren la especificación de valores iniciales para los parámetros que influyen en los resultados finales y no explotan las propiedades estructurales de los documentos representados en un grafo.

Para contribuir a la solución del problema científico antes planteado, se formularon las siguientes **hipótesis generales de investigación**:

¿Utilizar los resultados de los sistemas de recuperación de información para procesar información científico-técnica (agrupar, seleccionar, evaluar, etiquetar) permite ofrecer a los usuarios el resultado de la recuperación de una forma organizada y genera conocimiento a partir de la información recuperada?

¿Explotar las propiedades estructurales de los documentos representados en grafos permite un mejor agrupamiento?

En conformidad con la hipótesis de investigación identificada, el **objetivo general de la investigación** consiste en diseñar e implementar un sistema para la gestión de información científica, a partir de los resultados de la recuperación de artículos científicos obtenidos mediante las herramientas de software apropiadas, agrupando los documentos utilizando las propiedades estructurales de sus representaciones gráficas y verificando los resultados del agrupamiento utilizando la teoría de los conjuntos aproximados.

Para cumplimentar el **objetivo general**, los siguientes objetivos específicos se proponen:

- ✓ Diseñar e implementar el sistema para la gestión de la información científica que combine métodos de agrupamiento con los resultados de la recuperación textual.
- ✓ Evaluar las facilidades de LIUS[1] y Lucene[2] en el proceso de recuperación de información y preprocesamiento textual.
- ✓ Aplicar métodos de agrupamiento basados en la intermediación de aristas.
- ✓ Verificar los resultados del agrupamiento mediante la validación y etiquetamiento de los grupos obtenidos.

[1] LIUS: Iniciales del inglés *Lucene Index Updating Search*

[2] Lucene: Biblioteca para la Recuperación de Información de código abierto desarrollada en Java

Este documento se estructuró de la manera siguiente: En el capítulo 1 se abordan los sistemas de manipulación de documentos, así como aquellos tópicos de la minería de textos que pueden contribuir al mejoramiento de estos sistemas para gestionar información y conocimiento. Particularmente se tratará acerca de la recuperación de información, agrupamiento y su valoración. En el segundo capítulo se describirá el Sistema para la Gestión de Artículos Científicos recuperados a partir de Lucene (GARLucene). Este sistema puede asistir al usuario al enfrentarse a una colección de artículos científicos mediante la indexación, recuperación, agrupamiento, evaluación y obtención de los documentos más representativos de los grupos. El capítulo 3 presenta el manual de usuarios de GARLucene, donde aparece una descripción detallada de cada una de las opciones del sistema. Finalmente, se realizan las conclusiones y recomendaciones del trabajo.

1 ACERCA DE LOS SISTEMAS DE MANIPULACIÓN DE DOCUMENTOS

La información es cada día más creciente, heterogénea, diversa, dinámica y constituye una fuente importante de conocimiento. Gestionar la información y el conocimiento son retos que tienen hoy las organizaciones. Enfrentarlos en dominios textuales es un desafío aún mayor, ya que se hace necesario el desarrollo de sistemas de manipulación de documentos que contribuyan a dicha gestión. Dentro de estos sistemas, aquellos que se encargan de recuperar, organizar y analizar de forma precisa y eficiente la información, y recomendar acciones a partir del procesamiento realizado, constituyen la principal motivación de este trabajo.

En este capítulo se presentará el estado del arte de las características principales de los sistemas manipuladores de documentos, así como aquellos elementos de la minería de textos que contribuyen al desarrollo de estos sistemas, y a su vez, a la gestión de información y conocimiento en las organizaciones. Además, se analizará en detalles las potencialidades del agrupamiento y las formas de evaluarlo y caracterizarlo, para refinar resultados de procesos de recuperación de la información.

1.1 Gestión de información y conocimiento: manipulación de documentos

La información y el conocimiento surgen de acciones humanas que interconectan señales, signos y artefactos en diversos medios. El conocimiento se fundamenta en una acumulación de experiencias, mientras que la información depende de la agregación de los datos. El espacio de información está dado por su codificación, abstracción y difusión (Choo, Detlor et al. 2000). Crear categorías que faciliten la clasificación de fenómenos y minimizar el número de categorías necesarias, constituyen los procesos correspondientes a las dos primeras dimensiones, mientras que la tercera combina éstas. Las organizaciones requieren utilizar la información no sólo para darle significado en su entorno, sino para crear nuevo conocimiento, compartirlo y tomar decisiones (Choo, Detlor et al. 2000). Sus principales incentivos para la gestión del conocimiento son (Tiwana 2000, Müller, Spiliopoulou et al. 2005): buscar, aportar, contribuir, diseminar, explotar y evaluar el conocimiento. Por tanto, se hace necesario

transformar información en conocimiento: estructuración de datos e información y la acción humana sobre éstos (Choo, Detlor et al. 2000).

La aplicación que aquí se propone contribuye a la primera forma de transformación, porque manipula documentos mediante el agrupamiento y post-agrupamiento revela el orden de los datos e impone patrones en los grupos descubiertos. Al manipular documentos, se está trabajando con un conocimiento objetivo, que ha sido formalizado acorde a algún esquema de codificación (por ejemplo, patente, reporte, artículo, norma). Este conocimiento es clasificado: explícito. Otras dos clasificaciones son: tácito, cuando es personal, en un contexto específico y se hace muy difícil de formalizar, e implícito, cuando es subjetivo (Fürnkranz, Scheffer et al. 2006, Choo, Detlor et al. 2000).

El conocimiento explícito tiene gran importancia porque codifica aprendizaje pasado, habilita la coordinación de actividades y funciones, reduce el procesamiento de información mediante la estipulación de premisas, criterios y opiniones, y significa habilidades, técnicas y procedimientos que contribuyen a una autopresentación de la calidad de las organizaciones (Fürnkranz, Scheffer et al. 2006, Choo, Detlor et al. 2000). Además, se comunica fácilmente, aunque transferirlo requiere conocimiento colateral del receptor para entenderlo y aplicarlo. Conocimiento colateral que tiene naturaleza tácita porque los expertos interpretan el significado de la nueva información y surgen incógnitas cuando tratan de usarlo. Por ejemplo, un investigador que se enfrenta al resultado de una colección de artículos científicos automáticamente agrupada, y a la determinación automática de los documentos más representativos y relacionados con cada grupo, requiere utilizar conocimiento tácito para interpretar adecuadamente los resultados y tomar decisiones en el estudio del arte en función de la recomendación recibida automáticamente.

Una de las formas de creación de conocimiento se alcanza moviéndolo desde el nivel de individuos hasta el de grupos, durante cuatro etapas del ciclo de conversión del mismo: socialización, exteriorización, combinación e internalización (Nonaka and Takeuchi 1995). La aplicación que se propone contribuye a la manipulación del conocimiento a partir del conocimiento codificado e y tributa a las dos últimas etapas del ciclo. Por un lado, se crea conocimiento explícito mediante el agrupamiento y valoración de los grupos textuales a partir

de la recopilación del conocimiento explícito de múltiples fuentes. Así, se pone en práctica una de las formas de llevar a cabo la combinación: producir nuevo conocimiento explícito mediante la combinación (categorización y organización) del conocimiento explícito acumulado (Choo, Detlor et al. 2000). Por otro, estos resultados recomiendan a los usuarios cómo enfrentarse a la colección textual y por tanto tienen más elementos para personificar el conocimiento explícito y aumentar sus experiencias en el tácito.

Ya se conoce que las organizaciones deben considerar la opción de servicios de conocimiento para lograr la integración de fuentes locales (por ejemplo, intranet local, servidores de ficheros, sitios públicos), intraredes y extraredes (Tiwana 2000). Pero sólo esto no es suficiente, es necesario utilizar adecuadamente las componentes de las tecnologías de la información para desarrollar gestores de conocimiento. Los sistemas de gestión de documentos, entre otras componentes como sistemas de almacenamiento, sistemas de soporte de búsquedas, modelos de categorización y análisis de contenidos, ontologías y servicios de control de acceso y coordinadores de trabajo colaborativo, han tomado un gran auge en la actualidad (Müller, Spiliopoulou et al. 2002).

Docyoument es un ejemplo de sistema manipulador de documentos creado por Media-style[3]. Éste encuentra y lista información relevante en varias fuentes a partir de tópicos y preguntas definidas por los usuarios, genera reportes y resúmenes de los documentos, encuentra contenidos relacionados a partir de la información importada. Por otro lado, Text Miner[4] es un grupo de herramientas para descubrir y extraer conocimiento desde documentos textuales. Otros ejemplos los constituyen:

- ✓ Worldox[5], que permite la seguridad de los documentos, el control del acceso, búsquedas a texto completo, visualizadores para formatos diferentes de documentos, archivar y guardar la historia de un documento.

[3] Creación, publicación, gestión, organización, descubrimiento y análisis de contenidos. http://www.media-style.com
[4] http://www.sas.com/technologies/analytics/datamining/textminer
[5] http://www.worldox.com/

- ✓ Autonomy[6], por su parte, permite un servicio de búsqueda inteligente en Internet a partir de fuentes en 65 lenguajes y deriva el significado de las palabras dependiendo del contexto, genera automáticamente los perfiles de los usuarios.
- ✓ Knexa[7] es un ejemplo de plaza de mercado electrónico para conocimiento documentado.

Otros sistemas para el comercio de conocimiento en Internet fueron identificados por el Instituto Kaieteur[8]: Knowinc.com, Keen.com, Yet2.com, iExchange.com, Saba.com, cordis.lu, IQ4Hire.com, petrocore.com y eBrainx.com.

Estos sistemas y herramientas han sido desarrollados en su inmensa mayoría por importantes compañías que han invertido gran capital en la manipulación de documentos para contribuir a la gestión de información y el conocimiento; por ejemplo: Autonomy[9], ClearForest[10], IBM Intelligent Miner for Text[11], LexiQuest[12], Teragram[13] y SAS[14].

Los sistemas mencionados con anterioridad tienen un alto precio en el mercado internacional, debido esencialmente a los beneficios que les reportan a las organizaciones. Las universidades cubanas requieren la introducción de estos tipos de sistemas para el desarrollo de las actividades docentes y científicas. Así, los resultados teóricos de esta investigación se han aplicado al desarrollo de sistemas que permiten gestionar documentos, permitiendo la abstracción, descripción y modelación de las estructuras de información mediante el agrupamiento y post-agrupamiento de colecciones textuales, tributando a la gestión de información y conocimiento en las organizaciones.

[6] http://www.autonomy.com
[7] http://www.knexa.com
[8] Instituto para la gestión del conocimiento (Kaieteur Institute for Knowledge Management http://www.kikm.org)
[9] http://www.autonomy.com
[10] http://www.clearforest.com
[11] http://www-306.ibm.com/software/data/
[12] http://www.lexiquest.com/
[13] http://www.teregram.com
[14] http://www.sas.com

1.2 Recuperación de información

Recuperación de información (Information Retrieval; IR) puede ser definida como una aplicación de las tecnologías de la computación para la adquisición, organización, almacenamiento, recuperación, y distribución de información. IR está estrechamente relacionado con elementos prácticos y teóricos sobre la mejora de la tecnología de los motores de búsqueda, incluyendo la construcción y mantenimiento de grandes repositorios de información. En años recientes, los investigadores han incrementado y expandido su preocupación acerca de la bibliografía y búsqueda a texto completo en repositorios en la Web, tanto en datos de hipertextos pertenecientes a bases de datos como en multimedia.

La recuperación de información es una actividad, y como la mayoría de las actividades tiene sus propósitos. Un usuario de un motor de búsqueda comienza con la información que él necesita, para lo cual realiza una consulta con el objetivo de encontrar documentos relevantes. Esta consulta puede que no sea la mejor conexión que él necesita para encontrar lo que desea. Ésta puede contener errores ortográficos, desaprovechar palabras o haber realizado una selección pobre de las palabras. Sin embargo, esta es la única pista que tiene el motor de búsqueda respecto al objetivo de los usuarios.

Generalmente se dice que los documentos resultantes de la búsqueda pueden ser más o menos relevantes respecto a la consulta, pero, estrictamente hablando, esta expresión es errada. Los usuarios juzgan la relevancia respecto a la información que necesitan, no respecto a la consulta. Si se retornan documentos irrelevantes, los usuarios pueden o no darse cuenta de cuáles son los irrelevantes, y pueden o no encontrar la forma de mejorar la consulta.

1.3 El agrupamiento para refinar la recuperación de información

La mayoría de los sistemas que se desarrollan en la actualidad para gestionar conocimiento e información, como se observó al final del epígrafe 1.1, están más dirigidos al Comercio del Conocimiento (CC) en Internet que a la gestión del conocimiento en las organizaciones. Las diferencias entre estos dos enfoques pueden apreciarse en el Anexo 1. Las universidades e institutos de investigación, constituyen un caso particular de organizaciones donde la gestión del conocimiento juega un rol importante porque tienen una larga historia, sus procesos son

estables, generan y preservan valiosa información proveniente de diversos procesos, tienen acceso a importantes fuentes de información externa, poseen un capital humano bien capacitado y buen desarrollo de las tecnologías de la información.

El desarrollo de sistemas gestores de información y conocimiento se requiere en estas organizaciones en función de tareas específicas que tienen que asumir y en dependencia de las características y recursos de las mismas. Algunas de estas tareas son: recomendar a investigadores y docentes cómo enfrentarse a grandes volúmenes de artículos científicos al comenzar la revisión del estado del arte de un tema de investigación, ya que toda nueva investigación comienza con una amplia revisión bibliográfica sobre el tema en cuestión, organizar materiales por equipos de estudiantes para la docencia, organizar por temáticas los artículos que le han llegado al comité científico del programa de un evento, o tener una idea de las asociaciones que existen entre los documentos recuperados y así organizarlos.

Estas tareas son necesidades de automatización sobre todo en las universidades cubanas, donde el acceso a Internet es en alguna medida limitado, existen grandes depósitos centrales de información porque ésta se comparte y publica y el gran contenido de trabajo hace que el tiempo sea limitado al enfrentarse a revisiones bibliográficas. Además, estas tareas, en su mayoría, tributan a la producción científica que se lleva a cabo en las universidades, actividad relevante que se refleja en un gran número de los indicadores para la medición del capital intelectual en estas organizaciones (Bueno 1999).

Automatizar este tipo de tareas requiere la integración de varias áreas del saber: el descubrimiento de conocimiento en bases de datos, la minería de datos y de textos. Esta última integra la recuperación y extracción de información, el análisis de textos, el resumen, la categorización, la clasificación, el agrupamiento, la visualización, la tecnología de bases de datos, el aprendizaje automático y la minería de datos (Dixon 1997, Tan 1999). Dentro de éstos, el agrupamiento y los procesos post-agrupamiento permiten organizar la información, determinar información relevante y crear nuevo conocimiento a partir de la información disponible. Sobre todo el agrupamiento jerárquico que representa información a diferentes niveles de detalles y es conveniente para un buscador interactivo. En el tope del dendrograma, la colección es organizada en categorías más generales, al descender, las categorías

incrementan su especificidad. Típicamente a cada grupo de documentos se le asignan descriptores sobre su contenido. Así, uno de los objetivos del agrupamiento es mejorar la habilidad de los usuarios para acceder a la colección y los descriptores o etiquetas de alguna forma ofrecen conocimiento sobre la colección (Treeratpituk and Callan 2006).

1.4 Agrupamiento

El análisis de grupos es descrito como una herramienta para el descubrimiento porque tiene la potencialidad de revelar relaciones basadas en datos complejos no detectadas previamente. Los algoritmos de agrupamiento son usados para encontrar una estructura de grupos que se ajuste al conjunto de datos, logra homogeneidad dentro de los grupos y heterogeneidad entre ellos (Anderberg 1973).

Debe existir un alto grado de asociación entre los objetos de un mismo grupo y un bajo grado entre los miembros de grupos diferentes (Anderberg 1973). Cuando el agrupamiento se basa en la similitud de los objetos, se desea que los objetos que pertenecen al mismo grupo sean tan similares como se pueda y los objetos que pertenecen a grupos diferentes sean tan diferentes como sea posible (Höppner, Klawonn et al. 1999, Kruse, Döring et al. 2007). En otras palabras, seguir el principio de maximizar la similitud intra-grupo y minimizar la similitud inter-grupo.

El concepto de "similitud" tiene que ser especificado acorde a los datos. En la mayoría de los casos los datos son vectores de valores reales, entonces se requieren algunas medidas (distancias, similitudes, o disimilitudes) para cuantificar el grado de asociación entre ellos. Las medidas más usadas para comparar objetos se muestran en el Anexo 3.

Por otra parte, un reto para la minería de datos es descubrir grupos en datos que al relacionarse forman una estructura interesante para el análisis. Este tipo de datos ha tenido una mejor descripción cuando se representa como una colección de objetos interrelacionados y enlazados (Getoor and Diehl 2005). El enlace entre objetos es un conocimiento que puede ser explotado en el agrupamiento, ya que rasgos de objetos enlazados están correlacionados, y es probable la existencia de enlaces entre objetos que tienen elementos comunes. Así, varios métodos parten de representar los objetos y sus relaciones en un grafo y explotan su topología para descubrir

los grupos. Estas propuestas ven el conjunto de datos desde la perspectiva de las conexiones entre los objetos más que los objetos en sí mismos (Girvan and Newman 2002b). Los conjuntos de datos pueden intrínsicamente formar un grafo o se pueden obtener grafos de similitud a partir de la matriz de similitud entre los objetos.

En la actualidad se presupone que el conocimiento de la estructura de los datos es tan importante como los objetos en sí. Ese conocimiento puede ayudar a descubrir grupos que se ocultan en las comunicaciones entre los objetos (Baumes, Goldberg et al. 2005, Getoor and Diehl 2005, Tong and Faloutsos 2006). Propiedades de los grafos, sobre todo cuando éstos representan redes complejas, pueden ser indicadores importantes para el agrupamiento. Estas propiedades, en su mayoría, son computacionalmente difíciles de verificar, de ahí que muchos algoritmos sobre grafos tengan una complejidad temporal exponencial (Wu, Garland et al. 2004).

1.4.1 Clasificación de las técnicas de agrupamiento

En la literatura científica se ha propuesto una variedad de métodos de agrupamiento. Los métodos se clasifican siguiendo varios criterios: tipo de los datos de entrada del algoritmo, criterios para definir la similitud entre los objetos, conceptos en los cuales se basa el análisis y forma de representación de los datos, entre otros. Una clasificación general distingue dos tipos: aquellos que forman particiones y los jerárquicos (Han and Kamber 2001, Kruse, Döring et al. 2007).

Dado un número positivo *k*, el objetivo de métodos que forman particiones es encontrar la mejor partición de los datos en *k* grupos basada en una medida de similitud dada y conservar el espacio de particiones posibles en *k* subcojuntos solamente. La mayoría de los algoritmos que siguen esta técnica son esencialmente basados en prototipos, comienzan con una partición inicial, usualmente aleatoria, y proceden con su refinamiento (Kruse, Döring et al. 2007).

Los algoritmos jerárquicos, por su parte, hacen una descomposición jerárquica de los objetos. Dentro de ellos, los aglomerativos (bottom-up), comienzan considerando que cada objeto constituye un grupo, por tanto comienzan con tantos grupos como objetos tiene la colección, y sucesivamente van uniendo grupos, hasta que todos los objetos formen un único grupo,

generalmente considerando una medida de distancia. Mientras que los divisivos (top-down) consideran inicialmente que existe un único grupo al cual pertenecen todos los objetos y sucesivamente van dividiendo los grupos en grupos más pequeños, hasta que cada grupo contenga un único objeto. La construcción del dendrograma se puede detener por criterios automáticos o del usuario. Muchas veces combinar o dividir grupos es comprometido, y no puede ser deshecho o refinado.

Los métodos jerárquicos aglomerativos tienen problemas en el análisis de grupos de objetos representados en un grafo. Un problema es que fallan con cierta frecuencia al encontrar grupos correctos en grafos donde se conoce la estructura, lo cual hace difícil tener credibilidad en los casos donde funcionan correctamente. Otro problema es su tendencia a encontrar solamente los centros de los grupos y no incluir la periferia. Los nodos centrales de un grupo usualmente tienen una similitud alta, y por lo tanto son conectados tempranamente en un proceso aglomerativo, pero los nodos de la periferia que no tienen una similitud tan fuerte con los otros tienden a quedar abandonados. Generalmente estos nodos tienen un único enlace al grupo al que pertenecen, por eso, los métodos aglomerativos usualmente fallan al ubicarlos correctamente (Newman and Girvan 2004).

Los métodos jerárquicos organizan los datos en una secuencia anidada de grupos, que puede ser visualizada en forma de una jerarquía o árbol. Basándose en la jerarquía es posible decidir cuál es el número de grupos para el cual los datos están mejor representados según un propósito dado. Usualmente el número (verdadero) de grupos para un conjunto de datos dado se desconoce a priori. Sin embargo, al usar métodos que crean particiones usualmente se requiere especificar el número de grupos como un parámetro de entrada. Estimar este número es de gran interés, algunos sugieren la aplicación de técnicas estadísticas, otros el remuestreo, y muchas veces métodos que obtienen particiones que forman dendrogramas se han integrado con este propósito. Así, los resultados de métodos jerárquicos son refinados mediante la relocalización iterativa de puntos en la conformación de particiones.

Otros tipos de métodos han emergido para el análisis de grupos, principalmente motivados en problemas específicos de minería de datos (Han and Kamber 2001). El agrupamiento basado en densidad (Kriegel and Pfeifle) agrupa objetos vecinos de un conjunto de datos basándose

en condiciones de densidad. Éstos difieren de los algoritmos que obtienen particiones mediante la relocalización iterativa de puntos a partir del número de grupos. Otras clasificaciones dividen las técnicas de agrupamiento en estadísticas y conceptuales (Halkidi, Batistakis et al. 2001b), otros le llaman agrupamiento incompleto o heurístico a aquellos algoritmos que utilizan métodos geométricos y técnicas de proyección (Höppner, Klawonn et al. 1999), otras propuestas agrupan utilizando redes neuronales artificiales (por ejemplo, mapas autoorganizativos (Self Organizing Maps; SOM)) (Halkidi, Batistakis et al. 2001b), algunos agrupamientos se basan en modelos (Brun, Sima et al.) encontrando buenas aproximaciones de los parámetros del modelo que mejor ajusten a los datos. Por otra parte, el agrupamiento basado en celdas (grid-based clustering) es esencialmente propuesto para la minería de datos espaciales (Halkidi, Batistakis et al. 2001b). Otra clasificación, no mutuamente excluyente a las ya presentadas, considera la forma de manipular la incertidumbre en términos del solapamiento de los grupos: agrupamiento duro y borroso (Höppner, Klawonn et al. 1999). Las técnicas duras pueden ser deterministas o con solapamiento, mientras que las borrosas se subdividen en probabilísticas y posibilísticas (Kruse, Döring et al. 2007). Algunos algoritmos abordan el agrupamiento como un problema de optimización, basándose en funciones objetivos que asignan a cada posible partición un valor de calidad o error que tiene que ser optimizado (Höppner, Klawonn et al. 1999). Otra clasificación divide los algoritmos en incrementales, cuando trabajan con una colección de datos dinámica (por ejemplo, flujo de noticias (Gaber and Yu 2006)), o estáticos, cuando las colecciones de datos son estáticas. El análisis mixto de grupos (joint cluster analysis) es otra técnica de agrupamiento que integra aquellos métodos que trabajan tanto con las propiedades endógenas de los objetos así como las relaciones que existen entre ellos (Ester, Ge et al. 2006, Baumes, Goldberg et al. 2005). El agrupamiento basado en restricciones (constraint-based clustering) reúne a aquellos algoritmos que consideran aspectos más significativos acorde a los requerimientos de la aplicación (restricciones en el espacio de datos o de los usuarios) (Halkidi, Batistakis et al. 2001a).

1.4.2 Principales algoritmos para el agrupamiento

Existen varios algoritmos que crean particiones duras. Uno de los más usados es el *k*-medias (*k*-means) que tiene una complejidad temporal $O(Ikn)$[15] (Jain and Dubes 1988, Kaufman and Rousseeuw 1990, McQueen 1967, Xiong, Wu et al. 2006). Este algoritmo no funciona bien con grupos que no tengan forma convexa y requiere que el número de grupos a obtener sea especificado a priori, por tanto requiere un cierto conocimiento del dominio, ya que es sensible a cómo se hizo inicialmente la partición. A partir de él se han derivado varios como el *x*-medias (*x*-means) para una estimación eficiente del número de grupos, el conjunto *k*-medias (batch *k*-means) y el *k*-medias incremental (incremental *k*-means), y su variante mejorada medias (means) (Berry 2004), PAM (Kaufman and Rousseeuw 1990), y sus variantes mejoradas CLARA y CLARANS (Ng and Han 1994), y la modificación presentada en (Agarwal and Mustafa 2004). Todos intentan resolver las desventajas del *k*-medias, pero en su mayoría son costosos computacionalmente. Estos algoritmos funcionan bien cuando los datos tienen baja dimensionalidad y los grupos están bien separados y tienen alta densidad. Otros algoritmos que superan los resultados del *k*-medias son aquellos que utilizan análisis discriminatorio de grupos (Torre and Kanade 2006), K-SVMeans (Bolelli, Ertekin et al. 2007), este último auxiliándose de su combinación con SVM (Bordes, Ertekin et al. 2005).

Primero el más lejano (Farthest First) es otro algoritmo que mejora el *k*-medias (Hochbaum and Shmoys 1985). Parte de una selección aleatoria de los centros de grupos, calcula la distancia de cada instancia al centroide más cercano, y la instancia que quede más lejana del centroide más cercano es seleccionada como el centroide de un grupo. Este proceso es repetido hasta que el número de grupos sea mayor que un umbral especificado. Este algoritmo es utilizado en el agrupamiento de documentos (Liu, Cai et al. 2006).

EM (Bradley, Fayyad et al. 1998) y su mejora FREM (Ordonez and Omiecinski 2002) asignan a cada instancia una distribución de probabilidad de pertenencia a cada cluster. Aunque manipulan datos de alta dimensionalidad, realizan un refinamiento muy costoso.

[15] Se utiliza *I* para indicar número de iteraciones, *n* número de objetos, *k* número de grupos y *m* número de aristas.

La teoría de grafos ha sido una herramienta valiosa para desarrollar modelos de abstracción para el agrupamiento, proporcionando el formalismo matemático requerido. Sus conceptos básicos han sido utilizados para el desarrollo de algoritmos de agrupamiento, e incluso índices de validación. Los grafos proveen modelos estructurales para el análisis de grupos. Ejemplos de algoritmos que parten de una representación gráfica de los objetos a agrupar son Estrella (Star) (Aslam, Pelekhov et al. 1998) y sus extensiones (Gil-García, Badía-Contelles et al. 2003, Pérez and Medina 2007, Medina and Pérez 2007) que generan cubrimientos sobre los datos. Estrella Extendido (Extended Star) es independiente del orden de los datos y obtiene menor número de grupos respecto al algoritmo Estrella (Gil-García, Badía-Contelles et al. 2003). Los algoritmos Estrella Generalizada (Generalized Star) (Pérez and Medina 2007) y Estrella Condensada (Condensed Star; ACONS) (Medina and Pérez 2007) introducen nuevos conceptos de estrella y obtienen un menor número de grupos. No requieren especificar el número de grupos a obtener pero si son sensibles al umbral de similitud fijado inicialmente.

Los principales algoritmos basados en densidad son DBSCAN (Ester, Kriegel et al. 1996) y DENCLUE (Hinneburg and Keim 1998). Ambos tienen una complejidad $O(n \log n)$, no funcionan correctamente con datos de alta dimensionalidad y dependen altamente de los parámetros iniciales. OPTICS (Ankerst, Breunig et al. 1996) y el algoritmo propuesto en (Kriegel and Pfeifle 2005), son variantes mejoradas del DBSCAN. Otros algoritmos basados en densidad son el reportado en (Ruiz-Shulcloper, Alba-Cabrera et al. 2000, Qian, Zhang et al. 2004, Dourisboure, Geraci et al. 2007). En (Falkowski, Bartelheimer et al. 2006, Hu and Wu 2007) también se proponen algoritmos basados en densidad local de los nodos que utilizan las características de los grafos scale-free (Kalisky, Sreenivasan et al. 2006, Fortunato, Freeman et al. 2006, Stumpf, Wiuf et al. 2005).

Algunos algoritmos basados en celdas son STING (Wang, Yang et al. 1997), WaveCluster (Sheikholeslami, Chatterjee et al. 1998, Sheikholeslami, Chatterjee et al. 2000) y CLIQUE (Agrawal, Gehrke et al. 1998). Estos algoritmos son escalables, tienen complejidad $O(n)$, pero no son buenos para datos con alta dimensionalidad, porque se focalizan en la modelación de la estructura geométrica de objetos en el espacio y no dependen de una medida de distancia.

El agrupamiento basado en densidad permite descubrir grupos de varias formas, mientras que el agrupamiento basado en celdas se conoce por su alta velocidad. AGRID, CLONE (Zhao, Zhang et al. 2004) y GARDEN (Orlandic, Lai et al. 2005) son propuestas que combinan ambos enfoques.

Las técnicas jerárquicas han sido muy utilizadas en problemas de minería de datos (Jonyer, Cook et al. 2002), a pesar de tener alta complejidad temporal, generalmente cuadrática. BIRCH (Zhang, Ramakrishnan et al. 1996) es una variante con complejidad lineal, pero no descubre grupos con calidad, requiere de parámetros de entrada que pueden forzar el tamaño de los grupos, es sensible al orden de los datos de entrada y es cuestionable su uso en datos con alta dimensionalidad. CURE es capaz de captar grupos de varias formas y tamaños (Guha, Rastogi et al. 1998). Tiene una alta complejidad, $O(n^2 \log n)$, y es sensible a varios parámetros de entrada. BIRCH y CURE manejan bien los puntos fuera de rango, BIRCH es menos complejo, pero obtiene un agrupamiento de peor calidad.

El algoritmo PDDP (Boley 1998) y su mejora sPDDP son de los más referenciados dentro de los que aplican técnicas jerárquicas divisivas (Berry 2004). Las salidas de ellos se utilizan como entrada del algoritmo medias. Otra variante concatenada aparece en (Arco, Bello et al. 2006d) donde se utiliza la salida del algoritmo estrella extendida (Gil-García, Badía-Contelles et al. 2003) para inicializar los algoritmos SKWIC y SKWIC borroso (Berry 2004), y así obtener mejor calidad del agrupamiento, sin requerir conocimiento previo del dominio. En (Cheng, Vempala et al. 2005, Cheng, Kannan et al. 2006) se presenta una nueva metodología que combina la estrategia divisiva y la aglomerativa.

Los métodos jerárquicos a partir de una representación de los objetos usando grafos han sido muy trabajados. Varios siguen estrategias divisivas, entre ellos uno de los más conocidos es el método basado en la construcción de un árbol de expansión mínimo (Zahn 1971). Otros enfoques ampliamente utilizados son el enlace simple (Gower and Ross 1969, Gotlieb and Kumar 1968) y el enlace completo (Backer and Hubert 1976). En (Gil-García, Badía-Contelles et al. 2006) se presenta un marco de trabajo para algoritmos jerárquicos aglomerativos basados en grafos. Otras propuestas que parten de una representación gráfica de los datos son: STIRR (Gibson, Kleinberg et al. 1998) que aplica métodos espectrales; la

variante aglomerativa de complejidad $O(nk+n \log n+k^2 \log k)$ que construye grafos de k-vecinos más cercanos, particiona y combina los grupos por su inter-conectividad y cercanía (Karypis, Han et al. 1999); los algoritmos jerárquicos utilizados por el sistema SUBDUE para agrupar datos estructurados o no (Jonyer, Cook et al. 2002) y el algoritmo jerárquico propuesto en (Jenssen, Hild et al. 2003) que descubre grupos de formas irregulares en datos de alta dimensionalidad, pero tiene complejidad cuadrática. Muchos tipos de datos que provienen de aplicaciones de la minería de datos pueden ser modelados como grafos bipartitos (por ejemplo, términos y documentos, consumidores y productos, revisores y filmes), algunas propuestas recientes en (Gao, Liu et al. 2005, Deodhar and Ghosh 2007).

Entre los métodos que explotan las propiedades estructurales de las conexiones entre los objetos a agrupar está la propuesta en (Radicchi, Castellano et al. 2004a), exitosa para grafos densos, donde se definen grupos fuertes y débiles considerando el grado de las conexiones internas en los grupos y las conexiones externas hacia otros grupos. Este método se basa en el coeficiente de agrupamiento local, con un costo temporal $O(m^4/n^2)$. El algoritmo SCAN (Xu, Yuruk et al. 2007), con complejidad $O(m)$, utiliza la vecindad de los nodos como un criterio de agrupamiento, así los nodos se agrupan considerando los vecinos que comparten. El algoritmo SMTIN permite minar datos espaciales pero requiere la especificación de un umbral de distancia y con un único umbral no es posible obtener grupos con múltiples resoluciones (Epter and Krishnamoorthy 1999). Esta desventaja se supera con la extensión propuesta en (Epter, Krishnamoorthy et al. 1999). Otras propuestas basadas en las propiedades estructurales se presentan en (Cortes, Pregibon et al. 2001, Aggarwal and Yu 2005, Wasserman and Faust 1994b)

Ninguna forma o técnica de agrupamiento es mejor que otra, pero sí, algunas son más apropiadas para ciertos problemas. El conocimiento del dominio puede en muchos casos ayudar a determinar qué tipo de grupo se va a formar y que tipo de agrupamiento se va utilizar con el objetivo de obtener los mejores resultados. Por ejemplo, muchos algoritmos de agrupamiento de documentos se han desarrollado, pero muy pocas propuestas toman en consideración las ventajas de las estructuras inherentes en el lenguaje. Varias investigaciones muestran que el lenguaje existe en una red small-world (Ferrer and Solé 2004, Ferrer and Solé

2001, Watts and Strogatz 1998), sin embargo, raras veces ha sido utilizado en el agrupamiento en dominios textuales (Chee and Schatz 2007). A partir de la motivación principal de este trabajo y considerando de gran utilidad las propiedades topológicas de los lenguajes y corpus textuales, se ha decidido focalizar en el estudio de las técnicas de agrupamiento que siguen una estrategia jerárquica y que trabajan sobre una representación gráfica de los datos, sobre todo para utilizar la estructura de las interrelaciones de los objetos en el proceso de agrupamiento.

Los métodos jerárquicos divisivos generalmente averiguan cuáles son los pares de nodos que están más débilmente conectados. Otro enfoque posible, cuando estos métodos operan sobre representaciones gráficas, consiste en averiguar cuáles son las aristas en el grafo que están más "entre" otros nodos, significando que la arista es, en algún sentido, la responsable de conectar muchos otros pares de nodos, aunque tales aristas no necesitan ser débiles en el sentido de similitud. La esencia es detectar cuáles son las aristas con mayor centralidad e ir eliminándolas en un proceso divisivo y así ir construyendo la jerarquía de grupos.

La centralidad es una propiedad estructural importante de los grafos (Bavelas 1948, Shaw 1954, Sabidussi 1966, Anthonisse 1971, Freeman 1977, Freeman 1979, Wasserman and Faust 1994a). No existe un consenso de qué es exactamente, sólo hay cierta conciliación sobre los procedimientos apropiados para su medición (Freeman 1979). Freeman identificó tres variantes de centralidad: el grado de un vértice como índice del potencial de comunicación, su cercanía al resto de los vértices del grafo y su mediación en los caminos de comunicación (Freeman 1979), centralidad como actividad, independencia y control, respectivamente. Por otra parte, Borgatti y Everett en (Borgatti and Everett 2005) proponen otra clasificación de las medidas de centralidad: medidas radiales, aquellas que evalúan los caminos que comienzan o terminan en un vértice dado y medidas mediales, aquellas que cuentan el número de caminos que pasan a través de un vértice o arista dados (por ejemplo, medidas betweenness). Clasificación que coincide con la presentada en (Latora and Marchiori 2004): estar cerca y colocarse o mediar entre otros (be near to and stand between others).

El grado de los vértices (Shaw 1954, Nieminen 1974, Scott 2000), el agrupamiento o transitividad local (Watts and Strogatz 1998, Newman 2001c) y closeness centrality (Sabidussi 1966, Freeman 1979) son también medidas de centralidad.

En la actualidad varias investigaciones reportan métodos de agrupamientos basados en la centralidad de nodos y aristas (Girvan and Newman 2002b, Newman 2003a, Wu, Garland et al. 2004, Clauset, Newman et al. 2004, Donetti and Muñoz 2004, Radicchi, Castellano et al. 2004b, White and Smyth 2005, Pinney and Westhead 2006). M. J. E. Newman es un autor prolífero en este tópico (Newman 2001a, Newman 2001b, Newman 2003b, Newman 2004a, Newman 2005, Newman 2006a, Newman 2006b, Newman 2004b).

La mayoría de los métodos trabajan con la medición de la centralidad de la arista contando el número de caminos más cortos que pasan a través de ella, llamadas medidas de la intermediación de los caminos más cortos (shortest-path betweenness) (Girvan and Newman 2002a). Estas medidas indican el potencial que tiene una arista para controlar el flujo de información en el grafo; así, favorecen a las aristas que se encuentran entre grupos y desfavorecen a aquellas incidentes a nodos de un mismo grupo. Intuitivamente, si una arista actúa en la interacción de muchos nodos su nivel de intermediación debe ser alto[16].

Uno de los primeros algoritmos jerárquicos divisivos usando la intermediación de los geodésicos se propuso (Girvan and Newman 2002a) bajo el supuesto: los caminos más cortos entre grupos viajan por el pequeño número de aristas que los comunican, produciendo en éstas alta intermediación. Observe en el Anexo 5 que este algoritmo sigue un proceso divisivo mediante la eliminación de las aristas con mayor intermediación. Este algoritmo se nombra GN debido a Girvan y Newman. Aunque es poco conocido en el área del agrupamiento de documentos, pues se desarrolló para analizar las redes complejas, se ha incluido en este estudio. Las aristas con altos valores de intermedicación producen un incremento de la distancia geodésica entre un gran número de pares de nodos cuando éstas son eliminadas del grafo (Wasserman and Faust 1994a, Girvan and Newman 2002b, Newman 2003a). Así, el

[16] Intuición original para la intermediación de nodos. Bavelas, A. (1948) A mathematical model for group structures. Human Organization, 7: 16-30.

recálculo de la intermediación después de cada eliminación es distintivo de estas propuestas, lo que significa que no hay una función que pueda ser definida para cada arista en el grafo inicial tal que la jerarquía resultante sea la representación del agrupamiento jerárquico llevado a cabo usando la función (Newman 2003a).

En la literatura existen otros algoritmos jerárquicos divisivos que utilizan la intermediación según los caminos geodésicos (Pinney and Westhead 2006, Wu, Garland et al. 2004, Donetti and Muñoz 2004, White and Smyth 2005, Rattigan, Maier et al. 2007, Newman 2005), pero en este trabajo de diploma no se abordan debido a que GN se considera un clásico para este tipo de algoritmos.

1.5 Validación del agrupamiento

"El agrupamiento es un proceso subjetivo; el mismo conjunto de datos usualmente necesita ser agrupado de formas diferentes dependiendo de las aplicaciones" (Jain, Murty et al. 1999). Esta subjetividad hace el agrupamiento difícil, más aún su validación.

Una forma de evaluación del agrupamiento muy sencilla, puede ser, por ejemplo, mediante la visualización del conjunto de datos cuando éste es pequeño y los datos son bidimensionales. Sin embargo, esta forma de evaluación puede ser extremadamente difícil al intentar realizar una visualización efectiva de un conjunto de datos de alta dimensionalidad (Hansen and Johnson 2005). ¿Qué hacer cuando los datos no pueden representarse gráficamente, o es muy difícil o algunas veces imposible para un observador humano valorar el agrupamiento de los mismos; o no existe una forma simple de decidir si el resultado de un agrupamiento se ajusta a la división que deseamos de los datos?

Por otra parte, muchos algoritmos de agrupamiento varían sus resultados dependiendo de las características de los datos (por ejemplo, geometría y densidad de distribución) (Halkidi, Batistakis et al. 2001b). Otros dependen fuertemente de los valores asignados a los parámetros. Por ejemplo, si hay un parámetro que controle la resolución a la cual los datos son vistos, el algoritmo produce una jerarquía en función de ese parámetro. En este caso es necesario decidir cuál nivel de la jerarquía refleja mejor los grupos según propiedades que se desea que el agrupamiento satisfaga. Algunos algoritmos necesitan que se especifique

inicialmente el número de grupos a obtener, otros requieren que se especifique el número de vecinos de cada punto como un parámetro externo. Así, los resultados producidos son en función de los parámetros fijados y se hace necesario verificar cuáles se ajustan a los datos (Levine and Domany 2001).

Variaciones a partir de características de los datos, diferentes técnicas de análisis de grupos y definición de parámetros para el algoritmo a aplicar, indican que una evaluación de los resultados es necesaria para medir la calidad del agrupamiento. Una práctica común, en tal sentido, es aplicar medidas de validación de grupos (Stein, Eissen et al. 2003).

El procedimiento de evaluar los resultados de algoritmos de agrupamiento se conoce por validación del agrupamiento (Theodoridis and Koutroubas 1999, Halkidi, Batistakis et al. 2002). Se dice medida de validación de grupos a una función que hace corresponder a un agrupamiento un número real, indicando en qué grado el agrupamiento es correcto o no (Höppner, Klawonn et al. 1999). Estas medidas en su mayoría son heurísticas por naturaleza para conservar una complejidad computacional plausible.

1.5.1 Clasificación de las medidas

Las medidas de evaluación del agrupamiento se clasifican en: globales y locales, subjetivas y objetivas, internas, externas y relativas, y supervisadas y no supervisadas (Höppner, Klawonn et al. 1999, Silberschatz and Tuzhilin 1996, Kaufman and Rousseeuw 1990).

Las medidas globales describen la calidad del resultado completo de un agrupamiento usando un único valor real, mientras que las locales evalúan cada grupo obtenido (Höppner, Klawonn et al. 1999). Las medidas objetivas miden propiedades estructurales de los resultados de los agrupamientos, por ejemplo, la separación entre los grupos y la compactación o densidad de los mismos (Halkidi, Batistakis et al. 2001b). La presencia de tales propiedades no garantiza que los resultados sean interesantes para el usuario, estas medidas carecen del enlace con los usuarios, aunque su principal atractivo es que son independientes del dominio (Silberschatz and Tuzhilin 1996). Las medidas subjetivas evalúan considerando la usabilidad de los grupos (Stein, Eissen et al. 2003). Las investigaciones en medidas subjetivas han sido menos intensas que las realizadas en las objetivas (Tuzhilin 2002).

Una clasificación muy usada divide la validación del agrupamiento en: medidas internas, externas y relativas (Theodoridis and Koutroubas 1999, Kaufman and Rousseeuw 1990). Estas últimas tienen un alto costo computacional (Halkidi, Batistakis et al. 2001a). Otra división consiste en medidas supervisadas y no supervisadas, refiriéndose a externas e internas, respectivamente. Las medidas externas se basan en un criterio externo que es impuesto sobre los datos, por ejemplo, una estructura previamente especificada que refleje la intuición que se tenga del agrupamiento de los datos. No es posible aplicar estas medidas a situaciones del mundo real donde usualmente no está disponible una clasificación de referencia. Las medidas internas evalúan considerando solamente los resultados del agrupamiento en términos de cantidades que involucran los vectores de datos. Las medidas relativas se basan en la comparación del agrupamiento a evaluar con otros esquemas de agrupamiento o con resultados del mismo algoritmo con diferentes valores en los parámetros.

A continuación se mencionarán las principales medidas internas reportadas en la literatura para la evaluación de jerarquías y particiones duras. Las medidas externas no serán abordadas porque en la aplicación que se desarrolla no existen clasificaciones de referencia.

1.5.2 Medidas internas

Los algoritmos de agrupamiento generan una estructura espacial y se pueden definir medidas para diferentes aspectos de esta estructura, por ejemplo, densidad (Brun, Sima et al. 2007). Existen varios trabajos encaminados al desarrollo de medidas que validan el agrupamiento de una manera no supervisada. Algunos antiguos como el índice Goodman-Kruskal que tiene una alta complejidad computacional (Goodman and Kruskal 1954), el índice C apropiado cuando los grupos tienen tamaños similares (Hubert and Schultz 1976), los índices propuestos en (Akaike 1974, Schwartz 1978) utilizan criterios de información, seguidos por los propuestos en (Jain and Dubes 1988, Bock 1985). El cálculo de la dispersión intragrupo y la separación entre los grupos ha sido ampliamente trabajado, un ejemplo es el índice Calinski-Harabasz (Calinski and Arabas 1974), utilizado recientemente en (Maulik and Bandyopadhyay 2002).

Los índices para evaluar particiones generalmente se basan en alguna motivación geométrica para estimar cuán compactos y bien separados están los grupos. Un ejemplo son los índices

Dunn (Dunn 1974) y sus generalizaciones (Bezdek and Pal 1995). La medida Davies-Bouldin es basada en la idea que una buena partición es aquella con gran separación entre grupos y alta homogeneidad y compactación dentro de cada grupo (Davies and Bouldin 1979). En (Pal and Biswas 1997) se generalizan los índices Dunn y Davies-Bouldin utilizando las estructuras de grafos. El índice I también sigue un esquema general similar a los índices Dunn, pero utiliza la distancia máxima entre grupos y adiciona las distancias (en lugar de promediarlas) multiplicadas por el número de grupos (Maulik and Bandyopadhyay 2002).

Otras variantes de índices, en su mayoría con un alto costo computacional especialmente cuando el número de grupos y objetos es muy grande (Xie and Beni 1991), se han propuesto en (Dave 1996, Milligan and Cooper 1985). Una medida interna y global es el índice de separación (Höppner, Klawonn et al. 1999). En (Halkidi, Vazirgiannis et al. 2000) se presenta el índice de validación SD que suma el promedio de compactación de los grupos y la separación total entre ellos. En (Brun, Sima et al. 2007) hacen referencia al índice Silueta que es el promedio, sobre todos los grupos, del ancho de la silueta de sus puntos.

Como se ha podido apreciar, existen múltiples índices que permiten la validación no supervisada de los agrupamientos. En este trabajo se particulariza en la medida Overall Similarity y modularidad. La primera, porque permite valorar los grupos localmente. La segunda porque permite una valoración global de los grupos y además es la más utilizada para evaluar algoritmos de detección de comunidades en redes, tal es el caso del algoritmo GN.

La cohesión de los grupos se puede usar como una medida de validación de éstos. Overall similarity es un índice interno que se ha utilizado para medir la cohesión basándose en la similitud de los pares de objetos en un grupo (Steinbach, Karypis et al. 2000).

$$OverallSimilarity(Grupo) = \frac{1}{|Grupo|^2} \sum_{O_i, O_j \in Grupo} \text{distancia}(O_i, O_j) \qquad \textbf{(1.5.1)}$$

La modularidad, utilizada para evaluar agrupamientos jerárquicos, mide la fortaleza de los grupos encontrados analizando las interconexiones antes y después del agrupamiento realizado (Newman 2003a, Newman and Girvan 2004).

$$Q = \sum_i \left(e_{ii} - a_i^2 \right) = \mathbf{Tr}\ \mathbf{e} - \left\| \mathbf{e}^2 \right\| \qquad (1.5.2)$$

Donde **e** es una matriz simétrica de orden *k* cuyo elemento e_{ij} es la fracción de todas las aristas en el grafo que conectan nodos del grupo *i* con nodos del grupo *j*, $\|\mathbf{e}\|$ indica la suma de los elementos de la matriz **e** y $\mathbf{Tr}\ \mathbf{e} = \sum_i e_{ii}$ es la traza de la matriz que da la fracción de aristas en en el grafo que conectan nodos en el mismo grupo.

1.5.3 Uso de los conjuntos aproximados en la validación

Los conjuntos aproximados consideran que a todo objeto *x* de un universo *U* está asociada una cierta cantidad de información, expresada por medio de algunos atributos que describen el objeto (Komorowski, Pawlak et al. 1999, Bazan, Nguyen et al. 2004). La estructura de información básica de esta teoría es el sistema de información; par (*U*, *A*) donde $A=\{a_1, a_2, \ldots, a_m\}$ es el conjunto de atributos y *U* es un conjunto no vacío llamado universo de objetos descritos usando los atributos a_i (Komorowski, Pawlak et al. 1999)[17].

Cualquier subconjunto *X* (concepto) del universo *U* se puede expresar en términos de estos bloques de forma exacta o aproximada. La vaguedad es una propiedad de los conceptos y puede ser atribuida a los límites del conjunto, mientras que la incertidumbre es una propiedad de los elementos del concepto y tiene que ver con la pertenencia o no a éste (Pawlak, Grzymala-Busse et al. 1995). Cuando un concepto es vago, los elementos del universo no pueden ser identificados con certeza como elementos del concepto.

Algunas extensiones de la teoría clásica de los conjuntos aproximados no requieren que se cumpla la transitividad ni la simetría, tales como las relaciones llamadas de tolerancia o similitud. La extensión de RST clásico a relaciones de similitud R'_B acepta que objetos que no

[17] Esta definición es independiente a la definición de sistema de información de Shannon

son inseparables pero sí suficientemente cercanos o similares puedan pertenecer a la misma clase (Slowinski and Vanderpooten 1997). Varias medidas de similitud entre objetos o de comparación de atributos pueden ser utilizadas, obsérvese el Anexo 3.

Dos conceptos básicos fueron introducidos a partir de las relaciones de inseparabilidad: aproximaciones inferiores ($R'_*(X)$) y superiores ($R'^*(X)$) de un concepto X ($X \subseteq U$). Observe en las expresiones (1.5.3) y (1.5.4) sus cálculos a partir de relaciones de similitud.

$$R'_*(X) = \{x \in X : R'(x) \subseteq X\} \tag{1.5.3}$$

$$R'^*(X) = \bigcup_{x \in X} R'(x) \tag{1.5.4}$$

Aplicar RST a la evaluación del agrupamiento permite realizar una validación no supervisada, poco costosa computacionalmente y el cálculo común inicial de las relaciones y aproximaciones inferiores y superiores puede ser reutilizado por varias medidas de calidad, inclusión y proximidad de conceptos y el sistema en general.

Objetos descritos por rasgos constituyen un sistema de información y adicionalmente se trabaja con conceptos definidos sobre ese sistema de información. Cada concepto X_i se corresponde con un grupo resultante de un proceso de agrupamiento al cual dichos objetos fueron sometidos. Obsérvese la Tabla 1.5.1. Sólo se considera en esta investigación resultados de agrupamientos duros y deterministas, donde los conceptos forman una partición. No obstante, la teoría puede ser aplicada a la evaluación de cubrimientos.

Tabla 1.5.1 Sistema de información.

	Rasgo_1	Rasgo_2	…	Rasgo_m
Objeto_1	Valor_{11}	Valor_{12}	…	Valor_{1m}
Objeto_2	Valor_{21}	Valor_{22}	…	Valor_{2m}
…	…	…	…	…
Objeto_n	Valor_{n1}	Valor_{n2}	…	Valor_{nm}

Así, es posible calcular para cada objeto agrupado las relaciones de similitud siguiendo la expresión (1.5.5), donde $s(x, y)$ retorna un valor de similitud entre los objetos x e y, y ξ es el

umbral de similitud que será considerado. La forma de medir la similitud y qué umbral utilizar para formar los conjuntos de relaciones depende del dominio donde fue aplicado el agrupamiento, cómo fueron descritos los objetos y qué nivel de granularidad se desea evaluar en los resultados. Obsérvese en el Anexo 4 posibles variantes para el cálculo del umbral. Adicionalmente, se pueden calcular las aproximaciones inferiores y superiores de cada grupo (concepto) usando (1.5.3) y (1.5.4), respectivamente.

$$R'(x)=\{y\in U : yR'x, \text{ es decir } y \text{ está relacionado con } x \text{ si y sólo si } s(x,y)>\xi\} \qquad (1.5.5)$$

A partir del cálculo de las aproximaciones inferiores y superiores por grupos, se propone validar el agrupamiento y cada grupo mediante la aplicación de medidas ofrecidas por RST para evaluar los conceptos definidos sobre sistemas de información. Éstas permiten tener una noción de la proximidad de los conceptos y pueden ser aplicadas en varios esquemas de razonamiento (Zhong, Skowron et al. 1999). Trabajos previos de este enfoque se presentan en (Arco, Bello et al. 2006a, Arco, Bello et al. 2006c, Arco, Bello et al. 2006b).

Las medidas la calidad y precisión generalizadas del agrupamiento, expresiones (1.5.6) y (1.5.7), respectivamente, permiten validar globalmente resultados del agrupamiento. El peso asociado a un grupo X_i se representa por w_i, cumpliéndose las restricciones $w_i \geq 0$ y $\sum_{i=1}^{l} w_i = 1$ (Arco, Bello et al. 2006b, Caballero, Arco et al. 2007b).

$$\Gamma_G(DS) = \frac{\sum_{i=1}^{l} \left(\left| R'_*(X_i) \right| \cdot w_i \right)}{|U|} \qquad (1.5.6)$$

$$\mathrm{A}_G(DS) = \frac{\sum_{i=1}^{l} \left(\left| R'_*(X_i) \right| \cdot w_i \right)}{\sum_{i=1}^{l} \left(\left| R'^*(X_i) \right| \cdot w_i \right)} \qquad (1.5.7)$$

Varios criterios pueden ser empleados para ponderar los grupos y así captar mejor propiedades deseadas.

Una forma de medir la pertenencia de un objeto a un grupo es la función de pertenencia aproximada. La media de la pertenencia aproximada de los objetos a cada grupo puede también ser empleada para ponderar los grupos (Arco, Bello et al. 2006b, Caballero, Arco et al. 2007a, Caballero, Arco et al. 2007b, Caballero 2007). Observe las expresiones (1.5.8) y (1.5.9) para calcular la pertenencia aproximada y ponderar los grupos, respectivamente. Esta forma de cálculo fue propuesta por investigadores del Laboratorio de Inteligencia Artificial del Centro de Estudios de Informática.

$$\varpi_X(x) = \frac{|X \cap R'(x)|}{|X \cup R'(x)|} \tag{1.5.8}$$

$$w_i = \frac{\sum_{x \in X_i} \varpi_{X_i}(x)}{|X_i|} \tag{1.5.9}$$

La media de la pertenencia aproximada de los objetos por grupos puede utilizarse como una forma de validación local, mientras que tanto la precisión y calidad generalizada, así como la media armónica de estas dos últimas pueden utilizarse como validación global del agrupamiento.

1.6 Etiquetamiento de los grupos

En muchos casos existe un divorcio entre los resultados del agrupamiento de un conjunto de datos y los requerimientos de los usuarios de recuperar etiquetas de los grupos obtenidos, por lo que es necesaria una etapa de post-agrupamiento usualmente nombrada etiquetamiento (Chen and Liu 2004). Las etiquetas a obtener deben ser únicas, sintetizadoras, expresivas, tener poder discriminante, contiguas, poseer consistencia jerárquica, y no ser redundantes (Stein and Eissen 2004). Algunos de los mayores retos del post-agrupamiento son etiquetar grupos con formas irregulares, distinguir puntos fuera de rango y extender los límites de los grupos (Chen and Liu 2004).

Algunos trabajos han sido desarrollados en esta área (Chen and Liu 2004, Hasegawa, Sekine et al. 2004, Popescul and Ungar 2000, Jickels and Kondrak 2006, Treeratpituk and Callan 2006, Glover, Pennock et al. 2002, Pantel and Ravichandran 2004, Cutting, Karger et al. 1993,

Skupin and Jongh 2005, Stein and Eissen 2004, Guha, Rastogi et al. 1998, Zhang, Ramakrishnan et al. 1996); sin embargo, etiquetar es usualmente ignorado por los investigadores en agrupamiento y se le ha prestado menos atención a crear buenos descriptores de grupos, una de las razones es que el problema del agrupamiento no ha sido aún resuelto. Emergen más y más algoritmos de agrupamiento efectivos y que típicamente no tienen una complejidad lineal, entonces la etapa de etiquetamiento se hace crítica para grandes conjuntos de datos, especialmente cuando los puntos en las regiones fronteras de los grupos son significativos para las aplicaciones (Chen and Liu 2004).

Es deseado que el proceso de etiquetamiento sea no supervisado, de forma tal que se puedan producir etiquetas rápidamente sin el alto costo de la intervención humana en la anotación de conjuntos de entrenamiento (Jickels and Kondrak 2006).

En este trabajo se utiliza RST para etiquetar mediante la extracción de los objetos más representativos de cada grupo. La idea del uso de RST para etiquetar es reemplazar el concepto vago sobre representatividad por el concepto llamado aproximación inferior, ya que la aproximación inferior de un grupo incluye todos aquellos objetos que con certeza pertenecen al grupo, por tanto esos objetos son los más representativos del grupo (Arco, Bello et al. 2006b). Además, es posible controlar las regiones límites de cada grupo, al considerar un umbral para construir las relaciones de similitud. Por tanto, es posible regular la granularidad del conjunto de objetos más representativos de cada grupo. Así, una de las principales ventajas de etiquetar considerando las aproximaciones inferiores a partir de relaciones de similitud usando un umbral es que los usuarios pueden especificar el tamaño del conjunto de los objetos más representativos de cada grupo, conjunto que no tienen que girar necesariamente alrededor de un centro o tener una forma esférica. Además, no se requieren cálculos adicionales si previamente fue utilizada RST para validar el agrupamiento.

1.7 Consideraciones finales del capítulo

La revisión bibliográfica realizada sugiere el uso de métodos de agrupamiento y sus formas de valoración, mediante la validación y etiquetamiento de los grupos, para mejorar resultados de procesos de recuperación de información. Por un lado, los métodos basados en el cálculo de la

intermediación de las aristas parecen promisorios para trabajar en dominios textuales. Por otro, la validación no supervisada (usando medidas internas) es extremadamente útil para valorar resultados de agrupamientos de documentos, por no requerir información adicional del dominio.

2 SISTEMA PARA LA GESTIÓN DE ARTÍCULOS CIENTÍFICOS

El Sistema para la Gestión de Artículos Científicos recuperados a partir de Lucene (GARLucene) puede asistir al usuario al enfrentarse a una colección de artículos científicos mediante la indexación, recuperación, agrupamiento, evaluación y obtención de los documentos más representativos de los grupos. Estos procesos se realizan de forma automática, totalmente no supervisada y sin requerir conocimiento a priori del dominio de aplicación.

GARLucene permite indexar los artículos científicos para su posterior recuperación a partir de palabras claves o frases. Además, con los artículos recuperados forma grupos jerárquicos divisivos mostrándolos al usuario en una vista de árbol comenzando por el grupo raíz. Cada grupo es evaluado según el resultado de medidas de validación permitiéndole al usuario conocer el grado de certeza del agrupamiento obtenido mediante la evaluación de la consistencia de los grupos, de la similitud global de los grupos y la modularidad de los mismos. Además marca los documentos más representativos en los grupos. El sistema brinda la posibilidad de que el usuario mediante la expansión y contracción de los grupos evalúe cortes del árbol, de esta forma el usuario puede conocer la calidad de los grupos formados para un corte dado en la jerarquía o que el usuario pueda ver algún artículo científico en especifico de los grupos formados.

En este capítulo se presenta el diseño general de GARLucene así como la especificación de cada uno de sus módulos principales. GARLucene utiliza varios elementos del marco de trabajo LIUS y la biblioteca Lucene, por tanto, una descripción de los mismos es realizada. Además de una explicación detallada de aquellos módulos que permiten ser reutilizados es presentada, así como los principales conceptos de la programación orientada a objetos (POO), utilizados en el diseño e implementación del sistema.

2.1 LIUS y Lucene

LIUS y Lucene soportan toda el proceso de indexado de los distintos tipos de documentos así como la búsqueda dentro de éstos.

Lucene es una novedosa herramienta que permite tanto la indexación cómo la búsqueda de documentos. Creada bajo una metodología orientada a objetos e implementada completamente en Java, no se trata de una aplicación que pueda ser descargada, instalada y ejecutada sino de una interfaz de programación de aplicaciones (Application Program Interfaces; API) flexible, muy potente y realmente fácil de utilizar, a través de la cual se pueden añadir, con pocos esfuerzos de programación, capacidades de indexación y búsqueda a cualquier sistema que se esté desarrollando (Lucene).

Originalmente escrita por Doug Cutting, en Septiembre de 2001 pasó a formar parte de la familia de código abierto de la fundación Jakarta. Desde entonces, debido a su mayor disponibilidad, ha atraído a un gran número de desarrolladores, incluso empresas como Hewlett Packard, FedEx. Usan, o al menos lo han evaluado.

A continuación se detallan algunas características de Lucene:

- ✓ Multiplataforma
- ✓ Alto rendimiento, escalable
    - Unos 20MB/minuto en Pentium M 1.5GHz
    - Bajo consumo de memoria, (solo 1MB heap)
    - Tamaño de índices aproximado 20-30% del tamaño del texto indexado
- ✓ Indexación incremental e indexación por lotes

    El término de indexación por lotes se utiliza para referirse a aquellos procesos de indexación, en los cuales, una vez que ha sido creado el índice para un conjunto de documentos, es difícil añadirle nuevos documento. Por eso, en esta forma de indexado se opta por reindexar todos los documentos de nuevo cuando es necesario hacer una actualización. Sin embargo, en la indexación incremental se pueden añadir documentos

a un índice ya creado con anterioridad de forma fácil. Lucene soporta ambos tipos de indexación.

- ✓ Origen de datos

  Muchas herramientas de indexación sólo permiten indexar ficheros o páginas web, lo que supone un serio inconveniente cuando se tiene que indexar contenido almacenado en una base de datos. Lucene permite indexar tanto documentos y páginas web como el contenido procedente de una base de datos.

- ✓ Contenido etiquetado

  Algunas herramientas, tratan los documentos como simples flujos de palabras. Pero otras, como Lucene, permiten dividir el contenido de los documentos en campos y así poder realizar consultas con un mayor contenido semántico. Así, se pueden buscar términos en los distintos campos del documento concediéndole más importancia según el campo en el que aparezca. Por ejemplo, si se dividen los documentos en dos campos, título y contenido, puede concederse mayor importancia a aquellos documentos que contengan los términos de la búsqueda en el campo título.

- ✓ Técnica de indexación

  Existen palabras tales como *a, unos, el, la* que añaden poco significado al índice, son palabras poco representativas del documento y se llaman palabras de parada o gramaticales (stop-words). Al eliminar estas palabras del índice se reduce considerablemente el tamaño del mismo así como el tiempo de indexación. Estas palabras están contenidas en lo que se denomina lista de parada, que es la técnica de indexación contemplada por Lucene.

- ✓ Concurrencia

  Lucene gestiona que varios usuarios puedan buscar en el índice de forma simultánea así como también que un usuario modifique el índice al mismo tiempo que otro lo consulta.

- ✓ Elección del idioma

Tal y como ya se indicó con anterioridad Lucene trabaja con listas de parada, las cuales son proporcionadas por el desarrollador que está utilizando Lucene, esto permite escoger el idioma a utilizar.

LIUS es un marco de trabajo (framework) para la indexación y búsqueda de documentos basado en Lucene, agrega una fina capa por encima de Lucene para organizar el trabajo de extraer datos de distintos tipos de documentos y mapear esos datos a campos de Lucene. Además, el marco de trabajo permite manipular mediante archivos de configuración en formato XML qué campos se crean, de qué tipo y qué analizadores se deben usar, entre otros. LIUS ofrece más comodidad a la hora de configurar Lucene y decidir qué, cómo y dónde se indexa.

LIUS se compone de una biblioteca propia y de otras de terceros que se usan para extraer textos de distintos tipos de documentos. Es capaz de indexar los tipos de documentos siguientes (LIUS):

- ✓ DOC (MS Word 6.0/96/97/2000/XP): usando la biblioteca Textmining de Apache (http://www.textmining.org).
- ✓ XLS (MS Excel 95/97/2000/XP/2003): usando la biblioteca JExcelAPI (http://jexcelapi.sourceforge.net/).
- ✓ PPT (MS PowerPoint 95/97/2000/XP/2003): usando la biblioteca POI de Apache (http://poi.apache.org).
- ✓ RTF: usando las rutinas de Java de manipulación de Rich Text Format (javax.swing.text.rtf).
- ✓ PDF: usando la biblioteca PDFBox (http://www.pdfbox.org).
- ✓ XML: usando el analizador sintáctico XML de Java (org.w3c.dom).
- ✓ HTML: usando las bibliotecas Jtidy (http://jtidy.sourceforge.net) y NekoHTML (http://people.apache.org/~andyc/neko/doc/html).
- ✓ TXT: usando un analizador sintáctico propio.

- ✓ SXW/ODT, SXC/ODS, SXI/ODP (OpenOffice 1/2): usando el analizador sintáctico de XML de Java (org.w3c.dom).
- ✓ ZIP: usando el procesador de ZIP de Java (java.util.zip).
- ✓ MP3: usando las bibliotecas de audio de Java (javax.sound.sampled).
- ✓ VCF (VCard): usando un analizador sintáctico propio.
- ✓ Latex: usando un analizador sintáctico propio.
- ✓ Java Beans: usando reflexión.

En el sistema desarrollado en este trabajo se permite sólo un subconjunto de estos tipos de documentos. Observar el capítulo 3.

LIUS se basa en ficheros XML de configuración que permiten describir como se debe configurar Lucene para indexar y/o buscar en los documentos. Lo usual es tener un solo fichero XML, pero se pueden usar más, por ejemplo, para definir distintas formas de indexar o buscar. En el fichero de configuración se puede especificar:

- ✓ El analizador que debe usar Lucene.
- ✓ Los parámetros de creación del índice.
- ✓ Los datos que se deben extraer de cada tipo de documento y a qué campos de Lucene se deben hacer corresponder. Por ejemplo, se puede especificar que el contenido de los PDF sea incorporado en el campo content de Lucene, el título en otro llamado title, y el autor en el campo writer.
- ✓ Cómo se deben configurar los campos de Lucene en los que se introducen los datos. Por ejemplo, se tienen los tipos de campos siguientes:
    - o Text: campo de texto almacenado y tokenizado.
    - o TextReader: campo de tipo stream.
    - o Keyword: campo de texto almacenado y no tokenizado.
    - o concatDate: campo tipo fecha almacenado y no tokenizado.

- UnIndexed: campo de texto almacenado y no indexado.
- UnStored: campo de texto no almacenado y tokenizado.
- Qué peso (boost) se debe aplicar a cada campo y al documento entero.

✓ Qué tipo de búsqueda se puede emplear.

- queryTerm: búsqueda por un solo campo.
- rangeQuery: búsqueda de rango por un solo campo.
- queryParser: usar el analizar sintáctico de consultas de Lucene sobre un solo campo.
- multiFieldQueryParser: usar el analizador sintáctico de consultas de Lucene sobre varios campos.

✓ Qué campos debe devolver la búsqueda y como formatearlos (si se deben fragmentar o no y si se deben resaltar los términos buscados).

2.2 Diseño general de GARLucene

En este epígrafe se mencionará inicialmente los principales conceptos de la programación orientada objetos y del Lenguaje Unificado de Modelación (Unified Modeling Language; UML) que permitirá describir posteriormente el diseño general de GARLucene. Se particularizará en los módulos del sistema que permiten preprocesamiento textual, recuperación, agrupamiento y valoración de grupos. Se describirán cada uno de los algoritmos, así como la forma en que fueron implementados en el sistema.

2.2.1 La programación orientada a objetos (POO)

Es importante tratar algunos conceptos fundamentales de la Programación Orientada a Objetos (POO), el uso de interfaces, así como aspectos sobre la metodología de análisis y diseño orientada a objetos UML utilizada durante las etapas de desarrollo del sistema (Rumbaugh, Booch et al. 1997a, Rumbaugh, Booch et al. 1997b).

2.2.1.1 Conceptos generales

El sistema de diseñó y desarrolló utilizando el paradigma de programación orientado a objetos. Es por eso que es necesario precisar cuáles son las características particulares y propiedades que identifican los objetos que son utilizados en este diseño. En (Rodríguez 1999) se presentan las siguientes propiedades de los objetos:

Estado: Se define a partir de los valores que en un momento dado tienen los atributos del objeto. La estructura del objeto se define como el conjunto de todos los atributos o propiedades. Además un objeto puede conocer o contener a otros objetos, estas relaciones son también parte de su estado.

Comportamiento: Define cómo actúan los objetos frente a estímulos externos en términos de cambio de estados.

Identidad: Esta es la propiedad de un objeto que lo distingue del conjunto de todos los demás objetos del universo al que pertenece. Los modelos de POO son representaciones abstractas de este tipo.

El protocolo de objeto define la envoltura del comportamiento admitido por el objeto, representa todas las vistas estáticas y dinámicas del objeto. Para la mayoría de las abstracciones no triviales, es útil dividir los protocolos en grupos lógicos de comportamiento. Las colecciones que dividen el espacio del comportamiento de un objeto denotan los roles que un objeto puede jugar. Un role es una máscara con la cual se presenta y define un contrato entre la abstracción y sus clientes.

El marco de referencia conceptual en un sistema orientado a objeto es el modelo de objetos que incluye cuatro conceptos fundamentales que son: la abstracción, el encapsulamiento, la modularidad y la jerarquía. También existen otros conceptos secundarios dignos a tener en cuenta que son: los tipos, la concurrencia y la persistencia.

Abstracción: Denota las características esenciales de un objeto que lo distinguen de todos los demás tipos de objeto y proporciona así fronteras conceptuales nítidamente definidas respecto a la perspectiva del observador.

Encapsulamiento: Es uno de los principios más importantes de la POO, ha permitido la reusabilidad de objetos. Constituye el proceso de almacenar en un mismo compartimiento los elementos de una abstracción que constituyen su estructura y su comportamiento. El cliente se interesa por lo que hace el objeto y no cómo lo hace.

Modularidad: Es la propiedad que tiene un sistema que ha sido descompuesto en un conjunto de módulos cohesivos y débilmente acoplados.

Jerarquía: Es una clasificación u ordenamiento de abstracciones.

Concurrencia: Es la propiedad que distingue un objeto activo de uno que no está activo.

Persistencia: Es la propiedad de un objeto por la que su existencia trasciende el tiempo, el espacio, o ambos.

En (Rodríguez 1999) se definen los conceptos de clase y tipo. La clase no es más que una representación abstracta que define la estructura y el comportamiento que le son comunes a un grupo de objetos. Mientras que el tipo es un protocolo usado en los mecanismos de comunicación e interacción entre objetos. Tiene identidad y generalmente está más relacionado a los mecanismos de comunicación que a la propia naturaleza de los objetos.

2.2.1.2 Interfaces

El uso de las interfaces en la programación orientada a objetos es una tendencia actual. Ellas esclarecen el diseño, lo hacen más cercano a la realidad y facilitan la implementación. En GARLucene ha sido muy efectivo en el diseño el uso de interfaces.

Cuando un objeto "implementa una interfaz" ese objeto implementa cada función miembro de la interfaz. Los objetos pueden, por supuesto, soportar simultáneamente múltiples interfaces. Las interfaces son inmutables, debido a que nunca pueden ser versionadas por lo que esto elimina los problemas de versionamiento. Una nueva versión de una interfaz, creada por la adición o eliminación de funciones, o cambios semánticos, es enteramente una nueva interfaz a la cual se le asigna un nuevo identificador (ID) único. Por lo que la nueva interfaz no va a crear "conflictos" con la vieja.

La funcionalidad de encapsular los objetos accedidos a través de interfaces hace al sistema abierto y extensible. Es abierto en el sentido de que cualquiera puede proveer una implementación de una interfaz definida y cualquiera puede desarrollar una aplicación que use dichas interfaces y es extensible en el sentido de que interfaces nuevas, o extendidas puedan ser definidas sin cambiar las aplicaciones existentes y estas aplicaciones entienden las nuevas interfaces y las explotan mientras continúan interoperando con las viejas aplicaciones a través de las interfaces viejas.

2.2.2 Generalidades del sistema

GARLucene es un sistema manipulador de documentos basado en módulos que se encargan de su funcionalidad. Observar Figura 2.2.1.

1. Indexado
2. Recuperación
3. Representación textual
4. Representación de la colección de documentos recuperados en un grafo
5. Agrupamiento
6. Validación del agrupamiento

El procesamiento textual se realiza a partir del resultado del proceso de recuperación de la información asistido por LIUS y Lucene. Éstos soportan el desarrollo de los módulos uno y dos de la aplicación permitiendo que se realicen los procesos de indexado y recuperación de documentos.

GARLucene reutiliza algunas facilidades de Lucene para realizar la representación textual, correspondiente el tercer módulo de la aplicación.

En el tercer módulo la representación textual se realiza utilizando una representación espacio-vectorial (Vector Space Model; VSM) (Salton, Wong et al. 1975); la cual pasa por varios procesos de refinamiento con el fin de obtener una mejor representación.

La colección de documentos recuperados se representa mediante un grafo donde cada documento es un nodo y las aristas entre los nodos se ponderan con el valor de la similitud coseno que existe entre ellos. Esencialmente el cuarto módulo se encarga de la creación y manipulación de este modelo abstracto.

El módulo de agrupamiento consta de tres algoritmos FastClustering, GN Betweenness, GN Betweenness & Similarity, los cuales forman colecciones de grupos a partir de la eliminación sucesiva de diferentes aristas del grafo. Estos algoritmos son jerárquicos divisivos, generando una jerárquica de grupos.

El último módulo se encarga de la validación desde dos ópticas diferentes:

- ✓ Validación local de los grupos

  Brinda una idea de la calidad del grupo utilizando las medidas Overall Similarity y la media de la pertenencia aproximada (Mean of Rough Membership), esta última basada en RST.

  Como parte de esta sección se obtienen los documentos más representativos utilizando la aproximación inferior basada en RST.

- ✓ Evaluación de colecciones de grupos

  Brinda una idea de la calidad de la colección de grupos que se tiene utilizando modularidad (Modularity), precisión aproximada (Rough Accuracy), calidad aproximada (Rough Quality) y Rugh F-measure. Estas tres últimas basadas en RST.

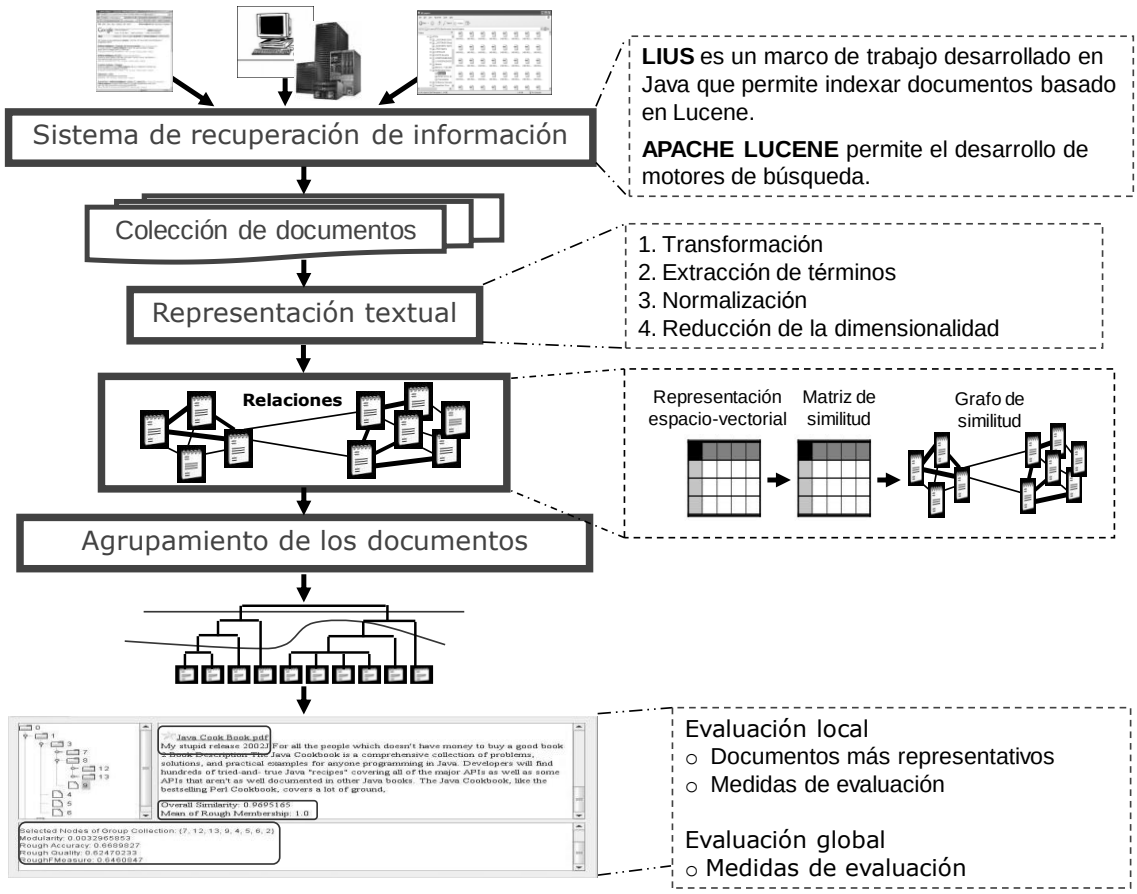

Figura 2.2.1. Generalidades del sistema

2.3 Biblioteca independiente cluster.jar

Esta biblioteca se creó con el propósito de incorporar diferentes algoritmos que se puedan reutilizar de forma independiente del dominio de aplicación. Todos estos algoritmos trabajan a través de interfaces, las cuales sólo hay que implementar para poder crear una instancia de un algoritmo y ejecutarlo. Estas interfaces suponen, al igual que el módulo de representación gráfica, la representación de la colección de documentos recuperados en un grafo donde cada documento es un nodo y las aristas entre los nodos están ponderadas con el valor de la similitud coseno que existe entre ellos. A continuación se describen las principales interfaces y clases que existen en la biblioteca.

2.3.1 Interfaces

Esta biblioteca contiene varias interfaces, a continuación se describen aquellas que pudieran ser necesarias implementar cuando sea preciso reutilizar alguno de los algoritmos que incluye.

- ✓ iGroup

  Grupo formado por nodos, devuelve su número de identificación, el de su padre, la cantidad de objetos que contiene, así como un objeto específico de su colección.

- ✓ iGroups

  Colección de grupos, devuelve un grupo específico de su colección o se puede conocer la cantidad de grupos que contiene.

- ✓ iRelation

  Relaciones entre los objetos, se le pasa un objeto y devuelve todos los objetos que estén relacionados con él.

- ✓ iConnection

  Conexiones entre los objetos, devuelve la cantidad de conexiones que existen en el universo o si entre dos objetos existe una conexión.

- ✓ iEdges

  Aristas entre los objetos de un grupo, devuelve el valor que existe entre la conexión de dos objetos o borra dicha conexión.

Estas son las interfaces generales que contiene la biblioteca y son implementadas en el sistema.

2.3.2 Algoritmos de agrupamiento

La biblioteca implementa tres algoritmos de agrupamiento, los cuales realizan sus operaciones sobre grupos y no se preocupan si en una iteración se realizaron o no divisiones de grupos. Esto queda a consideración de las clases que los reutilicen, posibilitándoles que construyan la jerarquía de grupos de forma divisiva.

✓ Fast Clustering

Elimina aquellas aristas que tienen una similitud inferior al umbral especificado. Observe en la Figura 2.3.1 que la clase FastClustering para su creación necesita un objeto de tipo iEdges. Esta clase contiene un método execute que para su ejecución necesita un objeto del tipo iGroup y un valor de tipo primitivo float que representa el umbral de corte.

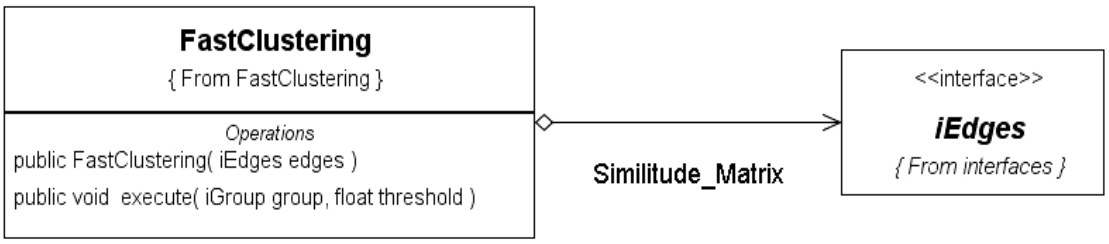

Figura 2.3.1. Algoritmo de agrupamiento Fast Clustering

✓ GN Betweenness

Calcula los valores de intermediación para todas las aristas en el grafo, encuentra todas las aristas con mayor valor de intermediación y las elimina. Observe en la Figura 2.3.2 que la clase GN_Betweenness tiene un método ready necesario para inicializar los valores de la clase. Este método necesita un objeto de tipo iPath para obtener los caminos de un nodo a otro, un objeto de tipo iGroup que representa el grupo a analizar y un objeto de tipo iEdges representante de la matriz de similitud entre los documentos. GN_Betweenness contiene una lista de la clase interna between que representa la intermediación entre un par de nodos del grupo.

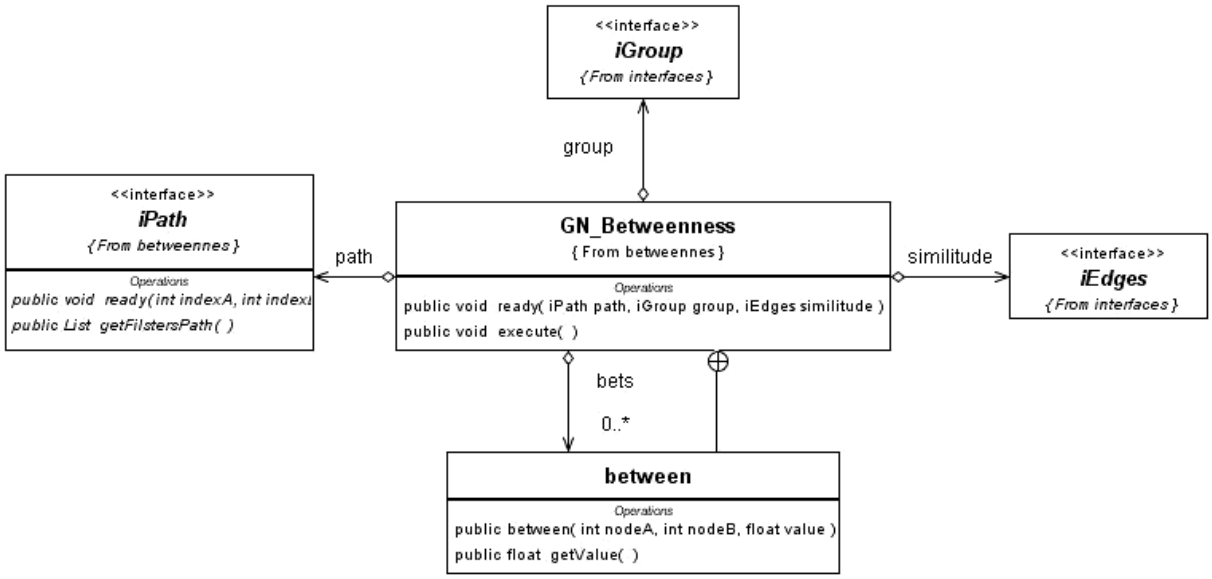

Figura 2.3.2. Algoritmo de agrupamiento GN Betweenness

- ✓ GN Betweenness & Similarity: Este algoritmo es similar al algoritmo GN Betweenness. La diferencia consiste en que sólo elimina aquellas aristas con máxima intermediación y que tienen una ponderación de similitud inferior a la media de las similitudes de todas aquellas con máxima intermediación. Este algoritmo fue creado por los desarrolladores de esta tesis en el transcurso de la misma con el propósito de reducir la formación de grupos con un único nodo. La causa de esta variante fue la deficiencia encontrada en el algoritmo GN Betweennes, ya que el proceso de agrupamiento tiende a aislar nodos que son puentes entre diferentes comunidades de nodos. La principal desventaja de este algoritmo consiste en que al eliminar menos aristas, el proceso de división de los grupos tiende a ser un poco más lento que GN Betweenness, por lo que se necesitan más iteraciones del algoritmo para lograr las divisiones. Observe en la Figura 2.3.3 que esta clase hereda directamente de GN_Betweenness y redefine el método execute.

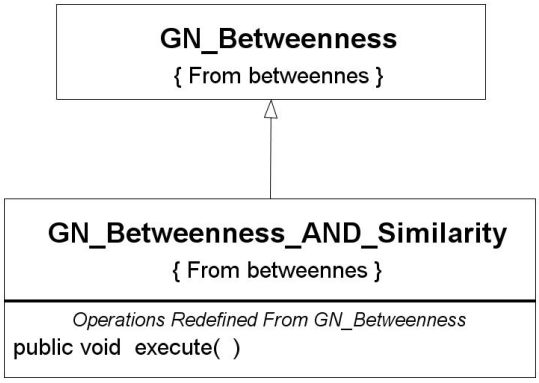

Figura 2.3.3. Algoritmo de agrupamiento GN_Betweenness_AND_Similarity

2.3.3 Métodos de evaluación

Como se dijo en el epígrafe 2.2.2 los métodos de evaluación se pueden aplicar localmente a los grupos o globalmente a colecciones de grupos. El valor devuelto por casi todos métodos es de tipo primitivo float y está normalizado entre 0 y 1.

Dos métodos para evaluar los grupos localmente fueron incorporados. Adicionalmente, un método para determinar los documentos más representativos de un grupo ha sido incluido:

- ✓ Overall Similarity

  Calcula similitud promedio de todos los documentos en un grupo. Observe en la Figura 2.3.4 que la clase OverallSimilarity necesita para su creación un objeto de tipo iEdges y contiene un método para la evaluación de los grupos getMeasure. A éste último se le pasa un objeto de tipo iGroup y devuelve el valor representante de la calidad del grupo.

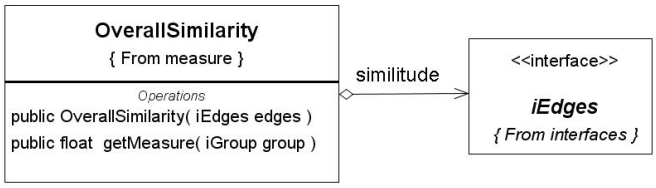

Figura 2.3.4. Método de evaluación Overall Similarity

- ✓ Mean of Rough Membership

Calcula la pertenencia aproximada promedio de los documentos al grupo, basándose en RST. Observe en la Figura 2.3.6 que este método necesita un objeto de tipo iGroup y devuelve el valor representante de la calidad del grupo.

- ✓ Lower approximation

  Determina los documentos más representativos del grupo, calculando la aproximación inferior del mismo. Observe en la Figura 2.3.6 que este método necesita un objeto de tipo iGroup y devuelve un objeto de tipo Set que contiene los documentos más representativos del grupo.

Cinco métodos encargados de validar globalmente el agrupamiento han sido incorporados observe en la Figura 2.3.6 que para usar los métodos de evaluación global de la clase RST es necesario primero ejecutar el método ready, que necesita un objeto de tipo iGroups:

- ✓ Modularity

  Mide la fortaleza de los grupos encontrados analizando las interconexiones antes y después del agrupamiento realizado. Observe en la Figura 2.3.5 que la clase Modularity para su creación necesita de un objeto de tipo iConnection y consta de un método para evaluar la colección de los grupos de la jerarquía. Éste necesita de un objeto de tipo iGroups y devuelve el valor representante de la calidad del corte especificado en la jerarquía.

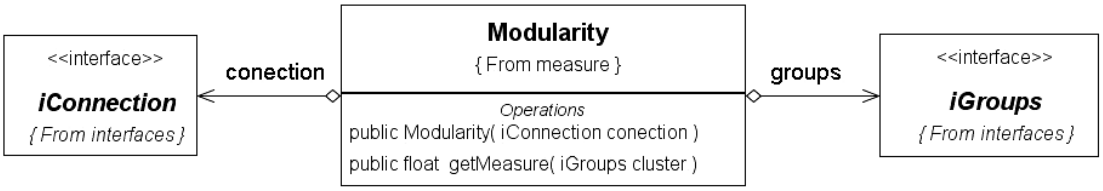

Figura 2.3.5. Método de evaluación Modularity

- ✓ Rough Accuracy

  Basada en RST y calcula la precisión aproximada de la partición a analizar. Observe en la Figura 2.3.6 que este método devuelve el valor representante de la precisión del corte especificado en la jerarquía.

- ✓ Rough Quality

  Basada en RST y calcula la calidad aproximada de la partición a analizar. Observe en la Figura 2.3.6 que este método devuelve el valor representante de la calidad del corte especificado en la jerarquía.

- ✓ Rough F-measure

  Sigue la idea de Rugh F-measure y calcula la media harmónica de la precisión y calidad aproximadas. Observe en la Figura 2.3.6 que este método devuelve el valor representante de la media armónica entre la precisión y la calidad del corte especificado en la jerarquía.

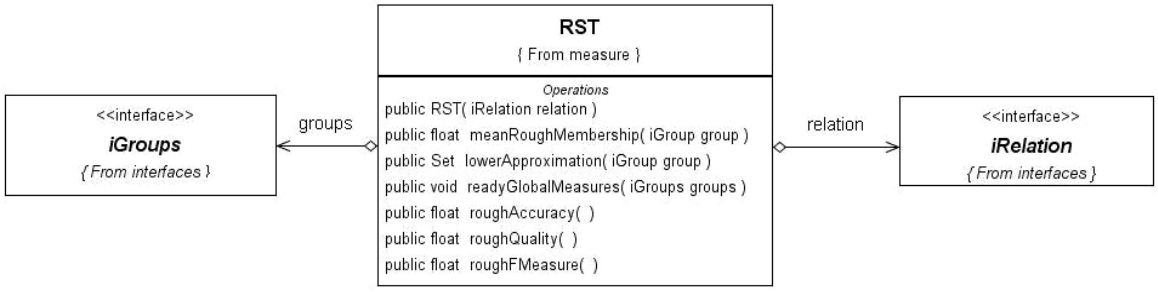

Figura 2.3.6. Métodos de evaluación basados en Teoría de los conjuntos aproximados

2.3.4 Extracción de componentes conexas

En la biblioteca se incluyó un método para la extracción de componentes conexas a partir de un grupo de nodos. Estas componentes conexas constituyen los nuevos grupos en el proceso de agrupamiento. Observe en la Figura 2.3.7 que la clase que se encarga de ésto es NodesConnectextract y requiere para su creación un objeto de tipo iConnection. Esta clase tiene un método extract al cual se le pasa una lista de nodos del grafo y devuelve una lista que contiene listas de nodos conexos.

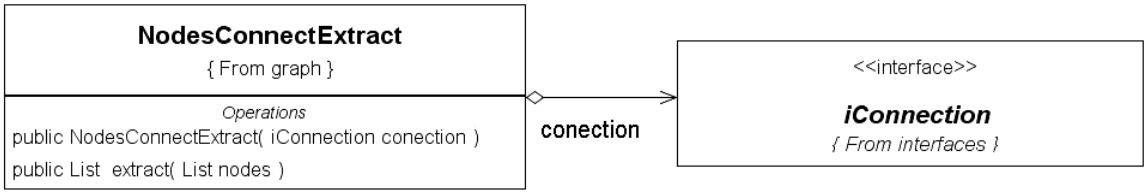

Figura 2.3.7. Extracción de componentes conexas

2.4 Módulos principales del sistema GARLucene

Como se dijo en el epígrafe 2.2.2 debido a la complejidad del sistema, éste se dividió en módulos para facilitar su desarrollo. Cada módulo se especializa y responsabiliza en su tarea, siéndole indiferente el comportamiento de los otros módulos.

2.4.1 Creación de índices

El módulo de indexado esencialmente está basado en el marco de trabajo LIUS, sólo hay que especificarle la dirección del documento o los documentos a indexar y los parámetros de configuración que él necesita. Este módulo devuelve una lista que contiene los resultados de cada proceso de indexado. De existir ausencia de al menos un documento especificado, el módulo reporta error de lectura por acceso denegado. Cuando el documento está corrupto, se retorna un error en la transformación del documento. Observar en la Figura 2.4.1 que la clase encargada de este proceso es StandardIndexing que hereda de la clase abstracta aIndexing.

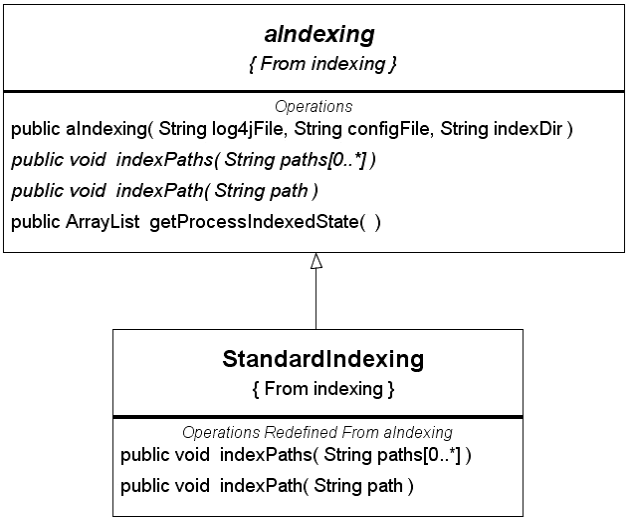

Figura 2.4.1. Diseño del proceso de indexado de los documentos

2.4.2 Recuperación de documentos

La recuperación de los documentos está soportada esencialmente por y Lucene. LIUS se utiliza para la recuperación de los documentos mediante consultas. Así, se obtienen los textos de los documentos que satisfacen la consulta. Lucene se utiliza para comprobar la existencia del índice y la obtención de algunas propiedades de los documentos, por ejemplo, su dirección. Observar en la Figura 2.4.2 que la clase encargada de realizar la recuperación e interactuar directamente con el marco de trabajo LIUS y realizar la consulta es StandardFound. Esta clase hereda de la clase Searching e implementa un método llamado run que necesita como parámetro un objeto de tipo String representando la consulta a realizar. Este método devuelve un objeto de tipo iFound. El método collection, perteneciente a iFound, devuelve un objeto de tipo Iterator. Este objeto permite iterar sobre los documentos resultantes de la consulta.

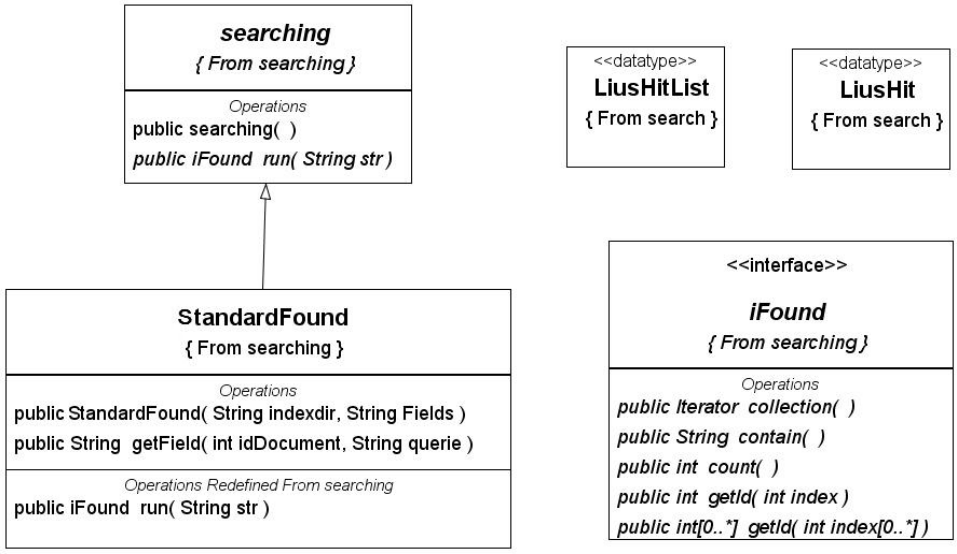

Figura 2.4.2. Diseño del proceso de la recuperación de la colección de documentos

2.4.3 Preprocesamiento textual

El sistema trabaja con textos (datos no estructurados), por tanto, la representación textual es indispensable para su procesamiento posterior (Lewis 1992). En esta investigación se ha seleccionado la representación espacio-vectorial (Vector Space Model; VSM) (Salton, Wong et al. 1975) por ser efectiva para representar documentos, ajustarse a otras formas de indexado

y ser ampliamente reconocida en la comunidad de minería de textos. Además, Lucene provee herramientas que permiten obtener y manipular esta representación. En VSM cada documento es identificado como un vector de rasgos en un espacio en el cual cada dimensión corresponde a términos indexados distintos. Un vector documento dado, en cada componente tiene un valor numérico para indicar su importancia. La representación resultante del texto es equivalente a la representación atributo-valor (Joachims 1997, Lanquillon 2001). Así, con esta representación la conformación de los sistemas de decisión para la aplicación de la teoría de los conjuntos aproximados queda de una forma natural. Esta es otra de las razones para la selección de VSM.

Debido a la complejidad del módulo de representación textual se decidió dividirlo en submódulos para tener una mejor representación de tan importante proceso. Transformación del corpus, extracción de términos, normalización y pesado de la matriz, reducción de la dimensionalidad son los submódulos que componen el preprocesamiento textual (Lanquillon 2001).

Transformación del corpus es el primer submódulo que se aplica y su objetivo principal es convertir los ficheros de entrada en una secuencia de ítems lingüísticos, los cuales son referidos como tokens de palabras. El primer paso en la trasformación del corpus es reconocer los componentes textuales desde los diferentes formatos de textos, pero éste formó parte de los módulos de indexado y recuperación. Observar los subepígrafes 2.4.1 y 2.4.2. Asumiendo que los textos recuperados de los documentos se encuentran en texto plano, se puede asumir que cada texto de documento debe ser dividido en una secuencia de tokens[18] para su posterior transformación.

Se utilizó una modificación de la clase StandardAnalyzer de la biblioteca Lucene para realizar las transformaciones que se aplican en este módulo, agregándole la eliminación de los tokens alfanuméricos y la obtención de raíces, método que se encuentra en Lucene. Algunas de estas transformaciones son:

- ✓ Eliminar las marcas de puntuación al final de los tokens.

[18] Los tokens son cadenas de caracteres delimitadas por espacios en blanco (por ejemplo espacios, cambios de líneas, tabs)

- ✓ Identificar o marcar los nombres de personas, localidades, organizaciones y productos. Este etiquetado lo realiza Lucene pero no es tratado en GARLucene.
- ✓ Convertir las letras todas a minúsculas.
- ✓ Quitar los tokens que están dentro de la lista de palabras de paradas. La lista de palabras de parada fue enriquecida: adicionándole más palabras de parada e incorporando algunas contracciones del idioma inglés.
- ✓ Eliminar los apóstrofes del tipo 's y la eliminación de los acrónimos al quitarles los puntos.
- ✓ Omitir los tokens que contienen caracteres alfanuméricos o los tokens constituidos por un solo carácter.
- ✓ Sustituir los tokens por sus raíces (stemming).

El submódulo de transformación devuelve por cada documento un vector atributo-valor, donde los atributos son los tipos de palabras (representantes de los tokens), los valores numéricos asociados a cada tipo indican su importancia dentro de cada documento a partir del número de tokens que aparecen en ellos. Este procesamiento se mezcla con los submódulos: extracción de términos, normalización y pesado. Observe en la Figura 2.4.3 que la clase encargada de este proceso general es SearchingResult la cual realiza las búsquedas mediante el método execute. Este método necesita para su ejecución un objeto de tipo String representando la consulta a realizar y otro de tipo iIndex que contiene todos los repositorios de índices a consultar. Esta clase tiene dos métodos fundamentales:

- ✓ getDocumentTerms devuelve un arreglo contenedor de objetos de tipo iDocumentTerms, objeto que facilita el proceso de creación del VSM a partir de un documento. Este método se redefinió a partir de la interfaz iSearchingResult.
- ✓ getStore devuelve un objeto de tipo iStoreData que contiene los datos de los documentos recuperados para ser mostrados al usuario.

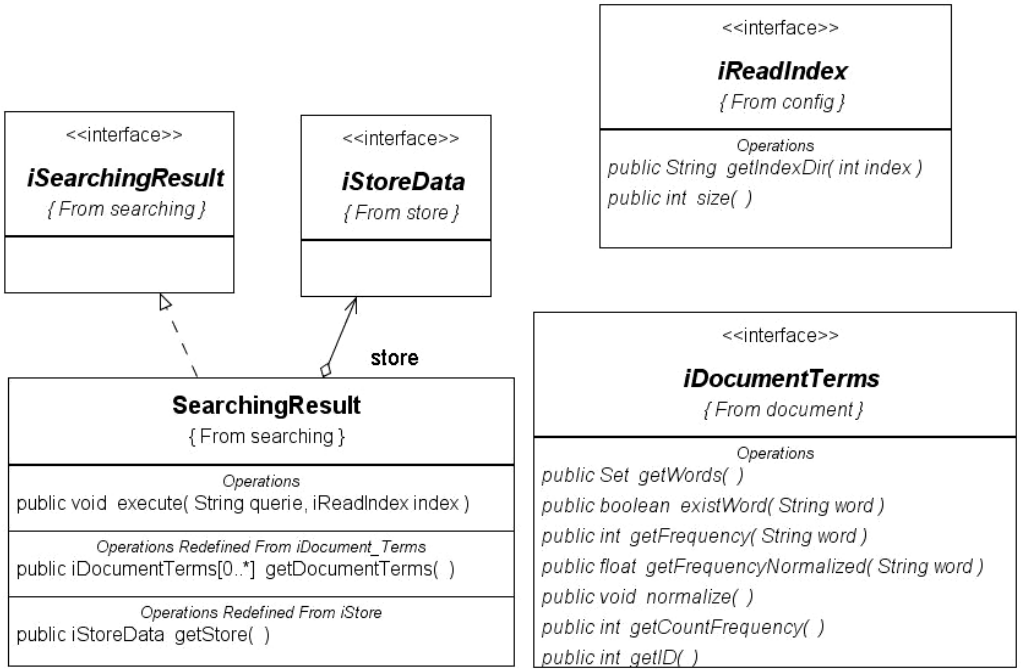

Figura 2.4.3. Diseño de clases de la primera transformación de los resultados

El módulo de extracción de términos parte de una secuencia de tokens, obtenida a partir de la transformación del corpus y produce una secuencia de términos indexados basados en esos tokens. La forma en que esa secuencia de términos indexados es usada depende del procesamiento posterior que se le vaya a realizar al documento y si el vocabulario ha sido construido o no. En la aplicación que se presenta en este trabajo el vocabulario no se establece a priori, sino que se crea a partir de los términos indexados resultantes de la extracción de términos.

Acorde a la definición de VSM, las dimensiones del vector corresponden a distinguir los términos indexados en la colección de documentos bajo consideración. Representar documentos del lenguaje natural por el significado de un conjunto de índices de términos es un reto, sobre todo porque la información siempre depende del contexto. Así, se hace necesario considerar los niveles que se han establecido para realizar análisis lingüístico de los textos, observe el Anexo 2. En (Blair 1992) se comenta que puede ser razonable asumir la representación de textos más compleja teniendo en cuenta que niveles más altos de análisis textual deben tender a la obtención de herramientas más efectivas. Pero, mientras más

compleja sea la definición de términos indexados, más compleja será la representación de los textos, y la dimensionalidad de los rasgos crecerá correspondientemente. Por tanto, escoger un nivel adecuado sobre el cual basar la definición de los términos es siempre un equilibrio entre la expresividad semántica y la complejidad de la representación. En la mayoría de las aplicaciones de la minería de textos, la definición de términos simples es dominante. Típicamente, se enfoca el análisis de textos a partir de los dos primeros niveles, y a su vez, alguna información sintáctica puede ser también incorporada.

En esta aplicación se realiza un análisis léxico de los textos identificando las palabras simples como rasgos (Salton and Buckley 1988). Así, se explota básicamente el plano estadístico de los textos y no se considera la secuencia de aparición de las palabras en un documento (modelo bolsa de palabras; bag-of-words model) (Lewis and Ringuette 1994), aunque alguna información sintáctica puede enriquecer posteriormente los resultados. Esta selección se debe a que la definición de términos es independiente del lenguaje y computacionalmente muy eficiente. Además, la representación resultante, a diferencia de los *n*-gramas (Cavnar and Trenkle 1994), es fácil de analizar por los humanos. Por tanto, es posible lograr la interpretabilidad que se requiere al extraer los términos relevantes como parte del post-agrupamiento. Adicionalmente, con este tipo de rasgos es natural obtener el sistema de información requerido para la aplicación de la teoría de los conjuntos aproximados. Otra razón es que la extracción de palabras es independiente del dominio, elemento ventajoso a diferencia del uso de frases que tienden a ser dependientes del dominio (Sahami 1998). Una desventaja es que cada inflexión de una palabra es un posible rasgo y el número de éstos puede ser innecesariamente grande, requiriendo la aplicación de técnicas de reducción de dimensionalidad, de ahí la necesidad de ese submódulo. Lucene soporta un análisis léxico de las palabras, adicionalmente incluye las etiquetas de los términos en el análisis.

Antes de reducir la dimensionalidad es necesario tener en cuenta el submódulo de normalización y pesado de la matriz para tener una mejor representación del contenido de los documentos. Éste genera un vector pesado para cualquier documento basado en el vector de frecuencia de términos. Cada peso expresa la importancia de un término en un documento con

respecto a su frecuencia en todos los documentos. En GARLucene se normalizan los vectores según los aspectos siguientes:

- ✓ Se normaliza la frecuencia de aparición de los términos en los documentos dividiendo la frecuencia absoluta de cada término en el documento por la frecuencia total de aparición de los términos en este documento.
- ✓ Se normaliza la frecuencia de aparición de los términos en los documentos dividiendo la frecuencia de aparición previamente normalizada por la suma global de las frecuencias.

  Observe en la Figura 2.4.4 que VSM es la clase encargada de realizar estos procesos y para su creación necesita un arreglo de objetos de tipo iDocumentTerms. La clase VSM hereda de la clase Matrix que a su vez implementa los métodos de la interfaz iMatrix.

  - o getMatrixValue: necesita el índice del documento y del término a buscar, para retornar el resultado del proceso posterior a las normalizaciones comentadas anteriormente en forma del tipo de dato primitivo float.
  - o getCountDocumentWithTerm: retorna un tipo de dato primitivo int representante de la cantidad de documentos que tienen un término especifico.

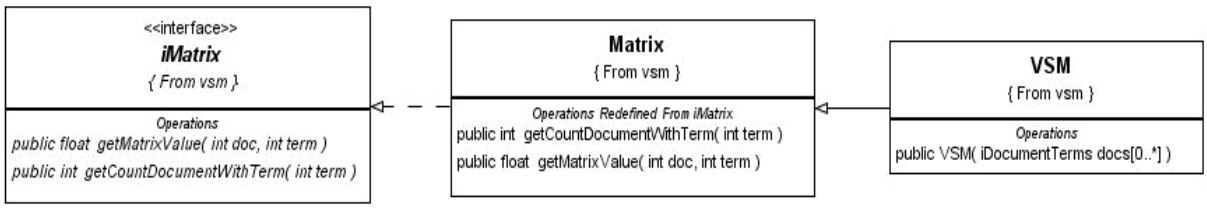

Figura 2.4.4. Diseño de clases del VSM de los documentos recuperados

- ✓ Se calcula la importancia de un término en la colección de documentos utilizando la variante de la fórmula TF-IDF publicada en (Manning and Shütze 2000, Berry 2004). Observe en la Figura 2.4.5 que la clase principal es TF_IDF que hereda de la clase abstracta TF y redefine el método weight que necesita dos parámetros de tipo primitivo int. El primero, es el índice del documento, y el segundo, el índice del término a pesar.

Este método devuelve un valor primitivo float que representa el peso del término *j* en el documento *i*.

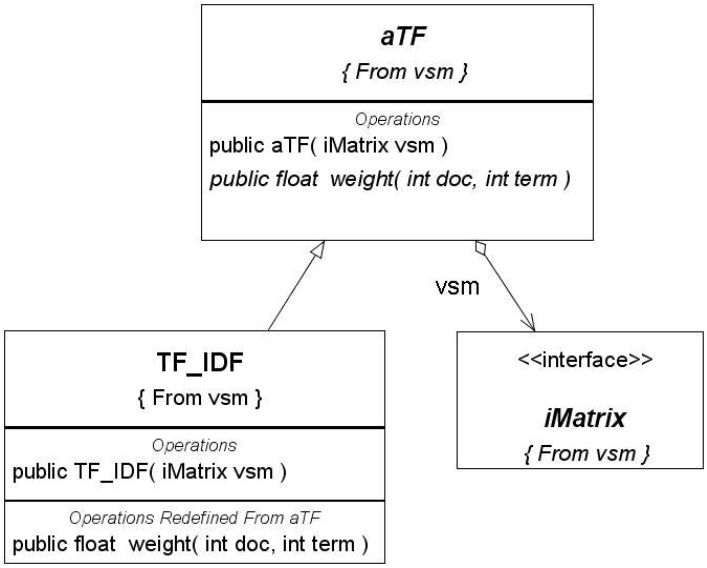

Figura 2.4.5. Diseño de clases de la fórmula TF_IDF

Una vez que el VSM fue pesado se puede aplicar el submódulo de la reducción de dimensionalidad. Al seleccionar las palabras como los términos a indexar en la representación VSM, es esencial controlar la dimensionalidad del espacio del vector documento. Las dos razones principales son: (i) la complejidad de muchos algoritmos de agrupamiento depende crucialmente del número de rasgos y reducirlo es necesario para hacer estos algoritmos tratables y (ii) existen palabras que son irrelevantes y provocan la obtención de peores resultados, por tanto, eliminarlas puede realmente aumentar la eficiencia del agrupamiento a realizar.

En este trabajo se utiliza el término reducción de dimensionalidad para abarcar cualquier técnica que su objetivo sea controlar la dimensionalidad del vector, incluyendo técnicas de selección de rasgos. Estas toman como entrada un conjunto de rasgos y la salida es un subconjunto relevante de éstos (Dash and Liu 1997). Realizar una búsqueda exhaustiva es intratable teniendo en cuenta que el número de palabras usualmente es elevado; por tal motivo, la selección de rasgos es guiada por heurísticas (Schürmann 1996). La heurística aplicada fue calidad de términos, asociándole a cada término el valor de su calidad teniendo en cuenta la

aparición de éste en la colección de documentos. Observe en la Figura 2.4.6 que la clase principal es TermQuality y necesita para su creación dos parámetros; uno de tipo iMatrix que representa la matriz a reducir y otro de tipo primitivo int que representa la cantidad de términos a conservar. La clase TermQuality implementa la interfaz iTermQuality mediante el método que retorna un objeto de tipo iMatrix que representa la matriz redimensionada.

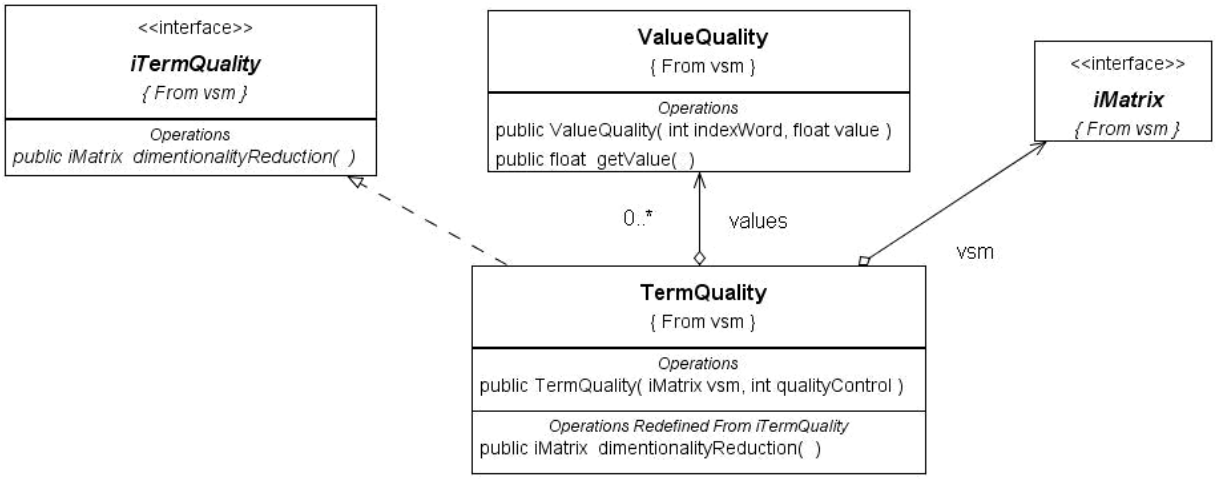

Figura 2.4.6. Diseño de clases de Calidad de términos

2.4.4 Representación de la colección de documentos recuperados

Los documentos recuperados se representan como grafos no dirigidos y ponderados, donde cada documento es un nodo y las aristas entre ellos están ponderadas con la similitud Coseno entre los mismos calculada a partir de VSM resultante del preprocesamiento textual. Observe en la Figura 2.4.7 que la clase encargada de crear la matriz de similitud entre los documentos es Cosine, la cual hereda de la clase abstracta aSimilitude. Para su creación necesita un objeto de tipo iMatrix y redefine los dos métodos que existen en aSimilitude, uno de ellos devuelve la similitud que existe entre dos documentos dados y el otro devuelve un objeto de tipo iSimilitudeMatrix que representa la matriz de similitud.

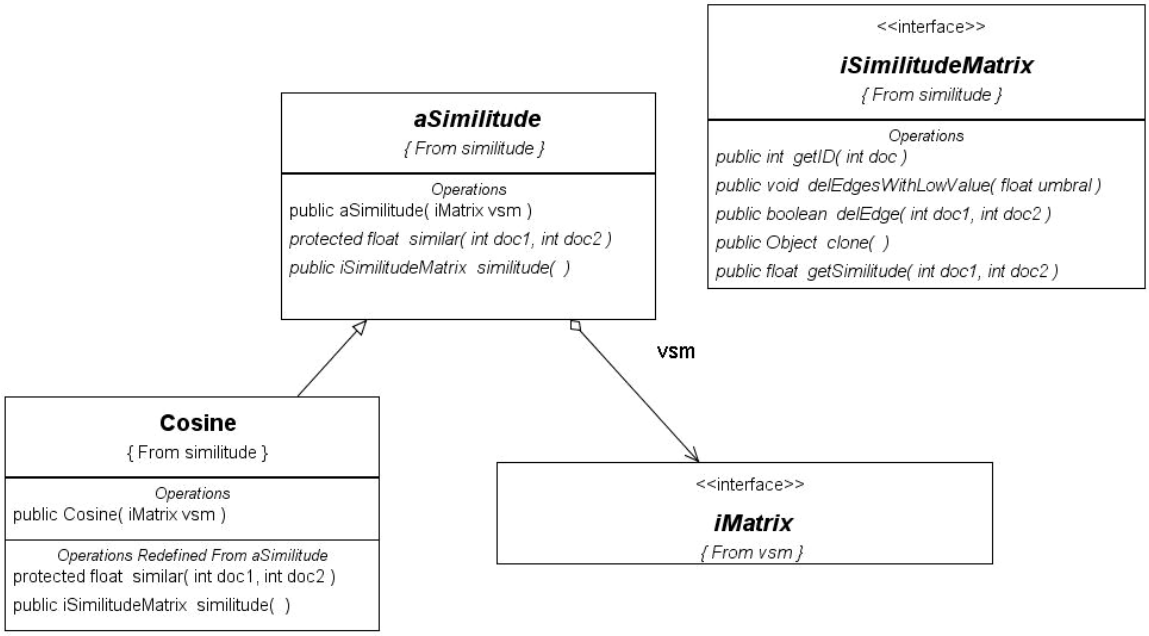

Figura 2.4.7. Diseño de clases de la similitud coseno

Inicialmente este grafo es completo, teniendo aristas de valor muy bajo o nulo que no aportan mucha información para la creación de grupos afines entre los documentos recuperados. Teniendo en cuenta esto, se decidió realizarle un corte donde se eliminaron aquellas aristas cuyo valor fuera inferior a la media de los valores de todas las aristas del grafo. De esta forma se agiliza el proceso de agrupamiento ya que existen menos conexiones en todo el grafo.

A partir de la matriz de similitud resultante del proceso anterior se construye una copia. Con el objetivo de utilizar una para el proceso de agrupamiento y la otra para la evaluación de los grupos formados por ese proceso.

2.4.5 Agrupamiento

El proceso de agrupamiento en éste modulo es sencillo, porque se reutilizan las clases existentes en la biblioteca cluster.jar que representa a los algoritmos de agrupamiento. Por tanto, es necesario limitarse solamente a la formación de los nuevos grupos obtenidos a partir de las iteraciones sucesivas de esos algoritmos sobre cada grupo formado hasta que se cumplan los criterios de parada. Estos criterios en GARLucene son:

- Obtener un resultado de la medida de evaluación que representa la calidad del grupo mayor que 0.95. Valor establecido en este trabajo debido a que un grupo con esa calidad es lo suficientemente bueno como para no ser dividido.
- Que el grupo candidato a ser dividido esté formado por un solo documento.

Como al grafo de similitud se le realiza un corte dependiendo del umbral especificado, este corte puede provocar que desde ese momento ya se identifiquen componentes conexas en el grafo. Por tanto, los algoritmos de agrupamiento jerárquicos divisivos implementados parten de una única componente conexa o se aplican simultáneamente a cada una de las componentes conexas obtenidas por el corte del umbral. Para cada componente conexa se procede de la forma siguiente:

1. Verificar si el grupo cumple con los criterios de parada, en este caso FIN.
2. Aplicar una iteración del algoritmo de agrupamiento elegido.
3. Extraer componentes conexas. Si se formaron nuevos grupos, a cada uno de ellos aplicarle el algoritmo desde el paso 1, sino volver al paso 2.

El tercer paso del algoritmo consiste en hacer la extracción de componentes conexas. Si no se generaron nuevas componentes conexas se devuelve vacío, pero si se crearon nuevas el algoritmo retorna la descripción de cada componente conexa; por ejemplo, identificador de su grupo padre y el o los valores resultantes de las medidas de validación del grupo.

Observe la Figura 2.4.8 donde se muestra el diagrama de las clases controladoras para estos algoritmos. La clase principal es ClusterizerController que hereda de la interfaz iClusterizerController y redefine su método execute que se encarga de la ejecución del algoritmo expuesto anteriormente para agrupar. El paso dos está identificado por el método clusterizer siendo abstracto para esta clase y el tercer paso quedaría definido por el método extract que se encarga de la extracción de los nodos conexos de un grupo de documentos. Las clases FastClusteringController y BetweennessController redefinen el método clusterizer y la clase BetweennnessSimilarityController, al heredar de BetweennessController, sólo cambia el

valor del objeto protegido de tipo GN_Betweenness por GN_Betweenness_AND_Similarity que hereda de éste.

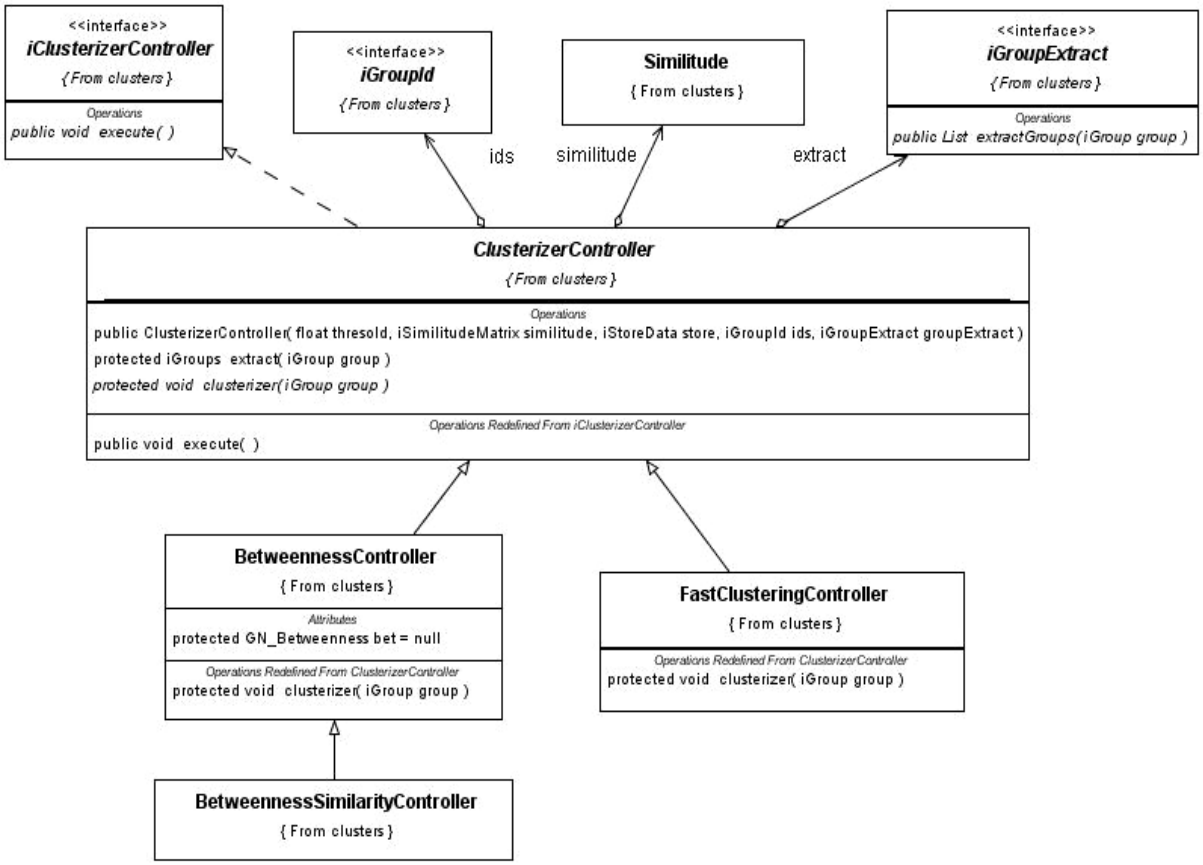

Figura 2.4.8. Diseño general de los algoritmos de agrupamiento

2.4.6 Valoración del agrupamiento

El módulo de evaluación del agrupamiento en GARLucene, reutiliza los algoritmos de evaluación de la biblioteca cluster.jar. Este módulo evalúa los grupos desde dos ópticas diferentes

- ✓ Local: al formarse un nuevo grupo el sistema automáticamente lo evalúa.
- ✓ Global: el sistema evalúa la colección de grupos que representan un corte en la jerarquía.

Observe en la Figura 2.4.9 el diseño general de clases para estos dos tipos de evaluaciones. La interfaz iMeasureValue es la encargada de definir la extracción de los valores de las medidas,

ya que un tipo de evaluación puede tener más de una medida implementada. También se creó la interfaz iMeasure, ésta hereda los métodos de iMeasureValue y define un nuevo método para los nombres de las medidas implementadas. De esta interfaz heredan iLocalMeasure que define los nombres de las medidas locales que serán aplicadas, iGlobalMeasure que define los nombres de las medidas globales a aplicar y la clase Measure que implementa esta interfaz. Además, en este diseño de clases existe la clase LocalMeasure que implementa la interfaz iLocalMeasure y hereda de la clase Measure. Así, el método nameMeasure retorna los nombres de las medidas locales que se implementan. De igual forma existe la clase GlobalMeasure que implementa la interfaz iGlobalMeasure y hereda de la clase Measure. El método nameMeasure retorna los nombres de las medidas globales que se implementan.

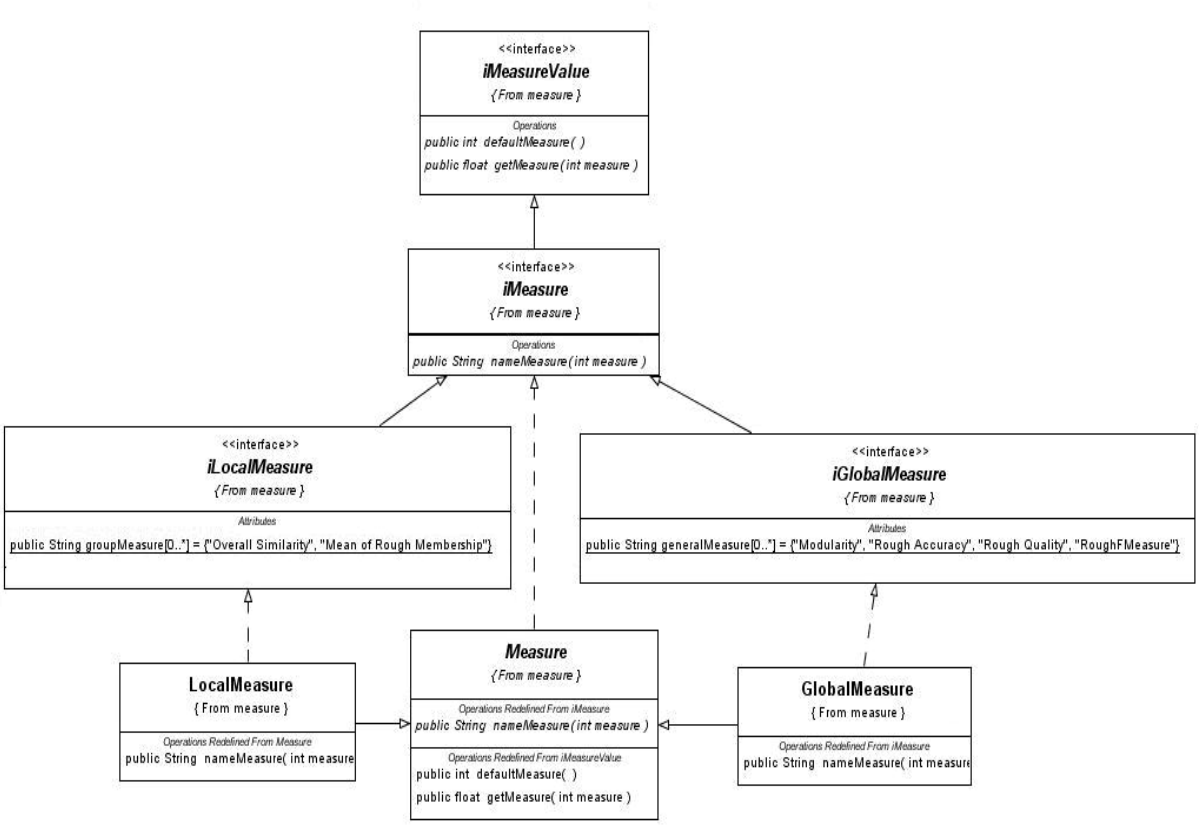

Figura 2.4.9. Diseño de la evaluación local y global de los grupos

Estas evaluaciones son tratadas de tres formas distintas. Observe en la Figura 2.4.10 la clase abstracta aCalculateMeasures que define los nombres de las distintas opciones de

agrupamiento que contiene GARLucene. Ésta define los métodos abstractos getLocalMeasure que retorna un objeto de tipo iLocalMeasure y getGlobalMeasure que retorna un objeto de tipo iGlobalMeasure. De la clase abstracta aCalculateMeasures heredan las clases BasedRST_AND_OverallSimilarityWithModularity, Based_RST y OverallSimilarityWithModularity. Cada una de ellas implementa los métodos abstractos heredados de aCalculateMeasures:

- ✓ OverallSimilarityWithModularity (Overall Similarity y Modularity) evalúa los grupos localmente con el algoritmo de evaluación Overall Similarity y globalmente con Modularity.
- ✓ Based_RST (basadas en Rough Set Theory) evalúa los grupos localmente con el algoritmo Mean of Rough Membership y globalmente con Rough Accuracy, Rough Quality y Rough F-measure.
- ✓ BasedRST_AND_ OverallSimilarityWithModularity (Todas) evalúa los grupos localmente con los algoritmos Overall Similarity y Mean of Rough Membership, siendo esta última la representante de la calidad del grupo. La evaluación global se realiza con Modularity, Rough Accuracy, Rough Quality y Rough F-measure.

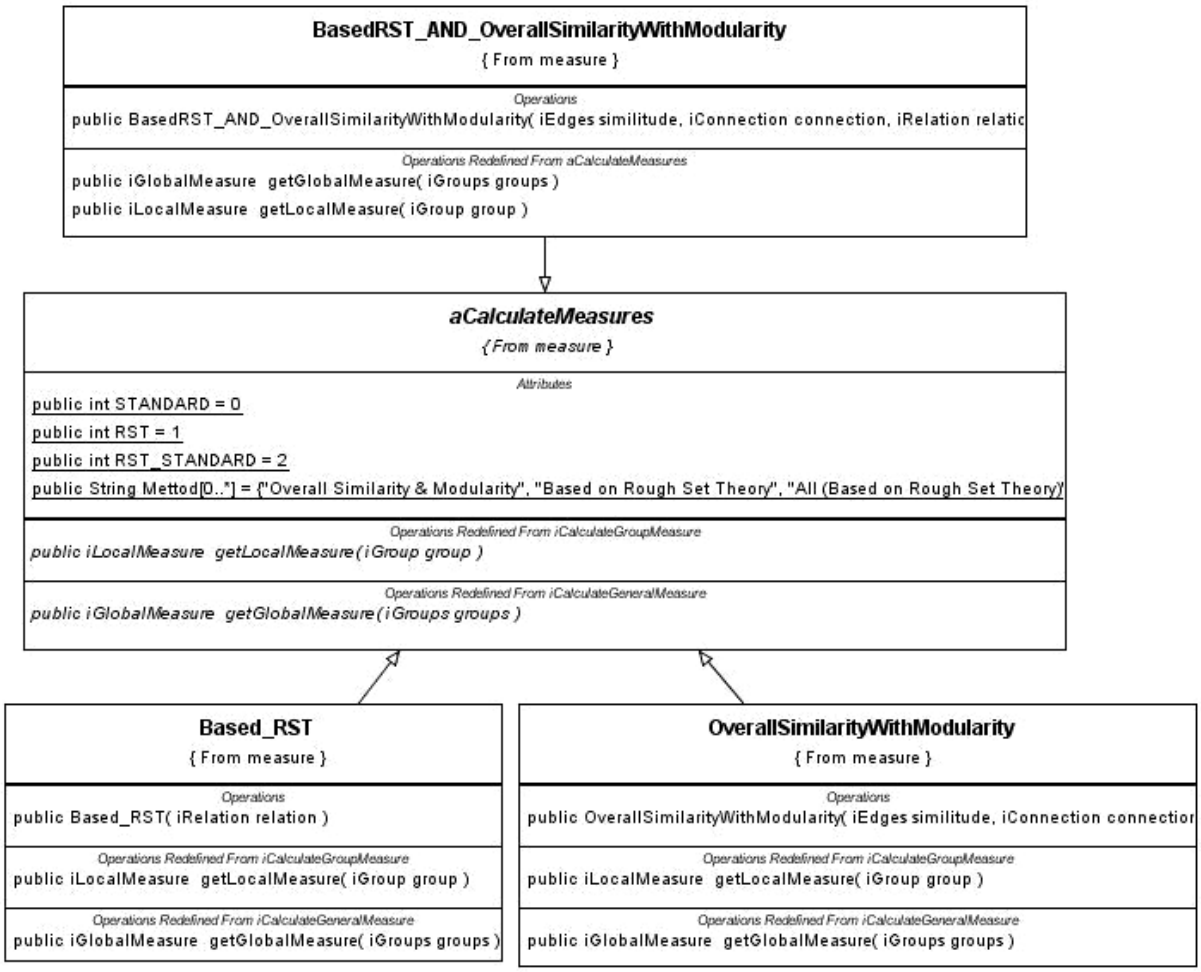

Figura 2.4.10. Diseño de clases de las opciones de agrupaciones en GARLucene

Además, cuando el sistema evalúa un grupo haya los documentos más representativos de éste, utilizando la aproximación inferior. Observe en la Figura 2.4.11 que la clase principal DocumentsMoreRepresentative implementa la interfaz iDocumentsMoreRepresentative. La cual define el método getDocumentMoreRepresentative que retorna un objeto de tipo Set conteniendo los documentos más representativos del objeto iGroup representante de los documentos a analizar. Esta clase principal necesita un objeto de tipo iSimilitudeMatrix que representa la matriz de similitud que se tiene para evaluar.

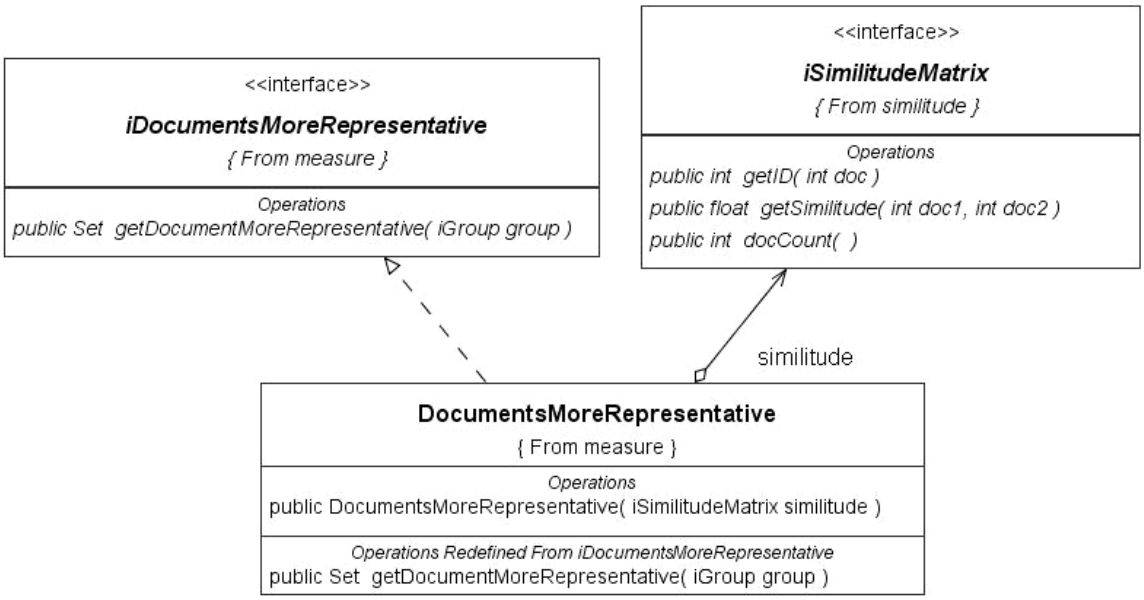

Figura 2.4.11. Diseño de clases para la obtención de los documentos más representativos

2.5 Diseño de la interfaz de usuarios

Se utilizó un modelo de tres capas donde existen objetos del dominio, objetos controladores y las interfaces. Observe la Figura 2.5.1 que cada capa tiene una función específica y se apoya en la capa inferior para resolver las tareas asignadas.

Figura 2.5.1. Diseño de capas de la interfaz de usuarios

En la primera capa se encuentra la interfaz de usuario que contiene todos los objetos que de una forma u otra interactúan directamente con el usuario (interfaces).

En la segunda capa se encuentran los controladores internos y externos. Siendo los controladores internos los encargados de controlar todos los objetos del dominio que interactúan con objetos existentes sólo dentro de GARLucene y los controladores externos se encargan de controlar todos los objetos del dominio que interactúan con los diferentes módulos externos al sistema.

Por último, en la tercera capa es donde se encuentran todos los objetos del dominio. Estos se dividen en dos grupos: objetos del dominio internos y externos.

Internos:

- ✓ Configurar: se encarga de los parámetros de configuración del sistema.
- ✓ Representar: se encarga de la representación en un grafo de la colección de documentos recuperados.
- ✓ Almacenar: se encarga del almacenamiento de datos recuperados para mostrar al usuario.
- ✓ Preprocesar: se encarga del procesamiento del texto recuperado.

Externos:

- ✓ Indexar: se encarga de la indexación de los archivos hacia los repositorios.
- ✓ Recuperar: se encarga de la recuperación de los archivos que se encuentran en los repositorios a partir de una consulta.
- ✓ Agrupar: se encarga del agrupamiento de los documentos recuperados.
- ✓ Validar: se encarga de validación de los grupos.

Observe en la Figura 2.5.2 del diseño de clases correspondiente a la interfaz gráfica de GARLucene que existe una clase principal Controller que se encarga del funcionamiento del sistema.

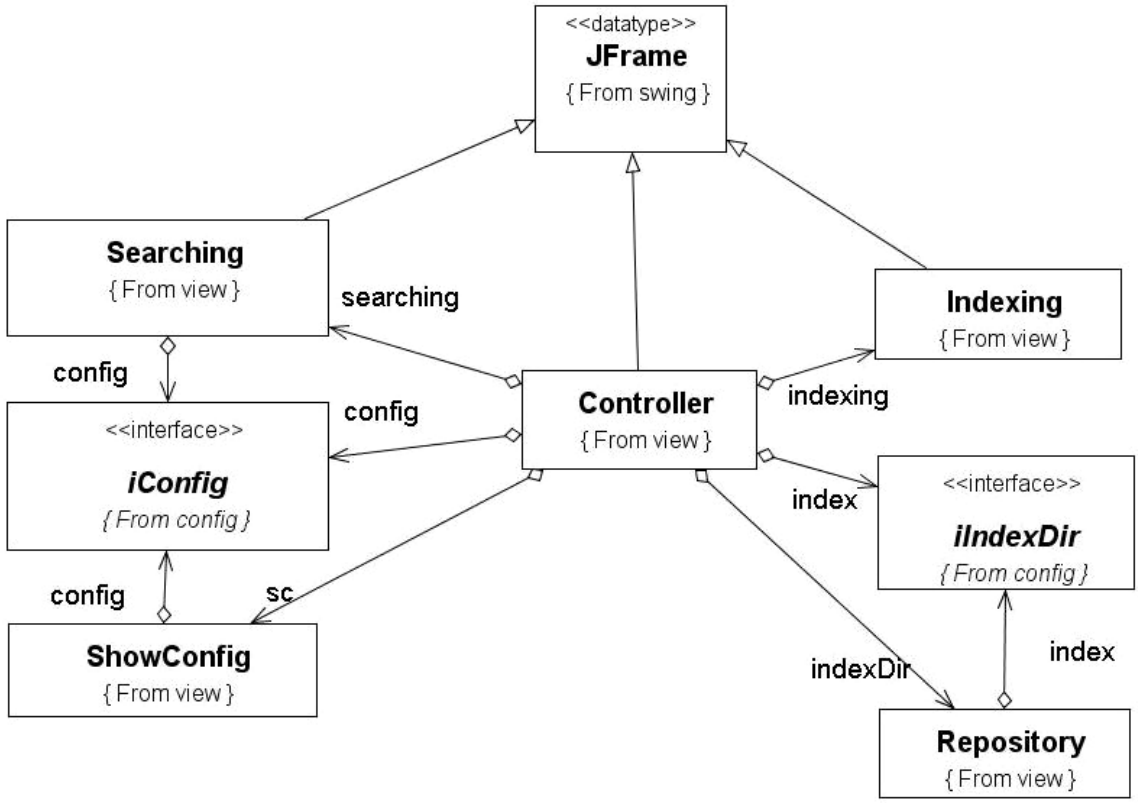

Figura 2.5.2. Diseño de la interfaz gráfica

2.6 Conclusiones parciales

En este capítulo se presentó el diseño general de GARLucene. Este diseño se realizó utilizando la metodología de Diseño Orientada a Objetos, de forma tal que el diseño es extensible y reutilizable con comodidad. El uso de las facilidades de LIUS y Lucene fue fundamental en el desarrollo del sistema, particularmente para el desarrollo de los módulos de creación de índices, recuperación de información y procesamiento textual. Cluster.jar es un módulo que permite esencial la reutilización de algoritmos de agrupamiento y medidas de validación en aplicaciones independientes a GARLucene. Se implementaron dos variantes del agrupamiento según intermediación GN con el objetivo de mejorar el desempeño de este tipo de algoritmos.

3 MANUAL DE USUARIOS

GARLucene fue desarrollado completamente en JAVA, característica que lo convierte en multiplataforma. Además, permite la indexación de múltiples tipos de ficheros (Ms Word, Ms Excel, Ms PowerPoint, RTF, PDF, XML, HTML, TXT, Open Office, ZIP) por lo que es necesario que el usuario especifique en los parámetros de configuración la dirección del programa con el cual desea ver algún artículo recuperado. También se requiere en el directorio de índices la especificación de al menos una dirección en la cual se desea que el programa guarde y recupere la información de los artículos científicos.

El sistema procesa documentos de diferentes idiomas; no obstante, el procesamiento de documentos en inglés hace que GARLucene obtenga un mayor desempeño.

3.1 Objetivo del sistema GARLucene

El objetivo del sistema GARLucene es asistir al usuario en el procesamiento (recuperación, indexado y agrupamiento) de un conjunto de documentos, por ejemplo artículos científicos, ya sea para comenzar una revisión del estado del arte, organizar materiales por equipos de estudiantes para la docencia, organizar por temáticas los artículos que le han llegado al comité científico del programa de un evento, o sencillamente para tener una idea de las asociaciones que existen a partir de la colección textual resultante de un proceso de recuperación de la información. Este procesamiento se realiza de una forma automática, totalmente no supervisada y sin requerir conocimiento a priori del dominio de aplicación.

3.2 Generalidades del sistema GARLucene

GARLucene es una herramienta desarrollada completamente en JAVA, su código está libre y es multiplataforma. Sólo requiere que el Sistema Operativo tenga instalado Java Runtime Enviroment (JRE). Su archivo .jar ocupa 286 KB y requiere que junto con éste se encuentren las carpetas:

- ✓ Config: contiene los archivos necesarios para la configuración del sistema.
  - liusConfig.xml: contiene toda la configuración del marco de trabajo de LIUS.

- log4j.properties: es utilizado por LIUS en el proceso de indexado y recuperación de los documentos.
- stopWords: contiene la lista de palabras de parada a eliminar en el análisis de la información contenida en los artículos científicos.
- help.mht: archivo de ayuda del sistema
- garlucene.gar: este guarda la configuración dada por el usuario con la posibilidad de utilizar en una ejecución posterior del programa. Este archivo no es necesario para la ejecución del sistema ya que en caso de ausencia el sistema cuenta con una configuración por defecto; usada por el programa hasta que el usuario realice algún cambio. Además, el sistema cada vez que se cierra salva automáticamente la configuración que tenga en ese momento, por lo que si no existe el fichero se crea.

✓ Lib: contiene las librerías de diferentes ficheros .jar necesarios para el proceso de indexado, recuperación, agrupamiento y evaluación de los grupos.

- commons-beanutils-1.6.1.jar, commons-collections-2.1.jar, commons-io-1.2.jar, commons-logging-1.0.3.jar, crimson.jar, dom.jar, javalayer-0.4.jar, jaxen-1.1-beta-6.jar, jaxen-core.jar, jaxen-jdom.jar, jdom.jar, jxl.jar, log4j-1.2.8.jar, lucene-analyzers-2.0.0.jar, lucene-core-2.0.0.jar, lucene-highlighter-2.0.0.jar, MimeType.jar, mp3spi1.9.1.jar, nekohtml.jar, PDFBox-0.7.2.jar, poi-2.5.1-final-20040804.jar, saxpath.jar, Tidy.jar, tm-extractors-0.4.jar, tritonus_jorbis.jar, tritonus_mp3.jar, tritonus_remaining.jar, tritonus_share.jar, vorbisspi1.0.jar, xalan.jar, xerces.jar, xercesImpl.jar, xml-apis.jar

  Todas estas librerías son necesarias para que el marco de trabajo LIUS funcione correctamente y son utilizadas por éste en los procesos de indexación o recuperación de los artículos científicos.

- Lius-1.0-RC2.jar

Esta librería contiene todas las clases del marco de trabajo LIUS y es necesaria para la indexación y recuperación de los artículos científicos.

- cluster.jar

  Contiene los algoritmos de agrupamiento y evaluación de los grupos utilizados en el sistema.

Tanto el ejecutable como las carpetas antes mencionadas requieren ser ubicados en un directorio donde se tenga permiso de escritura y lectura.

3.3 Interfaz gráfica de GARLucene

La interfaz gráfica del sistema GARLucene presenta una ventana principal donde existen cuatro opciones del menú principal que permiten interactuar con el sistema. Observar Figura 3.3.1.

Figura 3.3.1. Interfaz grafica de GARLucene (Foundation 2008).

- ✓ Realizar búsquedas (Search)
- ✓ Trabajar con los índices (Index)
- ✓ Configurar parámetros para manipular las colecciones de documentos recuperadas (Config)
- ✓ Acceder a la ayuda del sistema (Help)

La opción Index contiene dos sub opciones (Directory e Indexing) que se despliegan si el usuario hace clic sobre Index.

A continuación explicaremos de una forma más detallada las opciones del sistema. Observar Figura 3.3.2.

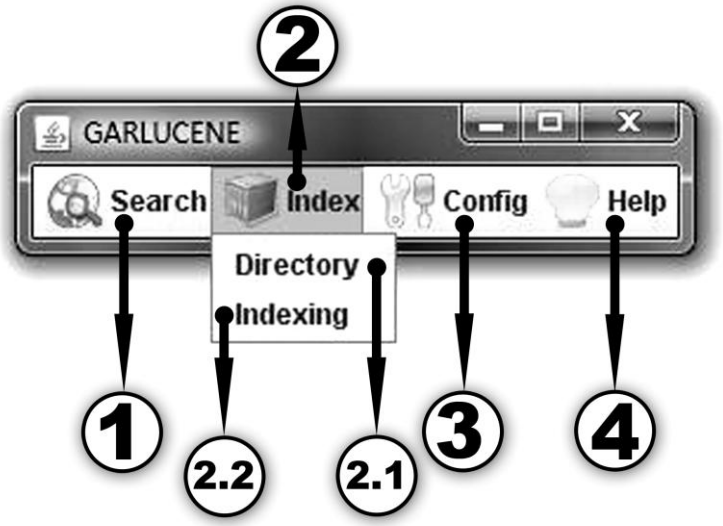

Figura 3.3.2. Interfaz gráfica de GARLucene detallada.

1. Búsqueda y recuperación de artículos científicos.
2. Despliegue de las opciones de indexado.

   2.1 Control de directorios de índices.

   2.2 Indexación de los documentos científicos.

3 Configuración general de GARLucene.

4 Ayuda del sistema.

Si se va a ejecutar GARLucene por primera vez sería conveniente configurarlo según nuestro gusto o necesidad. Por ejemplo:

- ✓ Visor: es el que muestra los artículos científicos que el usuario desea consultar. Se recomienda usar algún explorador como Internet Explorer o Mozilla Firefox, ya que estos permiten la visualización de diferentes tipos de ficheros lo cual es clave en nuestro sistema.
- ✓ Directorio de índices: contiene una lista de índices desde los cuales el usuario desea que el sistema haga la recuperación de los artículos científicos. También consta de un

directorio donde se almacenan los datos de la indexación de un archivo o artículo científico.

3.4 Operaciones con GARLucene

A continuación describiremos cada una de las operaciones que se pueden realizar con el sistema GARLucene.

3.4.1 Búsqueda y recuperación de artículos científicos

Al dar clic en el botón Search (observar la Figura 3.3.1) se muestra la ventana encargada de proporcionarnos una interfaz amigable para facilitar la búsqueda y recuperación de los artículos científicos previamente indexados. Como este proceso suele ser costoso, GARLucene en su interfaz grafica de búsqueda y recuperación, consta con las herramientas necesarias para salvar y recuperar los resultados posteriores a un proceso de búsqueda y recuperación de los artículos científicos. GARLucene utiliza un fichero de datos cuya extensión es (*.ddd). Observar la Figura 3.4.1.

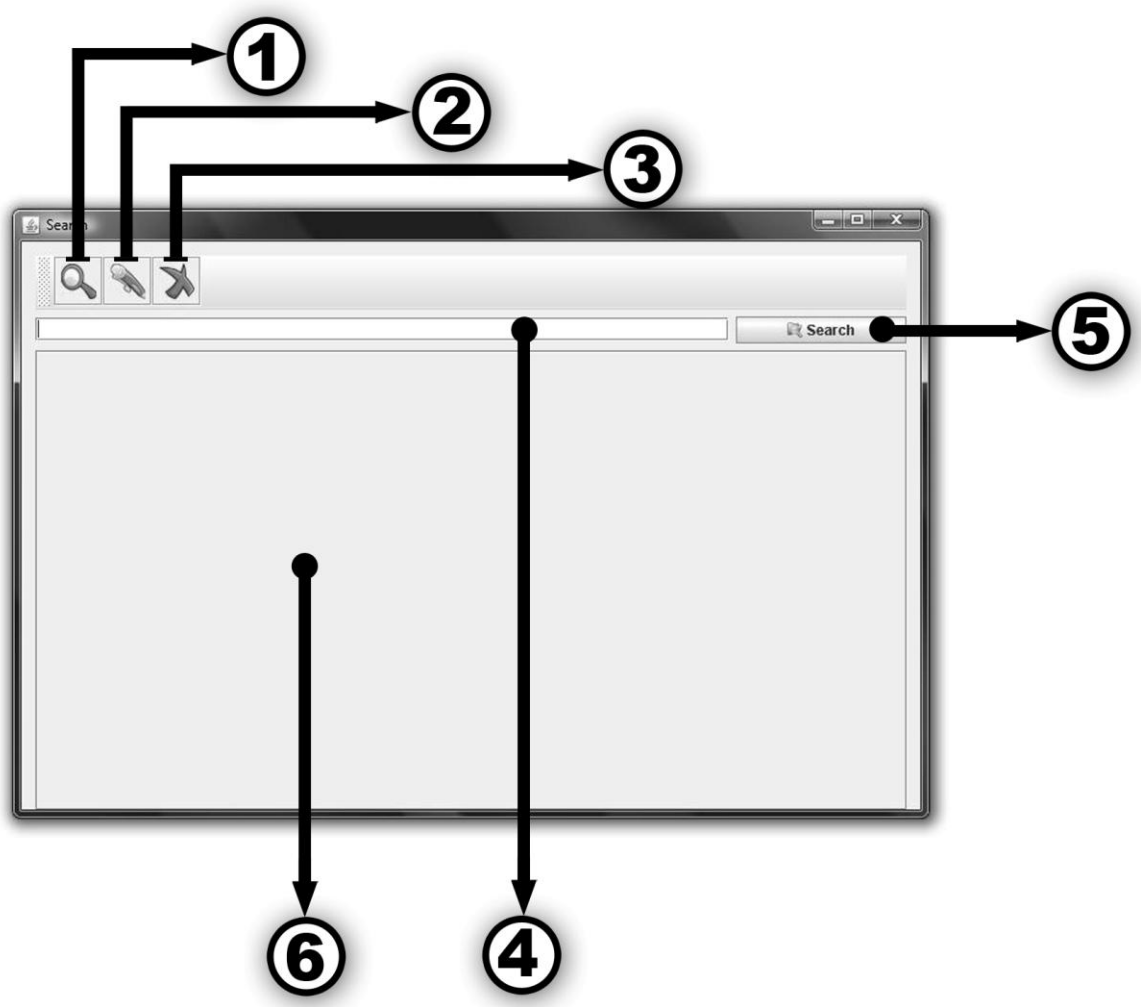

Figura 3.4.1. Búsqueda y recuperación a partir de una palabra clave o frase.

1. Carga el resultado de una búsqueda.

   Al dar clic en este botón se muestra una ventana gráfica de clase diálogo especializada en este tipo de proceso. Así, se le proporciona al usuario una mejor panorámica debido a que se le simplifica la cantidad de posibles opciones en cuanto a tipos de ficheros en un directorio dado. Es decir, sólo muestra los ficheros que tienen la extensión dada, así como los subdirectorios que allí se encuentran.

2. Salva el resultado de una búsqueda.

Al dar clic en este botón se muestra una ventana gráfica de clase diálogo especializada en este tipo de proceso. Para salvar un resultado de búsqueda sólo hay que especificar la dirección de un directorio donde se va a guardar esta consulta o simplemente seleccionar algún fichero correspondiente a resultados de búsquedas guardados. En este último caso se sobrescriben los datos previamente guardados en este fichero. En el caso contrario, si seleccionó la dirección de un directorio, dentro de éste se creará un fichero cuyo nombre está compuesto por las palabras que generaron la búsqueda, seguidas de .ddd.

3. Quita del recuadro de resultados la búsqueda que tenga la etiqueta seleccionada. Es decir, elimina la búsqueda actual en la que se está trabajando.
4. Área de texto donde se introduce la palabra o frase a buscar dentro de los artículos científicos.

   GARLucene permite los siguientes tipos de consultas:

   - Palabra

     Combinación de caracteres ejemplo “Hello”. Donde no existe diferencia ninguna entre las mayúsculas y las minúsculas, siendo igual para el sistema “Hello”, “hello”, “HELLO”.

   - Frase

     Combinación de palabras encerradas entre comillas ejemplo “Hello Dolly”.

   - Apoyada por comodines de textos

     Se pueden usar en cualquier lugar de la consulta exceptuando el primer carácter de ésta:

     - ✓ “?” significa un carácter en una frase o palabra incluyendo el carácter vacio.

       Ejemplo: te?t devolvería los artículos que contienen “text” o “test”

     - ✓ “*”significa varios caracteres en una frase o palabra.

       Ejemplo: test* los artículos que contienen “tests” o “tester”

- Borrosas

  Basadas en la distancias Levenshtein o la distancia Edit. Es necesario usar "~" como símbolo al final de una frase o palabra:

  Por ejemplo: para buscar todas las palabras similares con "roam" bastaría con poner roam~ y el sistema devolvería los artículos que contienen "foam" o "roams".

  Por defecto el sistema realiza las búsquedas borrosas con un umbral igual a 0.5. Este valor se puede cambiar al realizar una consulta sólo con poner un valor entre 0 y 1 seguido del símbolo. Por ejemplo roam~0.8 buscaría todas las palabras similares a "roam" pero que el grado de similitud sea mayor o igual a 0.8. Si el usuario pone el valor uno el sistema buscaría todas las palabras que sean igual a "roam".

- Por proximidad

  Es posible buscar palabras que están a una distancia especificada, distancia medida en número de palabras entre ellas. Para esto utilice el símbolo "~" al final de la frase. Por ejemplo:

  "jakarta apache"~10 devuelve aquellos documentos que tienen las palabras "apache" y "jakarta" separadas por 10 palabras entre ellas.

- Por rango

  Hace búsqueda entre dos rangos los cuales pueden estar incluidos o no. Por ejemplo:

  - ✓ [Aida TO Carmen] devuelve todos los documentos que contengan palabras entre Aida y Carmen incluyéndolas a ambas.
  - ✓ {Aida TO Carmen} devuelve todos los documentos que contengan palabras entre Aida y Carmen excluyéndolas a ambas.

- Fomentando un término

Dentro de una consulta se permite aumentar la relevancia que tiene una frase sobre otra. Por defecto este valor es igual a uno pero esto se puede cambiar añadiéndole el símbolo "^" seguido de un número, este número debe ser positivo. Por ejemplo si estamos realizando una búsqueda por "jakarta apache" y queremos que jakarta tenga más relevancia entonces cambiamos la consulta por:

jakarta^2 apache haciendo que jakarta tuviera más relevancia que apache.

- Operadores Booleanos

  El sistema permite realizar búsquedas auxiliándonos de los operadores booleanos para la conjunción de palabras o frases. Por defecto se utiliza el operador OR, hasta que el usuario no lo cambie en una consulta. Entonces la consulta jakarta apache es equivalente a jakarta OR apache.

  - ✓ OR

    Busca los documentos que tienen una frase o la otra.

    Ejemplo: "jakarta apache" OR jakarta

  - ✓ AND

    Busca los documentos que tienen ambas frases.

    Ejemplo: "jakarta apache" AND "jakarta"

  - ✓ +

    Busca los documentos que tienen la frase que sigue al símbolo y puedan contener la otra frase. Ejemplo:

    - + "jakarta" "apache"

  - ✓ NOT

    Buscan los documentos que no contienen la frase que sigue al símbolo. Ejemplo:

    - "jakarta apache" NOT "Apache Lucene"

Este operador no puede ser usado cuando solo existe un término. Ejemplo:

- ~~NOT "Apache Lucene"~~

✓ -

Buscan los documentos que estrictamente no contienen la frase que sigue al símbolo. Ejemplo:

- "jakarta apache" -"Apache Lucene"

- Agrupamiento

  El sistema se auxilia de los paréntesis para formar grupos de consultas. Por ejemplo:

  ✓ (jakarta OR apache) AND website

- Carácter de escape

  El sistema consta del carácter de escape "\" para usar con la siguiente lista de caracteres en caso de sernos necesarios:

  +, -, &&, ||, !, ( ), {, }, [,], ^, ", ~, *, ?, :, \

  Por ejemplo, si se quisiera ver los documentos que contienen la expresión matemática "(1+1):2" entonces el usuario debe poner "\(1\+1\)\:2" para obtener los resultados deseados.

- Cualquier combinación de las opciones anteriores.

5. Hace que el sistema muestre los resultados obtenidos a partir de la realización de una búsqueda y recuperación sobre los documentos previamente indexados. Adicionando estos al recuadro encargado de mostrar los resultados e identificándolos con una etiqueta que contiene la palabra o frase buscada.
6. Recuadro contenedor de los resultados obtenidos. Cada vez que se el usuario ejecuta una búsqueda el sistema le agrega un marco identificado por una etiqueta que contiene el texto con que se originó la búsqueda. Observar más detalles en la Figura 3.4.2.

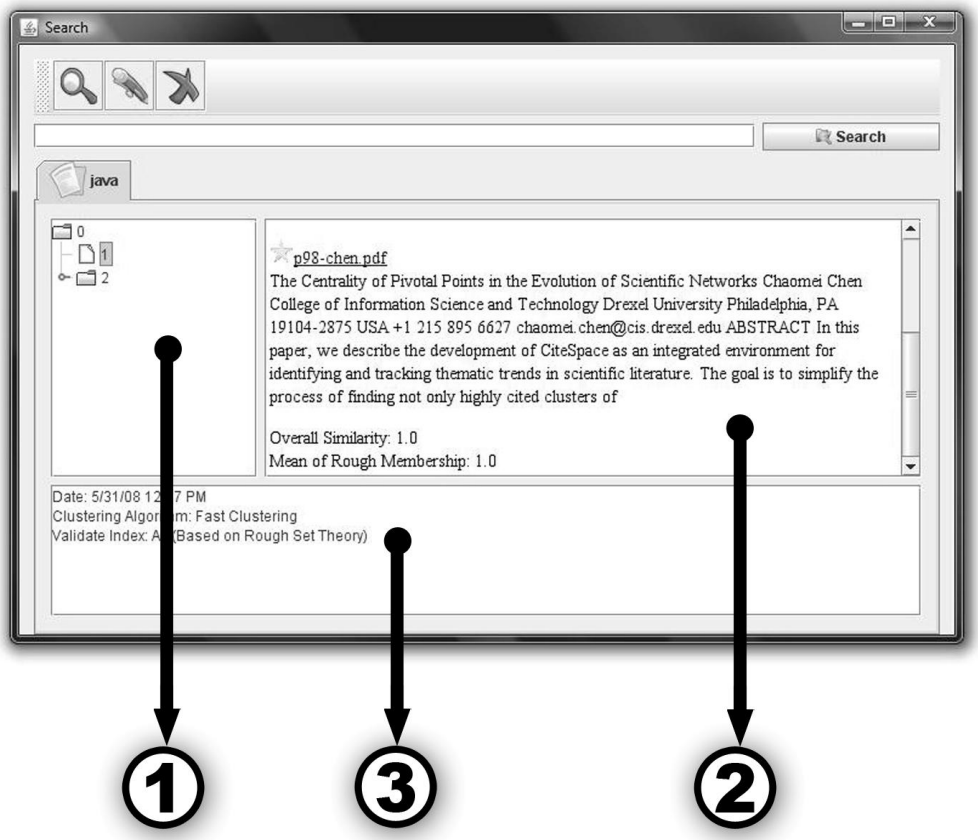

Figura 3.4.2. Recuadro contenedor de los resultados obtenidos.

1. Área donde se muestran los grupos formados por el sistema GARLucene.

   En esta área el sistema GARLucene muestra los grupos en forma de árbol jerárquico comenzando por la raíz, formada por todos los artículos científicos recuperados que guardan relación con la palabra o frase que originó la búsqueda. Cada grupo que el sistema dividió en varios grupos, es decir, que no es una hoja de la jerarquía generada, constituye una carpeta que se hace corresponder con un nodo del árbol que puede expandirse ( ) o contraerse ( ) dando doble clic sobre ella o simplemente

dando clic en el símbolo que identifica su estado. Si el grupo no fue dividido el sistema lo representa como una hoja de texto vacía.

Si el usuario da clic sobre cualquier grupo lo está seleccionando y automáticamente el contenido de este grupo se visualiza en el recuadro de texto encargado de mostrar el contenido de los grupos.

Si se quisiera evaluar la calidad del corte de la jerarquía; es decir, si quisiéramos evaluar la calidad del corte seleccionado por el usuario en esta jerarquía, bastaría dar clic secundario sobre cualquier parte de este marco y luego hacer clic sobre la opción del menú que se nos muestra. Así, se evaluará la calidad del corte seleccionado, asumiendo que un grupo a evaluar es aquel que constituye una hoja en el corte realizado por el usuario, aunque éste pudiera ser en otro momento expandido. Los resultados de la evaluación pasarán al recuadro de texto encargado de mostrar la información general de la búsqueda.

2. Se encarga de mostrar el contenido de los grupos.

   Para esto el sistema GARLucene muestra una lista de los documentos contenidos en el grupo mostrando primero el nombre físico del documento en forma de enlace hacia el documento original, seguido de una síntesis del documento.

   El sistema también señala los documentos más representativos del grupo identificándolos con un icono de estrella antes del nombre del documento.

   Al finalizar, nos muestra el nombre de las medidas con que fueron evaluados los grupos así como su valor. Estas medidas le brindan al usuario una idea de la calidad del grupo.

3. Muestra información general.

   Inicialmente contiene información como:

   - La fecha y hora en que se originó la consulta.
   - El algoritmo de agrupamiento con que se obtuvieron los grupos.
   - La opción de validación de agrupamiento con que fueron validados los grupos.

También puede contener la información relacionada con la evaluación realizada a cortes en la jerarquía especificados por el usuario. Esta información no es persistente si se salva esta consulta.

3.4.2 Directorio de índices

Al desplegar el menú del botón Index y seleccionar la opción Directory de la Figura 3.3.2, se muestra una ventana gráfica con la cual podemos interactuar y efectuar cambios en los índices de directorio. Observar Figura 3.4.2.

Hay que tener en cuenta que no solo podemos tener índices en la computadora donde estemos trabajando, es posible utilizar índices desde otra computadora en la Red. Para esto solo hay que tener permiso de lectura si queremos recuperar información, permiso de escritura en si queremos indexar artículos hacia ese índice, o lectura/escritura en caso de querer hacer ambas operaciones. Los permisos son los mismos que existen en el Sistema Operativo vigente en su computadora. Así, le recomendamos probar primero desde su Sistema Operativo si tiene los permisos necesarios según sus necesidades con los directorios de índices.

El sistema GARLucene permite que el usuario tenga más de un directorio de índice para realizar el proceso de búsqueda y recuperación. Además, solo permite que el usuario tenga seleccionado un índice para el proceso de indexado de los artículos científicos y este índice debe pertenecer a la lista de índices sobre los cuales se le pueda realizar el proceso de búsqueda y recuperación.

Cualquier cambio realizado en esta ventana del sistema es salvado automáticamente por lo que se recomienda ser cuidadoso al realizar alguna modificación. Observar Figura 3.4.3.

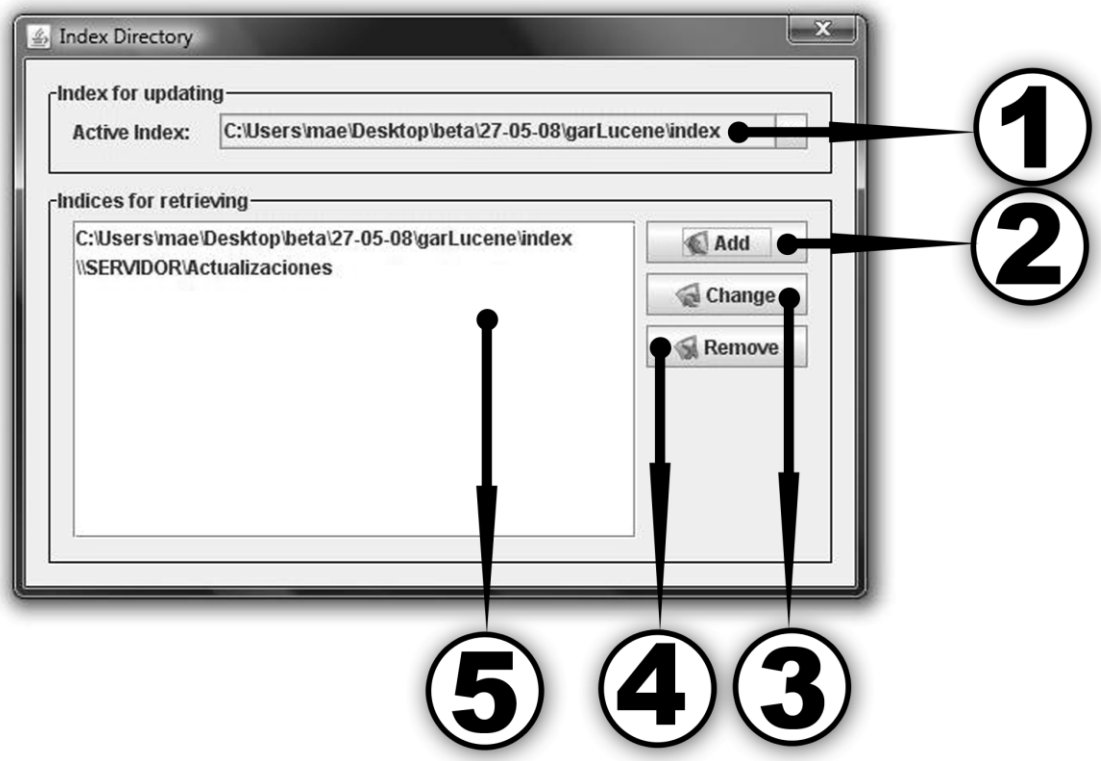

Figura 3.4.3. Directorio de índices.

1. Índice sobre el cual queremos indexar los artículos científicos.
2. Adiciona a la lista de directorios de índices un directorio seleccionado por el usuario para que el sistema pueda hacer el proceso de búsqueda y recuperación.

   Para esto muestra una ventana de tipo diálogo que permite seleccionar un directorio.
3. Cambia la dirección del índice seleccionado en la lista de directorios de índices auxiliándose del botón 2.
4. Borra el índice seleccionado en la lista de directorios de índices.
5. Lista de directorios de índices.

3.4.3 Indexación de los documentos científicos

Al desplegar el menú del botón Index y seleccionar la opción Indexing de la Figura 3.3.2 se muestra una ventana grafica con la cual podemos interactuar y actualizar el índice de escritura con nuevos artículos científicos.

El sistema GARLucene admite diferentes tipos de archivos para indexar (Ms Word, Ms Excel, Ms PowerPoint, RTF, PDF, XML, HTML, TXT, Open Office, ZIP).

También se guarda el estado de la ventana hasta que se cierra el sistema. Es decir, aunque cerremos la ventana no se pierden los valores establecidos por el usuario. Observar Figura 3.4.4.

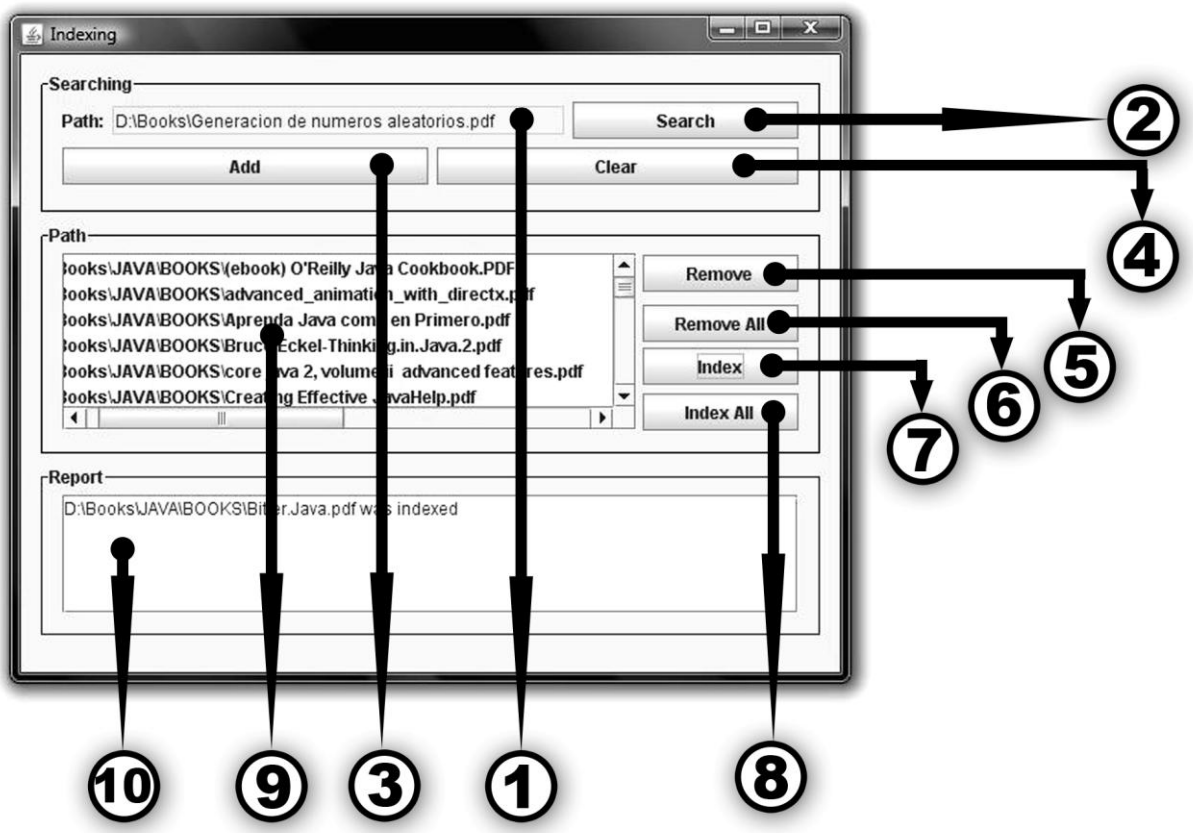

Figura 3.4.4. Indexación de los artículos científicos.

1. Ruta completa del artículo o de los artículos científicos seleccionados en 2.

   En caso de ser más de un artículo el sistema GARLucene pone cada ruta separada de la otra por la cadena " #@# " (espacio, número, arroba, número, espacio).

2. Muestra una ventana gráfica del tipo diálogo donde el usuario puede seleccionar uno o varios archivos para indexar.

   El sistema se encarga de mostrar solo los tipos de archivos que son soportados por éste. Además, se le brinda al usuario la oportunidad de seleccionar directorios donde el sistema se encargaría de extraer los archivos soportados por él y adicionarlos en el cuadro de texto que contiene las direcciones de los archivos seleccionados en la búsqueda.

3. Adiciona los archivos que aparecen en el cuadro de texto que contiene las direcciones de aquellos seleccionados en la búsqueda, a otro cuadro que contiene la lista de archivos ya seleccionados. El usuario puede indexar los archivos adicionados si lo desea.

4. Quita todas las direcciones de archivos que aparezcan en el cuadro de texto que contienen las direcciones de los archivos seleccionados en la búsqueda.

5. Elimina el archivo seleccionado en el cuadro de texto que contiene todas las direcciones de archivos previamente adicionadas por el usuario.

6. Elimina todas las direcciones de archivos que aparezcan en el cuadro de texto que contiene todas las direcciones de archivos previamente adicionadas por el usuario.

7. Indexa el archivo seleccionado hacia el índice de escritura.

8. Indexa todos los archivos seleccionados hacia el índice de escritura.

9. Contiene todas las direcciones de archivos previamente adicionados por el usuario para su posible indexación.

10. Contiene todos los reportes de los archivos que el usuario mandó a indexar, mostrando las direcciones de los archivos más una descripción sobre el resultado del proceso de indexado. Especifica adicionalmente si el archivo fue indexado o no.

3.4.4 Configuración general de GARLucene

El sistema GARLucene consta con la ventana gráfica de configuración para que el usuario pueda configurar los aspectos generales del software. Observar Figura 3.4.5.

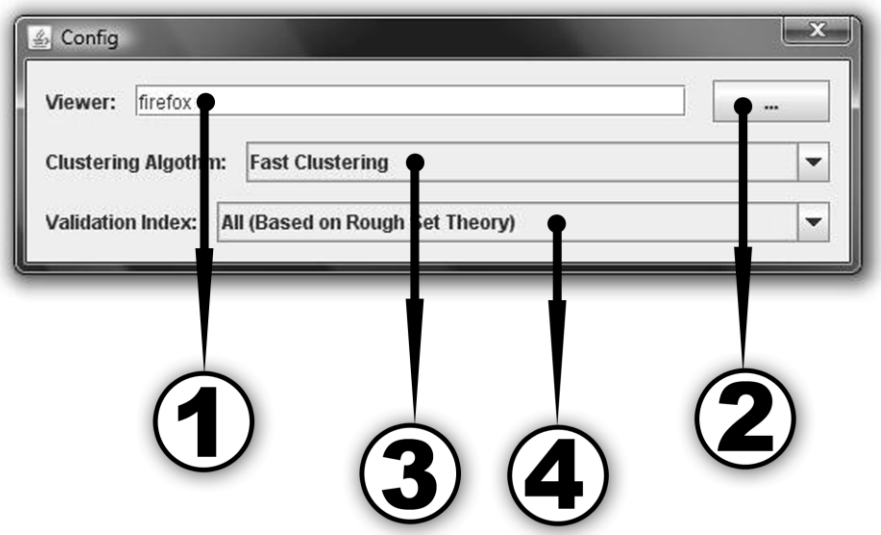

Figura 3.4.5. Configuración general del sistema.

1. Ruta de algún ejecutable externo con que el usuario desea ver el artículo científico. Debido a que el sistema GARLucene permite la indexación de diferentes tipos de archivos, incluyendo *.html, se recomienda usar algún software que permita visualizar diferentes tipos de archivos. Sugerimos, por ejemplo: IExplorer, Mozilla Firefox y Netscape. El sistema GARLucene pasa esta dirección al Sistema Operativo para que éste ejecute el archivo al cual hace referencia esta dirección. Si el Sistema Operativo tiene en su variable de entorno la dirección donde se encuentra el archivo, entonces es suficiente poner el nombre del ejecutable.
2. Permite establecer la ruta del visor a utilizar.
3. Algoritmo seleccionado por el usuario para llevar a cabo el agrupamiento de los artículos científicos. GARLucene brinda la posibilidad de escoger qué método de

agrupamiento se desea aplicar a la colección recuperada Fast Clustering, GN Betweenness o GN Betweenness & Similarity.

4. Opciones de evaluación con que el usuario desea que el sistema GARLucene evalúe la calidad de los grupos formados. El sistema consta de tres opciones: Overall Similarity & Modularity, Based on Rough Set Theory y All (Based on Rough Set Theory). Observe en la Tabla 3.4.1 la relación que existe entre las opciones que ofrece el sistema y las formas de evaluación locales y globales de los agrupamientos realizados.

Tabla 3.4.1. Opciones de evaluación local y global

Opción del Sistema	Evaluación local de grupo	Evaluación global del agrupamiento correspondiente al corte realizado por el usuario
Overall Similarity & Modularity	Overall Similarity	Modularity
Based on Rough Set Theory	Mean of Rough Membership	Rough Accuracy Rough Quality Rough F-measure
All (Based on Rough Set Theory)	Overall Similarity Mean of Rough Membership	Modularity Rough Accuracy Rough Quality Rough F-measure

3.5 Ayuda del sistema GARLucene

Brinda ayuda a los usuarios en la manipulación de GARLucene. Las principales opciones aquí descritas fueron plasmadas en la ayuda, así como sugerencias de cómo dirigir el trabajo con el sistema.

Conclusiones

Como resultado de esta investigación se diseñó e implementó el sistema GARLucene para la gestión de información científica, a partir de los resultados de la recuperación de artículos científicos usando LIUS. Con esto se agrupa los documentos a partir de las propiedades estructurales de las representaciones gráficas, cumpliéndose de esta forma el objetivo general planteado, ya que:

- ✓ El marco de trabajo LIUS y el API Lucene fueron exitosamente utilizados para la creación de índices, recuperación de información y procesamiento textual. GARLucene explotó las facilidades de LIUS y Lucene, por estar el código disponible y tener un diseño extensible.
- ✓ Los métodos de agrupamiento basados en la intermediación de aristas fueron implementados en GARLucene. Éstos posibilitan organizar los resultados de procesos de recuperación de información y por tanto contribuir a una mejor gestión de los artículos científicos que los usuarios desean analizar. Específicamente, se implementó el algoritmo clásico GN y una variante modificada que considera la media de las similitudes. Adicionalmente, el algoritmo Fast clustering fue incluido a partir de estudios empíricos con colecciones textuales.
- ✓ La aplicación de la teoría de los conjuntos aproximados permitió caracterizar los resultados de los agrupamientos, permitió la verificación de los grupos y agrupamientos en general, determinó los documentos más representativos por grupos. Adicionalmente se utilizó la medida Modularity para evaluar dendrogramas, así como Overall Similarity se incluyó en la evaluación de los grupos. Además las medidas basadas en RST y Overall Similarity se utilizaron para realizar cortes en la jerarquía a construir.
- ✓ El diseño e implementación del sistema GARLucene es extensible y permite su reutilización. El módulo cluster.jar permite utilizar los principales algoritmos de GARLucene en otras aplicaciones. El procesamiento de los documentos se realiza en un tiempo razonable y asequible a los usuarios.

Recomendaciones

Teniendo en consideración que el sistema propuesto es extensible se recomienda:

- ✓ Incorporar otros métodos basados en el cálculo de la intermediación, de forma tal que se reduzca la complejidad computacional en el proceso de agrupamiento y se obtengan menos grupos aislados.
- ✓ Incorporar otras formas de validación diseñadas especialmente para valorar resultados de agrupamientos sobre representaciones gráficas.
- ✓ Mejorar la interacción de GARLucene con LIUS y Lucene en el proceso de recuperación de información para que el sistema consuma menos memoria.
- ✓ Variar las denominaciones en la configuración del software para lograr una mejor comprensión de los mismos.

Referencias bibliográficas

Agarwal, P. K. and Mustafa, N. H. (2004) k-means projective clustering. In PODS 2004. ACM Press, Paris, France, pp. 155-165.

Aggarwal, C. C. and Yu, P. S. (2005) Online analysis of community evolution in data streams. In Proceedings of SIAM International Data Mining Conference.

Agrawal, R., Gehrke, J., Gunopulos, D. and Raghavan, P. (1998) Automatic subspace clustering of high dimensional data for data mining applications. In International Conference on Management of Data. Vol. 27 ACM Press, Seattle, WA, USA, pp. 94-105.

Akaike, H. (1974) A new look at the statistical model identification. IEEE Trans. Automat. Control, 19: 716-723.

Anderberg, M. R. (1973) Clustering Analysis for Applications, New York: Academic.

Ankerst, M., Breunig, M. M., Kriegel, H.-P. and Snader, J. (1996) OPTICS: Ordering points to identify the clustering structure. In International Conference on Management of Data. Vol. 28 ACM Press, Philadelphia, PA, USA, pp. 49-60.

Anthonisse, J. M. (1971) The rush in a directed graph. Stichting Mathematicsh Centrum, Amsterdam.

Arco, L., Bello, R. and Artiles, M. (2006a) New clustering validity measures based on rough set theory. In Proceedings of International Symposium on Fuzzy and Rough Sets (ISFUROS'06). (Eds, Falcón, R. and Bello, R.) Santa Clara, Cuba.

Arco, L., Bello, R. and Artiles, M. (2006b) Un nuevo enfoque del uso de los conjuntos aproximados en la solución de problemas de la minería de textos. In VII Conferencia Científica Internacional de la Universidad de Ciego de Ávila (UNICA2006). Ciego de Ávila, Cuba.

Arco, L., Bello, R. and García, M. M. (2006c) On clustering validity measures and the rough set theory. In Proceedings of the Fifth Mexican International Conference on Artificial Intelligence (MICAI'06). IEEE Computer Society, Apizaco, México, pp. 168-177.

Arco, L., Bello, R., Mederos, J. M. and Pérez, Y. (2006d) Agrupamiento de documentos textuales mediante métodos concatenados. Revista Iberoamericana de Inteligencia Artificial, 10(30): 43-53.

Aslam, J., Pelekhov, K. and Rus, D. (1998) Static and dynamic information organization with star clusters. In Proceedings of the Conference of Information Knowledge Management. Baltimore.

Backer, F. B. and Hubert, L. J. (1976) A graph-theoretic approach to goodness-of-fit in complete-link hierarchical clustering. Journal of the American Statistical Association, 71: 870-878.

Batchelor, B. (1978) Pattern Recognition: Idead in Practice, Plenum Press, New York.

Baumes, J., Goldberg, A. and Magdon-Ismail, M. (2005) In Intelligence and Security Informatics, Vol. 3495/2005 Springer Berlin / Heidelberg, Berlin, pp. 27-36.

Bavelas, A. (1948) A mathematical model for group structures. Human Organization, 7: 16-30.

Bazan, J., Nguyen, H. S. and Szczuka, M. (2004) A view on rough set concept approximations. Fundamenta Informatica, 59(2-3): 107-118.

Berry, M. W. (2004) Survey of Text mining: Clustering, Classification, and Retrieval, Springer Verlag, New York, NY, USA.

Bezdek, J. and Pal, N. (1995) Cluster validation with generalized Dunn's indices. In Proceedings of the 2nd International two-stream Conference on ANNES. (Eds, Kasabov, N. and Coghill, G.) IEEE Press, Piscataway, NJ, pp. 190-193.

Blair, D. (1992) Information retrieval and the philosophy of language. The Computer Journal, 35(3): 200-207.

Bock, H. (1985) On significance tests in cluster analysis. J. Classification, 2: 77-108.

Bolelli, L., Ertekin, S., Zhou, D. and Giles, C. L. (2007) A clustering method for web data with multi-type interrelated components. In WWW '07: Proceedings of the 16th international conference on World Wide Web. ACM Press, Banff, Alberta, Canada.

Boley, D. (1998) Principal Direction Divisive Partitioning. Data Mining and Knowledge Discovery, 2(4): 325-344.

Bordes, A., Ertekin, S., Weston, J. and Bottou, L. (2005) Fast kernel classifiers with online and active learning. Journal of Machine Learning Research, 6: 1579-1619.

Borgatti, S. P. and Everett, M. G. (2005) A graph-theoretic perspective on centrality. Social Networks.

Bradley, P. S., Fayyad, U. and Reina, C. (1998) Scaling clustering algorithms to large databases. In 4th International Conference on Knowledge Discovery and Data Mining. AAAI Press, New York, NY, USA, pp. 9-15.

Brun, M., Sima, C., Hua, J., Lowey, J., Carroll, B., Suh, E. and Dougherty, E. R. (2007) Model-based evaluation of clustering validation measures. Pattern Recognition, 40: 807-824.

Bueno, E. (1999) Gestión del conocimiento y capital intelectual: análisis de experiencias en la empresa española. In Actas del X Congreso AECA. Zaragoza, España.

Bueno, E. (2001) Estado del arte y tendencias en creación y gestión del conocimiento. In IBERGECYT 2001. Congreso Iberoamericano de Gestión del Conocimiento y la Tecnología. La Habana, Cuba.

Caballero, Y. (2007) Aplicación de la teoría de los conjuntos aproximados en el proprocesamiento de los conjuntos de entrenamiento para algoritmos de aprendizaje. In Departamento de Ciencia de la Computación. Vol. Doctor en Ciencias Técnicas Universidad Central "Marta Abreu" de Las Villas, Santa Clara, Villa Clara.

Caballero, Y., Arco, L., Bello, R. and Marx-Gómez, J. (2007a) New measures for evaluationg decision systems using rough set theory: the application in seadonal weather forecasting. In Proceedings of the Third International ICSC Symposium on Information Technologies in Environmental Engineering (ITEE'07). (Eds, Marx-Gómez, J., Sonnenschein, M., Müller, M., Welsch, H. and Rautenstrauch, C.) Springer Verlag, Carl von Ossietzky Universität Oldenburg, Alemania, pp. 161-174.

Caballero, Y., Arco, L., Bello, R., Salgado, Y., Márquez, Y., León, P. and Álvarez, D. (2007b) Nuevas medidas de la teoría de los conjuntos aproximados para la evaluación de sistemas de información en Bioinformática. In II Congreso Internacional de

Bioinformática y Neuroinformática. XII Convención y Expo Internacional Informática'07. La Habana, Cuba.

Calinski, R. B. and Arabas, J. (1974) A dendrite method for cluster analysis. Comm. in Statistics, 3: 1-27.

Canals, A. (2003) Gestión del Conocimiento.

Cavnar, W. B. and Trenkle, J. M. (1994) N-gram based text categorizarion. In Proceedings of the Symposium on Document Analysisi and Information Retrieval. Las Vegas, pp. 161-175.

Chee, B. and Schatz, B. (2007) Document clustering using small world communities. In Proceedings of Joint Conference on Digital Libraries. pp. 53-62.

Chen, K. and Liu, L. (2004) ClusterMap: labeling clusters in large datasets via visualization. In Proceedings of the ACM/IEEE 13th Conference on Information and Knowledge Management CIKM'04. Washington, D.C., pp. 285-293.

Cheng, D., Kannan, R., Vempala, S. and Wang, G. (2006) A divide-and-merge methodology for clustering. ACM Trans. Database Syst., 31(4): 1499-1525.

Cheng, D., Vempala, S., Kannan, R. and Wang, G. (2005) A divide-and-merge methodology for clustering. In 24th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems. ACM Press, Baltimore, Maryland, pp. 196-205.

Choo, C. W., Detlor, B. and Turnbull, D. (2000) Web Work: Information Seeking and Knowledge Work on the World Wide Web, Klumer Academic Publishers.

Clauset, A., Newman, M. E. J. and Moore, C. (2004) Finding community structure in very large networks. In Physical Review E, Statistical, nonlinear, and soft matter physics. Vol. 70 (2), pp. 066111.1-066111.6.

Cortes, C., Pregibon, D. and Volinsky, C. (2001) Communities of interest. In Proceedings of the 4th International Conference on Advances in Intelligent Data Analysis. pp. 105-114.

Cutting, D. R., Karger, D. R. and Pederson, J. O. (1993) Constant interaction-time Scatter/Gather browsing of very large document collections. In Proceedings of 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Pittsburgh, Pennsylvania, United States, pp. 126-134.

Dash, M. and Liu, H. (1997) Feature selection for classification. Intelligent Data Analysis, 1(3).

Dave, R. N. (1996) Validating fuzzy partitions obtained through c-shells clustering. Pattern Recognition Letters, 17: 613-623.

Davenport, T. and Prusak, O. (Eds.) (1997) Knowledge management glossary information ecology, Oxford University Press, Oxford.

Davies, D. L. and Bouldin, D. W. (1979) A cluster separation measure. IEEE Transactions on Pattern Analysis and Machine Learning, 1(4): 224-227.

Deodhar, M. and Ghosh, J. (2007) A framework for simultaneous co-clustering and learning from complex data. In KDD '07: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM Press, San Jose, California, USA, pp. 250-259.

Diday, E. (1974) Recent prograss in distance and similarity measures in pattern recognition. In Second International Joint Conference on Pattern Recognition. pp. 534-539.

Dixon, M. (1997) An overview of document mining technology. http://www.geocities.com/ResearchTriangle/Thinktank/1997/mark/writings/dixm97_dm.ps.

Donetti, L. and Muñoz, M. A. (2004) Detecting network communities: a new systematic and efficient algorithm. Journal of Statistical Mechanics, P10012.

Dourisboure, Y., Geraci, F. and Pellegrini, M. (2007) Extraction and classification of dense communities in the web. In WWW '07: Proceedings of the 16th international conference on World Wide Web. ACM Press, Banff, Alberta, Canada, pp. 461-470.

Duch, W. (2002) Similarity-based methods: a general framework for classification. Control and Cybernetics, 29(4): 937-968.

Dürsteler, J. C. (2001) Minería de Textos. La revista digital de InfoVis.net, 27.

Dunn, J. (1974) A fuzzy relative isodata process and its use in detecting compact well-separated clusters. J. Cybernetics, 3: 32-57.

Epter, S. and Krishnamoorthy, M. (1999) A multiple-resolution method for edge-centric data clustering. In Proceedings of CIKM 1999 International Conference on Information and Knowledge Management. ACM Press, Kansas City, Missouri USA, pp. 491-498.

Epter, S., Krishnamoorthy, M. and Zaki, M. (1999) Clusterability detection and initial seed selection in large data sets. Rensselaer Polytechnic Institute, Computer Science Dept., Troy, NY 12180.

Ester, M., Ge, R., Gao, B. J., Hu, Z. and Ben-Moshe, B. (2006) Joint cluster analysis of attribute data and relationship data: the connected k-center problem. SDM: 246-257.

Ester, M., Kriegel, H.-P., Sander, J. and Xu, X. (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In 2nd International Conference on Knowledge Discovery and Data Mining. AAAI Press, Portland, OR, USA.

Falkowski, T., Bartelheimer, J. and Spiliopoulou, M. (2006) Mining and visualizing the evolution of subgroups in social networks. In WI '06: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence. IEEE Computer Society, Washington, DC, USA, pp. 52-58.

Fayyad, V. (1996a) Data mining and knowledge discovery in databases. In Communication of ACM. Vol. 39

Fayyad, V. (1996b) Mining scientific data. Communication of ACM, 39.

Ferrer, R. and Solé, R. V. (2001) The small wolrd of human language. Proc. R. Soc. Lond. B, 268(1482): 2261-2265.

Ferrer, R. and Solé, R. V. (2004) Patterns in syntactic dependency networks. Physical Review E, 69(5): 051915.

Fortunato, S., Freeman, L. C. and Menczer, F. (2006) Scale-free network growth by ranking. Physical Review Letters, 96(21): 218701.

Foundation, A. (2008) Apache Lucene. pp. Apache Lucene is a high-performance, full-featured text search engine library written entirely in Java. It is a technology suitable for nearly any application that requires full-text search, especially cross-platform.

Frakes, W. B. and Baeza-Yates, R. (1992) Information Retrieval. Data Structure & Algorithms, Prentice Hall, New York.

Freeman, L. C. (1977) A set of measures of centrality based upon betweenness. Sociometry, 40: 35-41.

Freeman, L. C. (1979) Centrality in social networks: I. Conceptual clarification. Social Networks, 1: 215-239.

Fürnkranz, J., Scheffer, T. and Spiliopoulou, M. (2006) Knowledge discovery in databases. In 10th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2006). Springer, Berlin, Germany.

Gaber, M. M. and Yu, P. S. (2006) A framework for resource-aware knowledge discovery in data streams: a holistic approach with its application to clustering. In SAC '06: Proceedings of the 2006 ACM Symposium on Applied Computing. ACM Press, Dijon, France, pp. 649-656.

Gao, B., Liu, T.-Y., Zheng, X., Cheng, Q.-S. and Ma, W.-Y. (2005) Consistent bipartite graph co-partitioning for star-structured high-order heterogeneous data co-clustering. In 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining. ACM Press, Chicago, Illinois, USA, pp. 41-50.

García, M. M. (1999) Monografía de reconocimiento de patrones. Universidad Central "Marta Abreu" de Las Villas, Santa Clara, Villa Clara.

Getoor, L. and Diehl, C. P. (2005) Link mining: a survey. SIGKDD Explor. Newsl., 7(2): 3-12.

Gibson, D., Kleinberg, J. M. and Raghavan, P. (1998) Clustering categoricl data: an approach based on dynamical systems. In 24th International Conference on Very Large Data Bases. Morgan Kaufmann, New York, NY, USA, pp. 311-322.

Gil-García, R., Badía-Contelles, J. M. and Pons-Porrata, A. (2003) Extended Star clustering algorithm. In Proceedings of CIARP. pp. 480-487.

Gil-García, R. J., Badía-Contelles, J. M. and Pons-Porrata, A. (2006) A general framework for agglomerative hierarchical clustering algorithms. In Proceedings of 18th International Conference on Pattern Recognition (ICPR'06). Vol. 2, pp. 569-572.

Girvan, M. and Newman, M. E. J. (2002a) Community structure in social and biological networks. PNAS Proc. National Academy of Sciences, 99: 7821-7826.

Girvan, M. and Newman, M. E. J. (2002b) Community structure in social and biological networks. PNAS Proc. National Academy of Science USA, 99(12): 7821-7826.

Glover, E., Pennock, D., Lawrence, S. and Krovetz, R. (2002) Inferring hierarchical descriptions. In Proceedings of the ACM CIKM International Conference on Information and Knowledge Management. Springer-Verlag, McLean, VA, USA, pp. 507-514.

Goodman, L. and Kruskal, W. (1954) Measures of associations for cross-validations. J. Am. Stat. Assoc., 49: 732-764.

Gotlieb, G. C. and Kumar, S. (1968) Semantic clustering of index terms. Journal of the ACM (JACM), 15(4).

Gower, J. C. and Ross, G. J. S. (1969) Minimum spanning trees and single-linkage cluster analysis. Applied Statistics, 18: 54-64.

Grau, A. (2007) Herramientas de Gestión del Conocimiento.

Guha, S., Rastogi, R. and Shim, K. (1998) CURE: An efficient clustering algorithm for large databases. In International Conference on Management of Data. ACM Press, Seattle, WA, USA.

Halkidi, M., Batistakis, Y. and Vazirgiannis, M. (2001a) Clustering algorithms and validity measures. In Proceedings of the 13th International Conference on Scientific and Statistical Database Management. IEEE Computer Society, pp. 3-22.

Halkidi, M., Batistakis, Y. and Vazirgiannis, M. (2001b) On clustering validation techniques. Journal of Intelligent Information Systems, 17(2/3): 107-145.

Halkidi, M., Batistakis, Y. and Vazirgiannis, M. (2002) Clustering validity checking methods: Part II. ACM SIGMOD Record, 31(3): 19-27.

Halkidi, M., Vazirgiannis, M. and Batistakis, Y. (2000) Quality scheme assessment in the clustering process. In Proceedings of PKDD. Lyon, France.

Han, J. and Kamber, M. (2001) Data mining: concepts and techniques.

Hand, D. J. (1981) Discrimination and classification, John Wiley and Sons.

Hansen, C. D. and Johnson, C. R. (Eds.) (2005) The Visualization handbook, Elsevier Academic press.

Hasegawa, T., Sekine, S. and Grishman, R. (2004) Discovering relations among named entities from large corpora. In Proceeding of ACL-2004. pp. 415-422.

Hinneburg, A. and Keim, D. A. (1998) An efficient approach to clustering in large multimedia databases with noise. In 4th International Conference on Knowledge Discovery and Data Mining. AAAI Press, New York, NY, USA, pp. 58-65.

Hochbaum, D. S. and Shmoys, D. (1985) A best possible heuristic for the k-center problem. Mathematics of Operations Research, 10(2): 180-184.

Holme, P. (2002) Edge overload breakdown in evolving networks. Physical Review E, Statistical, nonlinear, and soft matter physics, 66 (2A)(3): 036119.1-036119.7.

Holsheimer, M. and Siebes, A. (1994) Data Mining. In The search for knowledge in databases. Amsterdam.

Höppner, F., Klawonn, F., Kruse, R. and Runkler, T. (1999) Fuzzy cluster analysis: methods for classification, data analysis and image recognition., John Wiley & Sons Ltd., West Sussex, England.

Hu, X. and Wu, D. C. (2007) Data Mining and Predictive Modeling of Biomolecular Network from Biomedical Literature Databases. IEEE/ACM Trans. Comput. Biol. Bioinformatics, 4(2): 251-263.

Hubert, L. and Schultz, J. (1976) Quadratic asignment as a general data-analysis strategy. Br. J. Math. Stat. Psicol., 29: 190-241.

Jain, A. K. and Dubes, R. C. (1988) Algorithms for clustering data, Prentice Hall College Div, Englewood Cliffs, NJ.

Jain, A. K., Murty, M. N. and Flynn, P. J. (1999) Data clustering: a review. ACM Computing Surveys, 31(3): 264-323.

Jenssen, R., Hild, K. E., Erdogmus, D., Principe, J. C. and Eltoft, T. (2003) Clustering using Renyi's Entropy. In International Joint Conference on Neural networks. pp. 20-24.

Jickels, T. and Kondrak, G. (2006) Unsupervised labeling of noun clusters.

Joachims, T. (1997) Text categorization with support vector machines: learning with many relevant features.

Jonyer, I., Cook, D. J. and Holder, L. B. (2002) Graph-based hierarchical conceptual clustering. Journal of Machine Learning Research, 2: 19-43.

Kalisky, T., Sreenivasan, S., Braunstein, L. A., Buldyrev, S. V., Havlin, S. and Stanley, H. E. (2006) Scale-free networks emerging from weighted random graphs. Physical Review E, 73(025103).

Karypis, G., Han, E.-H. and Kumar, V. (1999) CHAMELEON: a hierarchical clustering algorithm using dynamic modeling. IEEE Computer, 32(8): 68-75.

Kaufman, L. and Rousseeuw, P. J. (1990) Finding groups in data: an introduction to cluster analysis, John Wiley and Sons.

Komorowski, J., Pawlak, Z. and Polkowski, L. (1999) In Rough-Fuzzy Hybridization: A New Trend in Decision Making(Eds, Pal, S. K. and Skowron, A.) Springer-Verlag, Singapore, pp. 3-98.

Kriegel, H.-P. and Pfeifle, M. (2005) Density-based clustering of uncertain data. In KDD '05: Proceeding of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining. ACM Press, Chicago, Illinois, USA.

Kruse, R., Döring, C. and Lesor, M.-J. (2007) In Advances in Fuzzy Clustering and its Applications(Eds, Oliveira, J. V. d. and Pedrycz, W.) John Wiley and Sons, Est Sussex, England, pp. 3-27.

Lanquillon, C. (2001) Enhancing Text Classification to Improve Information Filtering. In Research Group Neural Networks and Fuzzy Systems. Vol. PhD. thesis University of Magdeburg "Otto von Guericke", Magdeburg, pp. 231.

Latora, V. and Marchiori, M. (2004) A measure of centrality based on the network efficiency.

Levine, E. and Domany, E. (2001) Resampling method for unsupervised estimation of cluster validity. 2001, 13(11): 2573-2593.

Lewis, D. D. (1992) Representation and learning in information retrieval. In Department of Computer and Information Science. Vol. PhD. thesis University of Massachasetts, Massachusetts, USA.

Lewis, D. D. and Ringuette, M. (1994) A comparison of two learning algorithms for text classification. In Third Annual Symposium on Document Analysis and Information Retrieval. University of Nevada, Las Vegas.

Lezcano, R. D. (2002) Minería de Datos.

Liu, Y., Cai, J., Yin, J. and Huang, Z. (2006) An efficient clustering algorithm for small text documents. In Seventh International Conference on Web-Age Information Management (WAIM 2006). IEEE Commnunications Society.

Manning, C. and Shütze, H. (2000) Foundations of Statistical Natural Language Processing, MIT Press.

Maulik, U. and Bandyopadhyay, S. (2002) Performance evaluation of some clustering algorithms and validity indices. IEEE Trans. Pattern Anal Mach Intell, 24(12): 1650-1654.

McQueen, J. (1967) Some methods for classification and analysis of multivariate observations. In 5th Berkeley Symposium on Mathematics.

Medina, J. E. and Pérez, A. (2007) ACONS: a new algorithm for clustering. In CIARP.

Michalski, R. S., Stepp, R. E. and Diday, E. (1981) A recent advance in data analysis: clustering objects into classes characterized by conjuntive concepts. Progress in Pattern Recognition, 1: 33-56.

Milligan, G. W. and Cooper, M. C. (1985) An examination of procedures for determining the number of clusters in a data set. Psychometrika, 50: 159-179.

Müller, R. M., Spiliopoulou, M. and Lenz, H.-J. (2002) Electronic marketplaces of knowledge: Characteristics and sharing of knowledge assets. In Proceedings of the International Conference on Advances in Infrastructure for e-Business. Italy.

Müller, R. M., Spiliopoulou, M. and Lenz, H.-J. (2005) The influence of incentives and culture on knowledge sharing. In 38th Hawaii International Conference on System Sciences (HICSS-38 2005). IEEE Computer Society, Big Island, Hawaii, USA.

Newman, M. E. J. (2001a) Scientific collaboration networks. I. Network construction and fundamental results. Physical Review E, 64(016131).

Newman, M. E. J. (2001b) Scientific collaboration networks. II. Shortest paths, weighted networks, and centrality. Physical Review E, 64(016132).

Newman, M. E. J. (2001c) Who is the best connected scientist? A study of scientific coauthorship networks. Physical Review E, 64(1).

Newman, M. E. J. (2003a) Mixing patterns in networks. Physical Review E, 67(026126).

Newman, M. E. J. (2003b) The structure and function of complex networks. SIAM Review, 45(2): 167-256.

Newman, M. E. J. (2004a) Detecting community structure in networks. The European physical journal B, 38(2): 321-330.

Newman, M. E. J. (2005) A measure of betweenness centrality based on random walks. Social Networks, 27: 39-54.

Newman, M. E. J. (2006a) Finding community structure in networks using the eigenvectors of matrices. Physical Review E, 74(036104).

Newman, M. E. J. (2006b) Modularity and community structure in networks. American Physical Society, ASP March Meeting.

Newman, M. E. J. and Girvan, M. (2004) Finding and evaluating community structure in networks. Physical Review E, 69(026113).

Newman, M. J. E. (2004b) Fast algorithm for detecting community structure in networks. Physical Review E, 69(066133).

Ng, R. T. and Han, J. (1994) Efficient and Effective Clustering Methods for Spatial Data Mining. In 20th International Conference on Very Large Data Bases. Santiago de Chile, Chile, pp. 144-155.

Nieminen, J. (1974) On clustering in a graph. Scandinavian Journal of Psychology, 15: 322-336.

Nonaka, I. and Takeuchi, H. (1995) The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovations, Oxford University Press.

Ordonez, C. and Omiecinski, E. (2002) FREM: fast and robust EM clustering for large data sets. In CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management. ACM Press, McLean, Virginia, USA, pp. 590-599.

Orlandic, R., Lai, Y. and Yee, W. G. (2005) Clustering high-dimensional data using an efficient and effective data space reduction. In 14th ACM International Conference on

Information and Knowledge Management. ACM Press, Bremen, Germany, pp. 201-208.

Pal, N. R. and Biswas, J. (1997) Cluster validation using graph theoretic concepts. Pattern Recognition, 30(6): 847-857.

Pantel, P. and Ravichandran, D. (2004) Automatically labeling semantic classes. In Proceedings of the Human Language Technology / North American Association for Computational Linguistics (HLT/NAACL-04). Boston, M.A., pp. 321-328.

Passoni, L. (2005) Gestión del conocimiento: una aplicación en departamentos académicos. In Gestión y Política Publica. Vol. XIV.

Pawlak, Z., Grzymala-Busse, J. W., Slowinski, R. and Ziarko, W. (1995) Rough sets. Communications ACM, 38(11): 89-95.

Pérez, A. and Medina, J. E. (2007) A clustering algorithm based on generalized stars. In MLDM.

Pinney, J. W. and Westhead, D. R. (2006) Betweenness-based decomposition methods for social and biological networks.

Popescul, A. and Ungar, L. (2000) Automatic labeling of document clusters.

Qian, Y., Zhang, G. and Zhang, K. (2004) FA\ÇADE: a fast and effective approach to the discovery of dense clusters in noisy spatial data. In SIGMOD '04: Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data. ACM Press, Paris, France, pp. 921-922.

Radicchi, F., Castellano, C., Cecconi, F., Loreto, V. and Parisi, D. (2004a) Defining and identifying communities in networks. PNAS Proc. National Academy of Science USA, 101: 2658-2663.

Radicchi, F., Castellano, C., Cecconi, F., Loreto, V. and Parisi, D. (2004b) Defining and identifying communities in networks. PNAS Proc. National Academic of Science USA, 101(9).

Rattigan, M. J., Maier, M. and Jensen, D. (2007) Graph clustering with network structure indices. In ICML '07: Proceedings of the 24th international conference on Machine learning. ACM Press, Corvalis, Oregon, pp. 783-790.

Reed, S. K. (1972) Pattern recognition and categorization. Journal Cognitive Pshychology, 3: 382-407.

Rodríguez, M. (1999) El concepto de tipo y la teoría de programación actual. Revista GIGA, 4.

Ruiz-Shulcloper, J., Alba-Cabrera, E. and Sánchez-Díaz, G. (2000) DGLc: a density-based global logical combinatorial clustering algorithm for large mixed incomplete data. In Geoscience and Remote Sensing Symposium. IGARSS. Vol. 7 IEEE 2000 International, pp. 2846-2848.

Rumbaugh, J., Booch, G. and Jacobson, I. (1997a) UML Notation Version 1.1.

Rumbaugh, J., Booch, G. and Jacobson, I. (1997b) UML Semantics Version 1.1.

Sabidussi, G. (1966) The centrality index of a graph. Psychometrika, 31: 581-603.

Sahami, M. (1998) Using machine learning to improve informatio access. In Department of Computer Science. Vol. PhD. Thesis Stanford University, Standford, USA.

Salton, G. and Buckley, C. (1988) Term weighting approaches in automatic text retrieval. Information Processing and Management, 24(5): 513-523.

Salton, G., Wong, A. and Yang, C. S. (1975) A vector space model for automatic text retrieval. Communications of the ACM, 18(11): 613-620.

Schürmann, J. (1996) Pattern Clasification: a unified view of statistical and neural aproaches, New York, USA.

Schwartz, G. (1978) Estimation the dimension of a model. Ann Statu, 6: 461-464.

Scott, J. (2000) Social Network Analysis: A Handbook, Sage Publications, London.

Shaw, M. E. (1954) Group structure and the behavior of individuals in small groups. Journal of Psychology, 38: 139-149.

Sheikholeslami, G., Chatterjee, S. and Zhang, A. (1998) WaveCluster: A multi-resolution clustering approach for very large spatial databases. In 24th International Conference on Very Large Data Bases. Morgan Kaufmann, New York, NY, USA, pp. 428-439.

Sheikholeslami, G., Chatterjee, S. and Zhang, A. (2000) WaveCluster: a wavelet-based clustering approach for spatial data in very large databases. The VLDB Journal, 8(3-4): 289-304.

Silberschatz, A. and Tuzhilin, A. (1996) What makes patterns interesting in knowledge discovery systems. IEEE Trans. on Knowledge and Data Engineering, 8(6).

Skupin, A. and Jongh, C. d. (2005) Visualizing the ICA - a content-based approach.

Slowinski, R. and Vanderpooten, D. (1997) In Advances in Machine Intelligence & Soft-Computing, Vol. IV (Ed, Wang, P. P.), pp. 17-33.

Stein, B. and Eissen, S. M. z. (2004) Topic identification: framework and application. Journal of Universal Computer Science. Proceedings of the I-KNOW'04 4th International Conference on Knowledge Management: 353-360.

Stein, B., Eissen, S. M. z. and Wißbrock, F. (2003) On clustering validity and the information need of users. In Proceedings of 3rd IASTED Int. Conference on Artificial Intelligence and Applications (AIA 03). (Ed, Hanza, M. H.) ACTA Press, Benalmádena, Spain, pp. 216-221.

Steinbach, M., Karypis, G. and Kumar, V. (2000) A comparison of document clustering techniques. In Proceedings of KDD Workshop on Text Mining.

Strehl, A., Ghosh, J. and Mooney, R. (2000) Impact of similarity measures on Web-page clustering. In 17th NAtional Conference on Artificial Intelligence (AAAI-2000): Workshop of Artificial Intelligence for Web Search. Austin, Texas.

Stumpf, M. P. H., Wiuf, C. and May, R. M. (2005) Subnets of scale-free networks are not scale-free:Sampling properties of networks. PNAS Proc. National Academy of Sciences USA, 102: 4221-4224.

Tan, A. (1999) Text Mining: The state of the art and the challenges. In PAKDD'99. pp. 65-70.

Theodoridis, S. and Koutroubas, K. (1999) Pattern Recognition, Academic Press.

Tiwana, A. (2000) The Knowledge Management Toolkit, Prentice Hall Inc.

Tong, H. and Faloutsos, C. (2006) Center-piece subgraphs: problem definition and fast solutions. In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM Press, Philadelphia, PA, USA pp. 404-413.

Torre, F. d. l. and Kanade, T. (2006) Discriminative cluster analysis. In ICML '06: Proceedings of the 23rd International Conference on Machine Learning. ACM Press, Pittsburgh, Pennsylvania, pp. 241-248.

Treeratpituk, P. and Callan, J. (2006) Automatically labeling hierarchical clusters. In Proceedings of the International Conference on Digital Government Research. Vol. 151 San Diego, California, pp. 167-176.

Tuzhilin, A. (2002) In Handbook of Data Mining and Knowledge DiscoveryOxford University Press.

Wang, W., Yang, J. and Muntz, R. R. (1997) STING: a statistical information grid approach to spatial data mining. In 23rd International Conference on Very Large Data Bases. Morgan Kaufmann, Athens, Greece, pp. 186-195.

Wasserman, S. and Faust, K. (1994a) Social Network Analysis, Cambridge University Press.

Wasserman, S. and Faust, K. (1994b) Social network analysis: methods and applications, Cambridge University Press, Cambridge.

Watts, D. J. and Strogatz, S. H. (1998) Collective dynamics of 'small-world' networks. Nature, 393(6684): 440-442.

White, S. and Smyth, P. (2005) A spectral clustering approach to finding communities in graphs. Proc. SIAM International Conference of Data Mining.

Wilson, D. R. and Martínez, T. R. (1997) Improved heterogeneous distance functions. Journal of Artificial Intelligence Research, 6: 1-34.

Wu, A. Y., Garland, M. and Han, J. (2004) Mining scale-free networks using geodesic clustering. In KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM Press, Seattle, WA, USA, pp. 719-724.

Xie, X. L. and Beni, G. (1991) A validity measure for fuzzy clustering. IEEE Transactions on Pattern Analysis and Machine Learning, 13(4): 841-846.

Xiong, H., Wu, J. and Chen, J. (2006) K-means clustering versus validation measures: a data distribution perspective. In KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM Press, Philadelphia, PA, USA, pp. 779-784.

Xu, X., Yuruk, N., Feng, Z. and Schweiger, T. A. J. (2007) In KDD '07: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data MiningACM Press, San Jose, California, USA, pp. 824-833.

Zahn, C. T. (1971) Graph-theoretical methods for detecting and describing gestalt clusters. IEEE Trans. Comput., 20: 68-86.

Zhang, T., Ramakrishnan, R. and Livny, M. (1996) BIRCH: An efficient data clustering method for very large databases. In International Conference on Management of Data (SIGMOD). Vol. 25 ACM Press, Montreal, QB, Canada, pp. 103-114.

Zhao, Y., Zhang, C. and Shen, Y.-D. (2004) Clustering high-dimensional data with low-order neighbors. In IEEE/WIC/ACM International Conference on Web Intelligence. IEEE Computer Society, pp. 103-109.

Zhong, N., Skowron, A. and Ohsuga, S. (Eds.) (1999) New Directions in Rough Sets, Data Mining, and Granular Soft-Computing, 17th International Workshop, RSFDGrC'99, Yamaguchi, Japan, November 8-11, 1999, Proceedings, Springer.

Anexos

Anexo 1. GC en organizaciones *v.s* CC en Internet

Sistemas de gestión del conocimiento dentro de las organizaciones	**Mercado del conocimiento en Internet**
En un sistema interno y cerrado	En un entorno abierto
Equipado con un mecanismo de incentivos	Donde la motivación es ganar dinero
Diseñado para promover el establecimiento de la reputación y usualmente estimular reciprocidad	Donde la reputación es usada para el control de la calidad y para adquirir poder de negociación con configuraciones de precio
Los participantes conocen a cada una de las otras personas	Los participantes usualmente tienen solamente seudónimos
O al menos se conocen como roles	Y en algunos casos, conocer a cada uno de los otros, es irrelevante para la participación en el negocio
Hay relaciones institucionales entre los participantes	No hay relaciones institucionales entre los participantes
El dinero no es la motivación primaria para participar	Los expertos contribuyen al propósito directo de ganar dinero

Anexo 2. Enfoques lingüísticos para analizar significados respecto al contexto.

Estos enfoques se dividen en cinco niveles (Lanquillon 2001):

1. Nivel de grafema: Análisis sobre un nivel de sub-palabra, comúnmente concerniente a las letras.
2. Nivel léxico: Análisis concerniente a palabras individuales.

Note que los dos primeros niveles operan solamente con un plano estadístico sobre el texto, es decir básicamente sobre frecuencias de combinaciones de términos, que pueden ser letras o palabras.

3. Nivel sintáctico: Análisis concerniente a la estructura de oraciones.
4. Nivel semántico: Análisis relativo al significado de palabras y frases.
5. Nivel pragmático: Análisis relativo al significado, tanto dependientes del contexto como independientes del contexto (por ejemplo, aplicaciones específicas, contextos).

Los niveles más altos del análisis de textos existen para tratar de capturar mayor contenido semántico a través de la explotación de la cantidad creciente de información contextual tal como la estructura de las oraciones, párrafos o documentos, sin embargo, estos niveles implican un procesamiento computacional muy costoso.

Anexo 3. Distancias, similitudes y disimilitudes más usadas al comparar objetos

Sean los objetos O_i y O_j descritos por k rasgos, donde $O_i=(o_{i1}, \ldots, o_{ik})$ y $O_j=(o_{j1}, \ldots, o_{jk})$

Distancia Euclideana

$$D_{Euclideana}(O_i, O_j) = \sqrt{\sum_{h=1}^{k} (o_{ih} - o_{jh})^2} \qquad \textbf{(A3.1)}$$

Distancia <u>Minkowski</u> (Batchelor 1978)

$$D_{Minkowski}(O_i, O_j) = \left(\sum_{h=1}^{k} \left| o_{ih} - o_{jh} \right|^{\gamma} \right)^{\frac{1}{\gamma}} \text{ donde } \gamma \geq 1 \qquad \textbf{(A3.2)}$$

La distancia Minkowsky es equivalente a la distancia Manhattan o city-block, función alternativa que requiere menos esfuerzo computacional, y a la distancia Euclideana cuando γ es 1 y 2, respectivamente (Batchelor 1978). Para los valores de $\gamma \geq 2$, la distancia Minkowsky es equivalente a la distancia Supermum (Hand 1981, Reed 1972).

Distancia Euclideana heterogénea (Heterogenous Euclidean – Overlap Metric; HEOM)

$$D_{HEOM}(O_i, O_j) = \sqrt{\sum_{h=1}^{k} d_{local}(o_{ih}, o_{jh})^2}, \text{ donde}$$

$$d_{local}(o_{ih}, o_{jh}) = \begin{cases} d_{Overlap}(o_{ih}, o_{jh}) & \textit{si h simbólico} \\ d_{NormEuclidean}(o_{ih}, o_{jh}) & \textit{si h numérico} \end{cases} \qquad \textbf{(A3.3)}$$

$$d_{Overlap}(o_{ih}, o_{jh}) = \begin{cases} 0, & \textit{si } o_{ih} = o_{jh} \\ 1, & \textit{en otro caso} \end{cases} \quad \text{y} \quad d_{NormEuclidean}(o_{ih}, o_{jh}) = \frac{\left| o_{ih} - o_{jh} \right|}{\max_h - \min_h}$$

Distancia Camberra (Michalski, Stepp et al. 1981, Diday 1974)

$$D_{Camberra}(O_i, O_j) = \sum_{h=1}^{k} \frac{\left| o_{ih} - o_{jh} \right|}{\left| o_{ih} + o_{jh} \right|} \qquad \textbf{(A3.4)}$$

Correlación de Pearson (Wilson and Martínez 1997)

$$D_{Pearson}(O_i, O_j) = \frac{\sum_{h=1}^{k} (o_{ih} - \overline{atributo_h})(o_{jh} - \overline{atributo_h})}{\sqrt{\sum_{h=1}^{k} (o_{ih} - \overline{atributo_h})^2 \sum_{h=1}^{k} (o_{jh} - \overline{atributo_h})^2}} \qquad \textbf{(A3.5)}$$

Donde $\overline{atributo_h}$ es el valor promedio que toma el $atributo_h$ en el conjunto de datos.

Las expresiones de Chebychev, Mahalanobis, distancia de Hamming y la máxima distancia son otras variantes de cálculo de distancias entre objetos (Wilson and Martínez 1997). En

(Duch 2002) se presentan formas de medir la similitud considerando una relación asimétrica entre los objetos y teniendo en cuenta la probabilidad condicional de un objeto respecto al otro.

Existen varias formas de medir la similitud entre objetos, sin embargo, existen pocos estudios comparativos de ellas y sus efectos en el agrupamiento. En la minería de textos, al comparar documentos con el objetivo de agruparlos, la determinación de la similitud entre documentos depende de la representación del documento y de los pesos que se le asignen al caracterizarlo. A partir de estudios realizados, se ha demostrado que al agrupar documentos, los coeficientes Dice, Jaccard y Coseno, han reportado los mejores resultados (Frakes and Baeza-Yates 1992). Una valoración del imparto de la distancia Euclideana y los coeficientes Dice, Jaccard y Coseno en dominios textuales fue presentada en (Strehl, Ghosh et al. 2000).

Coeficiente Dice

$$S_{Dice}(O_i, O_j) = \frac{2\sum_{h=1}^{k}(o_{ih} \cdot o_{jh})}{\sum_{h=1}^{k} o_{ih}^2 + \sum_{h=1}^{k} o_{jh}^2} \qquad \textbf{(A3.6)}$$

Coeficiente de Jaccard

$$S_{Jaccad}(O_i, O_j) = \frac{\sum_{h=1}^{k}(o_{ih} \cdot o_{jh})}{\sum_{h=1}^{k} o_{ih}^2 + \sum_{h=1}^{k} o_{jh}^2 - \sum_{h=1}^{k}(o_{ih} \cdot o_{jh})} \qquad \textbf{(A3.7)}$$

Coeficiente Coseno

$$S_{Coseno}(O_i, O_j) = \frac{\sum_{h=1}^{k} (o_{ih} \cdot o_{jh})}{\sqrt{\sum_{h=1}^{k} o_{ih}^2 \cdot \sum_{h=1}^{k} o_{jh}^2}} \quad \textbf{(A3.8)}$$

Anexo 4. Variantes para el cálculo del umbral de similitud entre objetos

El modelo propuesto considera tres variantes para el cálculo inicial del umbral β_0 y a continuación son descritas (García 1999):

a) La primera variante calcula el umbral β_0 hallando la media de las distancias entre todos los pares de documentos posibles. Así se expresa en la fórmula (A4.1):

$$\beta_0 = \frac{2}{n(n-1)} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} d(O_i, O_j) \quad \textbf{(A4.1)}$$

b) La segunda variante halla la media de los valores máximos de las distancias entre cualquier par de documentos, según la expresión (A4.2):

$$\beta_0 = \frac{1}{n} \sum_{i=1}^{n} \max_{\substack{j=1..n \\ i \neq j}} d(O_i, O_j) \quad \textbf{(A4.2)}$$

c) La tercera variante toma la mínima de todas las distancias posibles entre pares de documentos, sin tener en cuenta las distancias que sean cero, así se muestra en la fórmula (A4.3):

$$\beta_0 = \min_{\substack{i=1..n-1 \\ i \neq j}} \left\{ \min_{j=i+1..n} d(O_i, O_j) \right\} \quad \textbf{(A4.3)}$$

La descripción de la notación utilizada es la siguiente: *n* es la cantidad de documentos de la colección y $d(O_i, O_j)$ es el valor de la distancia entre los vectores documento O_i y O_j.

Anexo 5. Algoritmo jerárquico divisivo GN, debido a Girvan y Newman

La forma general del algoritmo es la siguiente:

1. Calcular los valores de intermediación para todas las aristas en la red.
2. Encontrar la arista con mayor valor de intermediación y eliminarla.
3. Recalcular la intermediación para todas las aristas restantes.
4. Repetir desde el paso 2.

Donde la expresión **(A5.1)** corresponde a una variante normalizada para el cálculo de la intermediación $btw(e)$ de una arista e (Holme 2002), donde $cpath(i, j)$ es el número de caminos más cortos entre los nodos i y j del grafo y $cpath_e(i, j)$ es el número de aquellos que adicionalmente pasan por e. Este cociente puede ser interpretado como el rol que juega la arista e en la relación entre los nodos i y j.

$$btw(e) = \frac{cpath_e(i, j)}{cpath(i, j)} \qquad \textbf{(A5.1)}$$