ESCUELA SUPERIOR POLITECNICA DEL LITORAL

FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN

INFORME DE MATERIA DE GRADUACIÓN

"SISTEMA DE AGRUPAMIENTO Y BÚSQUEDA DE CONTENIDOS DE LA BLOGOSFERA DE LA ESPOL, UTILIZANDO HADOOP COMO PLATAFORMA DE PROCESAMIENTO MASIVO Y ESCALABLE DE DATOS (BONSAI)"

Previo a la obtención del título de:

INGENIERO EN COMPUTACIÓN ESPECIALIZACIÓN SISTEMAS TECNOLÓGICOS

PRESENTADA POR:

ALLAN ROBERTO AVENDAÑO SUDARIO

GUAYAQUIL – ECUADOR

2009

AGRADECIMIENTO

A Dios.

A mis padres, hermanas y familiares.

Especialmente a MsC. Cristina Abad y MsC. Carmen Vaca por su incondicional apoyo como mentoras y consejeras en todo momento.

DEDICATORIA

A mis padres por su cariño y
por su apoyo incondicional.
A mis hermanas y a mis familiares,
en especial a mi abuelita Carmen M.F. y
a mi tía abuela Teresita M.F.

TRIBUNAL DE GRADO

DIRECTORA DEL PROYECTO DE GRADUACIÓN

MsC. Cristina Abad R.

PROFESORA DESIGNADA POR EL SUBDECANO

MsC. Carmen Vaca R.

DECLARACIÓN EXPRESA

"La responsabilidad por los hechos, ideas y doctrinas expuestas en esta tesis, me corresponden exclusivamente; y, el patrimonio intelectual de la misma, a la Escuela Superior Politécnica del Litoral"

(Reglamento de exámenes y títulos profesionales de la ESPOL)

Allan Avendaño Sudario

RESUMEN

En la actualidad, con la implementación de la plataforma de blogs en la ESPOL ha permitido la diversificación de contenidos creados por los miembros de la comunidad politécnica; a pesar, que resulta casi imposible obtener entradas determinadas de acuerdo a términos específicos.

Debido a estas razones, el presente trabajo propone desarrollar un sistema de indexación y búsqueda de contenidos en la blogosfera politécnica con el propósito de mejorar la visibilidad de contenidos y asegurar la expansión de las pequeñas comunidades digitales conformadas espontáneamente.

En el primer capítulo se detallan los antecedentes sobre los que se plantearon los objetivos y la justificación para la realización del presente trabajo.

En el capítulo segundo, se expone toda la plataforma tecnológica y los conceptos esenciales durante el desarrollo del proyecto.

En el tercer capítulo se analizan los requerimientos, con los cuales se describe el diseño modular del sistema propuesto, compuesto por:

1. Módulo de agrupamiento de contenidos.
2. Módulo de indexación y búsqueda por términos.

Finalmente, en el capítulo cuarto se especifican las pruebas efectuadas al sistema propuesto.

ÍNDICE GENERAL

ÍNDICE DE GRÁFICOS

INTRODUCCIÓN

En la actualidad, medios digitales como los blogs, son utilizados para desarrollar los sentidos de colectividad y cooperación en comunidades previamente establecidas (por ejemplo instituciones educativas o empresas), por lo cual, resulta necesario fomentar la interrelación entre los miembros que les conforman.

Para lograr este propósito, se implementan herramientas que reflejen la actividad de sus miembros. Un ejemplo claro de este tipo de herramientas son los directorios, que generalmente muestran las actualizaciones recientes en la comunidad en un orden determinado.

Es por esto, que en el presente documento se describe el proceso de desarrollo e implementación de un motor de búsquedas de entradas en la blogosfera politécnica, cuyo objetivo es el de proporcionar un medio de socialización de los contenidos publicados en la comunidad. Adicionalmente, este sistema cuenta con un mecanismo de recomendación de entradas, el cual funciona a partir de los resultados obtenidos de las búsquedas.

CAPÍTULO 1

1. ANTECEDENTES Y JUSTIFICACIÓN

En este capítulo se provee al lector de los antecedentes sobre los que se plantearon los objetivos y la justificación del presente trabajo.

1.1. Antecedentes

La Escuela Superior Politécnica del Litoral, se ha planteado mejorar la visibilidad del contenido que se genera en su dominio Web, con el propósito de colocarse en los primeros lugares de la lista de sitios Webs de universidades a nivel mundial.

Una de las medidas adoptadas para cumplir con esta meta, consistió en crear la blogosfera de la ESPOL cuyos autores son los miembros que componen la comunidad politécnica.

Desde su creación, la comunidad de autores de blogs politécnicos, ha aumentado constantemente debido a la facilidad de publicación de contenidos personales e institucionales, lo cual ha permitido diversificar los temas de los cuales se escriben en la ESPOL.

Con una variedad de tópicos que referencien el dominio de ESPOL, se ha logrado una mayor visibilidad externa; en contraste, no existe un medio que permita socializar el contenido publicado por sus miembros.

En la actualidad, la blogosfera politécnica cuenta con un directorio, en el cual se listan las entradas recientemente publicadas [2].

1.2. Planteamiento del Problema

En la actualidad, los blogs son el medio digital de más fácil acceso para la publicación de contenidos personales e institucionales en la Web. Se caracterizan porque son de fácil publicación de contenidos y permiten rastrear las actualizaciones de otros blogs de interés, mediante sistemas de agregación de noticias (RSS[1]). Además, el proceso de publicación de artículos o entradas en un blog no requiere de mucho tiempo ni experticia en tecnologías de la información.

Dado que generalmente los blogs se utilizan para publicar contenido personal, los enlaces pueden considerarse como un medio de socialización entre los autores de blogs. Estas pequeñas sociedades de enlaces se estructuran según el criterio del autor de acuerdo a la

[1] RSS es un formato para la sindicación de contenidos de páginas web. Sus siglas responden a Really Simple Syndication.

semejanza de los artículos publicados entre blogs y también, debido a las motivaciones que fundamentan las comunidades Web [1]:

- **Identidad:** Los miembros utilizan las comunidades Web para crear una identificación en sus grupos sociales.
- **Singularidad:** Las contribuciones de los usuarios deben ser únicas y valiosas para la comunidad.
- **Reciprocidad:** La colaboración dentro de las comunidades Web, debe motivar a los usuarios a crear la conciencia de colectividad y ayuda.
- **Reputación:** Los miembros participan activamente con el objetivo de construir su reputación dentro de la comunidad Web que participan; además, para mejorar la relación con el resto de miembros.
- **Sentido de eficacia:** El efecto positivo de las colaboraciones en las comunidades Web, determina a los usuarios a continuar participando activamente.
- **Control:** Los miembros requieren cierto grado de control sobre la información que es compartida y mostrada dentro de la comunidad Web.

- **Pertenencia:** Los miembros participan porque se sienten comprometidos con la información que se maneja en la comunidad.

El “Sistema de indexación y búsqueda de contenidos generados en la blogosfera de la ESPOL, utilizando Hadoop como plataforma de procesamiento masivo y escalable de datos” ha sido diseñado con el propósito de consolidar la comunidad de autores de blogs y proporcionar una visión general de la blogosfera politécnica al socializar las publicaciones mediante la implementación de un motor de búsqueda y agrupamiento de contenidos.

1.3. Objetivos

El “Sistema de agrupamiento y búsqueda de contenidos de la blogosfera de la ESPOL, utilizando Hadoop como plataforma de procesamiento masivo y escalable de datos (BONSAI)”, fue planteado para mejorar las búsquedas dentro de los contenidos de la blogosfera de la ESPOL. Para cumplir con este fin, se plantearon los siguientes objetivos:

1. Implementar un módulo de agrupamiento de contenidos de la blogosfera de la ESPOL, con el uso de herramientas de procesamiento masivo de datos al aplicar el paradigma

Map/Reduce con el uso de Hadoop como plataforma de procesamiento.

2. Implementar un módulo de búsqueda de entradas publicadas en los blogs.

3. Implementar una interfaz Web para el sistema de búsqueda y agrupamiento de contenidos de la blogosfera de la ESPOL.

1.4. Justificación

La justificación principal para el desarrollo del “Sistema de agrupamiento y búsqueda de contenidos de la blogosfera de la ESPOL, utilizando Hadoop como plataforma de procesamiento masivo y escalable de datos (BONSAI)” consiste que, al ser implementado, permitiría aumentar la visibilidad de los contenidos que se producen en la ESPOL al proveer de un motor de búsqueda de blogs dentro de su dominio; con lo cual se aseguraría la expansión de las pequeñas comunidades digitales.

CAPÍTULO 2

2. FUNDAMENTACIÓN TEÓRICA

En este capítulo se revisa los fundamentos teóricos indispensables para el desarrollo del presente trabajo.

2.1. Tratamiento masivo de datos

2.1.1. Descripción

En la era de la información, las máquinas producen grandes cantidades de datos, resultado de registrar los eventos generados por máquinas y usuarios. Es así que en el 2006 se estimó que el volumen de información que se maneja en las redes digitales mundiales, alcanzó los 0.18 Zettabytes y se proyecta que para el 2011 llegará a los 1.8 Zettabytes [3]. Un Zettabyte es equivalente a un billón de Terabytes.

En la actualidad, almacenar estos volúmenes de datos resulta barato en comparación al desarrollo de recursos para la extracción de información relevante, útil para la elaboración de reportes y toma de decisiones.

Estas tareas de procesamiento de datos a gran escala requieren de recursos que en la actualidad no están disponibles en equipos de uso general. Para resolver este tipo de problemas es necesario analizar las diversas soluciones tecnológicas que existen en el mercado.

Una posibilidad para resolver este tipo de tareas consiste en adquirir equipos de cómputo de alto rendimiento, lo cual, resulta costoso y poco escalable en corto tiempo, debido al constante incremento del volumen de datos a procesar.

Otra alternativa, consiste en diseñar y construir un clúster de alta disponibilidad, el cual que luzca como una sola máquina. Esto requiere de una instalación especializada y una administración de servicios centralizada. En la mayoría de los casos, para esta solución se adquieren equipos de licencia propietaria costosa.

La solución más económica es la que suplen los servicios de la computación en la nube para el procesamiento masivo de datos. Generalmente, los datos que se procesan en este ambiente son transformados mediante algoritmos de instrucciones simples.

Por ejemplo, la red social de Facebook compuesta por más de 200 millones de usuario producen cerca de dos Petabytes de datos [4] [12], los cuales, son procesados en una plataforma de procesamiento distribuido de datos.

En el 2007, su infraestructura de procesamiento masivo de datos fue construida sobre un RDBMS comercial. Pero, debido al creciente volumen de registros, en el 2008 recurrieron a Hadoop, una propuesta de código abierto para el tratamiento masivo y distribuido de datos [13].

2.1.2. Paradigma Map/Reduce

Desde el año 2000 el equipo de Google, realiza cientos de transacciones computacionales para resolver problemas que comprenden numerosas unidades de procesamiento y grandes conjuntos de datos, por ejemplo:

- Búsqueda y escaneo de patrones de texto (grep en ambiente distribuido)
- Conteo de unidades.
- Problemas de indexación de términos (índices invertidos de páginas Web [7])
- Ordenamiento de grandes conjuntos de datos.
- Extracción de información con técnicas de minería de texto.

El equipo de desarrolladores de Google decidió diseñar e implementar una plataforma que permita paralelizar tareas y distribuir los datos a procesar en un clúster de máquinas de propósito general.

De acuerdo al diseño original, la plataforma debía permitir la planificación, seguimiento y reporte de tareas por cada nodo del clúster; además, debía detectar y recuperarse de las fallas comunes para este tipo de equipos [5].

Map/Reduce es el modelo de programación desarrollado por Google para resolver sus tareas de procesamiento de datos a larga escala, inspirado en las operaciones que implementan lenguajes funcionales como Lisp [11].

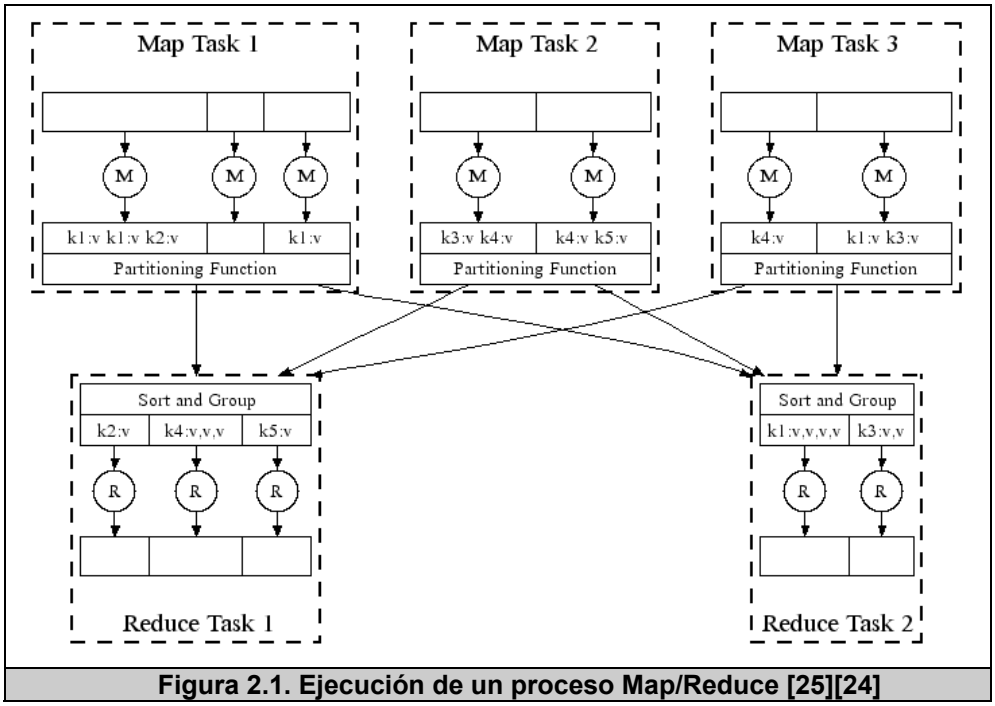

Figura 2.1. Ejecución de un proceso Map/Reduce [25][24]

En la **Figura 2.1** se muestra el esquema de ejecución de un proceso, diseñado bajo el paradigma Map/Reduce. La función *map* procesa pares del tipo clave/valor para generar un conjunto intermedio de

pares del mismo tipo, los cuales son parcialmente agrupados y ordenados. Esto pares intermedios son procesados con funciones llamadas *reduce*, las cuales emiten un archivo final con cada claves y sus respectivos valores asociados.

El paradigma Map/Reduce se ha difundido exitosamente [10] debido a la facilidad para expresar problemas comunes en procesos map y reduce por parte de los desarrolladores que no cuentan con experiencia en sistemas paralelos o distribuidos y a las consideraciones de diseño, como la tolerancia a fallos, el balanceo de carga en los nodos y a el paralelismo de los procesos.

Ejemplos exitosos de la expansión de este paradigma es la plataforma de código abierto, Hadoop, y los servicios bajo demanda para el procesamiento masivo de datos en la nube que ofrece Amazon a través de los Amazon Web Services [33].

2.1.3. Hadoop: Plataforma de procesamiento masivo de datos

Hadoop [6] es un proyecto de Apache Software Foundation, que provee una plataforma para el procesamiento en distribuido y masivo de datos en computadores de propósito general, basada en el estilo de programación Map/Reduce desarrollado por Google.

Esta plataforma, se presenta como una solución de código abierto para los programadores sin experiencia en desarrollo de aplicaciones

para ambientes distribuidos, ya que oculta la implementación de detalles propios de estos sistemas: paralelismo de tareas, tolerancia a fallos, administración de procesos y balanceo de carga [5][15].

Para procesar cientos o miles de archivos de gran tamaño (en el orden del Gigabytes ó Terabytes) se requiere de un sistema de archivos que permita múltiples operaciones de lectura en disco y muy pocos para escritura, a más de alta tolerancia a fallos y disponibilidad para ser instalado en máquinas de propósito general. Además de estos y otros requerimientos, como la replicación de bloques de datos entre los nodos, portabilidad entre diversos sistemas operativos y el alto rendimiento en acceso a datos, fue diseñado el HDFS (Hadoop Distributed FileSystem) [16] el cual está basado en el sistema de archivos GFS diseñado por Google [27].

El HDFS se basa en la arquitectura maestro/esclavo, con un único *Namenode* que administra el espacio de nombres de los archivos y regula el acceso a los mismos, y varios *Datanodes* que gestionan el almacenamiento de los archivos. Los archivos son divididos en bloques (el tamaño por defecto es de 64 megabytes), los cuales son replicados en los nodos clientes para asegurar la tolerancia a fallos del HDFS.

Debido a estas características requeridas para el procesamiento de datos a gran escala, Yahoo! puso en producción en febrero de 2008 [14], su índice de búsqueda generado por un clúster Hadoop compuesto por 10,000 nodos. Desde entonces, compañías como Last.fm [28], Facebook [12] y New York Times [29] han utilizado la plataforma de Hadoop para el procesamiento masivo de datos, a través de los servicios EC2 de Amazon o de clústeres propios.

2.2. Recuperación de Información (Information Retrieval)

2.2.1. Descripción

Con la aparición de los medios electrónicos ha aumentado la generación de documentos y, de igual manera, la capacidad de almacenamiento de los dispositivos electrónicos, aunque sigue siendo mínima la extracción de información contextual de los documentos.

El proceso de extracción de información consiste en representar, almacenar, organizar y acceder a documentos relevantes tomados a partir de una colección de documentos sin estructurar (generalmente en lenguaje natural), con el objetivo de satisfacer las necesidades de los usuarios [17] [18].

Estos sistemas se caracterizan por el volumen de datos a procesar, por ejemplo los sistemas de búsqueda deben explorar sobre millones

de documentos y obtener sólo aquellos que satisfagan una expresión lógica. Este tipo de búsquedas sobre grandes colecciones de documentos requieren de estructuras de índices que permitan el acceso rápido a documentos específicos, a este proceso se lo conoce como indexación.

Estos sistemas de extracción de información además, brindan el soporte a los usuarios al filtrar los resultados en grupos de documentos automáticamente creados. Las técnicas de agrupamiento o clustering permiten obtener grupos de documentos similares entre sí, sin la supervisión de expertos.

Es por esto que estas técnicas son empleadas para que los usuarios finales tengan una visión global de las características de colecciones de objetos con mayor facilidad.

2.2.2. Agrupamiento de objetos

Una de las técnicas utilizadas para comprender la naturaleza de las colecciones de objetos, consiste en dividirlas en pequeños grupos de elementos que compartan cierto grado de similitud entre sí.

Este enfoque ha sido utilizado en diversas áreas científicas y comerciales, incluyendo la organización de resultados de búsqueda y marketing. Es así, que se han utilizado para analizar el comportamiento histórico de los usuarios en la Web ó en las

bibliotecas al agrupar manualmente los libros de acuerdo a los tópicos tratados.

Para agrupar elementos no es necesario plantear un análisis previo en el que se determine, por ejemplo, la independencia de las variables ó de la supervisión de un experto que determine la similitud. Sin embargo, es necesario seleccionar las variables relevantes que describan con precisión la naturaleza de los objetos.

Otro de los requerimientos durante el agrupamiento de objetos es la métrica de distancia. La cual es una expresión matemática en la que se evalúan las características de los objetos para determinar la proximidad entre estos.

En el área de extracción de información, las técnicas de agrupamiento de documentos son utilizadas con diversos fines [18]. Es así, que en lugar de listar los resultados en la forma tradicional, los documentos resultantes de una búsqueda son agrupados de acuerdo a la similitud de los términos que contienen.

El resultado del proceso de agrupamiento de objetos son grupos cuyos elementos comparten características comunes. Los algoritmos se clasifican de acuerdo a las características del agrupamiento, descritas a continuación.

De acuerdo a la relación que existe entre los grupos. Los algoritmos de *particionamiento* generan grupos sin una estructura explícita que relacione se relacionen con otros grupos. Por el contrario, los algoritmos *jerárquicos* organizan los grupos en categorías y consecuentemente son organizados en sub-categorías.

De acuerdo a la membresía de los documentos, los algoritmos de agrupamiento *difusos* asignan una fracción de la membresía de los documentos a diversos grupos. En contraste, para los algoritmos de agrupamiento *fuerte* cada documento puede pertenecer a uno y sólo un grupo.

La decisión de utilizar un determinado algoritmo de agrupamiento depende en particular a cada problema.

2.2.3. Algoritmos de Agrupamiento

Uno de los algoritmos de agrupamiento más utilizado es el conocido como K-means. El cual genera k grupos de elementos. Este número debe ser seleccionado ante del procesamiento; además, debe ser mayor que dos y menor que el número de elementos que componen la colección [20].

El objetivo de este algoritmo consiste en minimizar el promedio cuadrado de la distancia de cada documento con el centroide de cada grupo.

2.2.3.1. K-means

El proceso de agrupamiento comienza por definir el número de grupos a crear. A cada grupo se le asigna un elemento escogido aleatoriamente. Luego, todos los elementos son comparados con cada uno de los elementos previamente asignados y colocados en el grupo con el que guarden la mayor similitud posible. El centroide de cada grupo es calculado en función de los elementos que le componen.

El proceso de agrupamiento continúa determinando la distancia de todos los elementos con los centroides de cada grupo. Si un elemento es más cercano a un grupo que a otro, dicho elemento es reasignado y cada grupo debe calcular el nuevo centroide. Los pasos de comparación de distancias y la asignación de elemento se realizan hasta que no haya elementos por catalogar.

2.3. Métricas de distancia

El punto de partida de cualquier análisis de agrupamiento es la matriz de elementos. Cada elemento (fila) consta de ciertas características (columnas) representadas por un valor numérico.

Luego de decidir las variables que simbolicen cada elemento, es necesario seleccionar un método para determinar la distancia entre

los objetos. Las métricas de distancia contemplan ciertas reglas básicas:

- La distancia que existe entre dos objetos es mayor o igual que cero.
- La distancia entre los objetos A y B es la misma en ambos sentidos.
- Dos objetos son similares si el valor numérico de la distancia es cero.

La métrica de distancia es utilizada como una función de relación entre cualquier par de valores y establece la similitud entre los elementos. A continuación se detallan las métricas más utilizadas:

Distancia Euclidiana. Es una de las funciones de distancia más utilizadas y se expresa como la distancia entre dos elementos (*p* y *q*):

$$d_{p,q} = \sum_{i=1}^{n} \sqrt{(p_i - q_i)^2}$$

Formula 1.4.1 Expresión de la Distancia Euclidiana entre dos elementos (p y q)

Distancia Manhattan. Consiste en la distancia absoluta entre los elementos. La expresión matemática para representar la distancia que existe entre las características de dos elementos (*p* y *q*):

$$d_{p,q} = \sum_{i=1}^{n} |p_i - q_i|$$

Fórmula 1.4.2 Expresión de la Distancia Manhattan entre dos elementos (p y q)

2.4. Indexación y búsqueda de documentos

Con el paso del tiempo, la organización y acceso a la información ha evolucionado mediante técnicas de categorización, clasificación y agrupamiento con la aplicación de esquemas jerárquicos de contenidos.

Uno de los ejemplos más claros de organización de información es el esquema utilizado por el índice que se encuentra al final de los libros para acceder a un término en una ubicación específica. Esta metodología sirve para establecer relaciones entre conceptos aparentemente diferentes que pueden ser útiles para futuros lectores.

Desde la expansión del Internet, se han diversificado los métodos eficientes de búsqueda de información en colecciones extensas de

documentos. Por ejemplo, a través del mecanismo orientado a término de *índices invertidos* [7] se consigue acelerar las tareas de búsqueda de documentos.

Esta estructura de índices invertidos está compuesta por dos elementos: un *léxico* y una *lista de incidencias*. El léxico o vocabulario está compuesto por todos los términos en el contenido del documento. Por cada término se asocia una lista de documentos que lo contienen, generalmente se añade la posición que se encuentre en el contenido, a esta lista se la conoce como lista de incidencias. Esta estructura simplifica las búsquedas y facilita el acceso directo a documentos de acuerdo a los términos que contienen [26].

2.4.1. Lucene: API de Indexación y Búsqueda de Documentos

Lucene es un proyecto de Apache Software Foundation [19] que provee un API de indexación y búsqueda de documentos en texto plano.

Lucene utiliza una estructura de *índices invertidos* que consiste en un conjunto de términos que referencian al o los documentos que lo contienen.

El índice generado por Lucene se considera como un *Directorio* abstracto de un índice almacenado físicamente en disco o en RAM,

que contiene instancias de *Documentos* conformados por *Campos* que consisten en pares de nombre y valor. Las búsquedas implementadas con el API de Lucene sobre el *Directorio* que contiene los *Documentos* que se ajusten a un patrón de términos de búsqueda [19].

CAPÍTULO 3

3. METODOLOGÍA DE DESARROLLO

En este capítulo se detalla el análisis de los requerimientos de la solución planteada a partir de la cual, se planificó el diseño modular para el desarrollo del sistema.

3.1. Análisis de requerimientos

El sistema “BONSAI” debe cumplir con los siguientes requerimientos funcionales:

- Permitir la búsqueda de entradas de blogs por términos.
    - Organizar los resultados de búsquedas en grupos ó clústeres.
- Visualizar los blogs relacionados a los resultados de búsquedas.
    - Mostrar las entradas de blogs que se encuentren en los mismos grupos.

Con la implementación de estos requerimientos se brinda a los usuarios un medio efectivo de búsqueda de contenidos en la blogosfera de la ESPOL.

3.2. Descripción del desarrollo modular

El sistema “BONSAI” está compuesto por los siguientes componentes:

1. **Módulo de agrupamiento de contenidos.**

   El módulo de agrupamiento de contenidos es el encargado de procesar y agrupar las entradas de blogs de acuerdo a la similitud en su contenido.

   Los resultados obtenidos de este módulo son indexados para asegurar el rápido acceso a las recomendaciones de entradas relacionadas solicitadas por el módulo de indexación y búsqueda por términos.

2. **Módulo de indexación y búsqueda por términos.**

   El módulo de indexación y búsqueda por términos funciona como un motor de búsquedas sobre las entradas extraídas de la blogosfera politécnica a partir de los términos recibidos desde la interfaz Web.

   Además provee los blogs relacionados a los resultados de las búsquedas. Estas recomendaciones de entradas son obtenidas a partir de los resultados del módulo de agrupamiento de contenidos.

Estos módulos fueron diseñados para proveer una metodología para la recuperación, extracción y visualización de resultados de búsqueda de blogs; además, de la obtención de recomendaciones de entradas relacionadas.

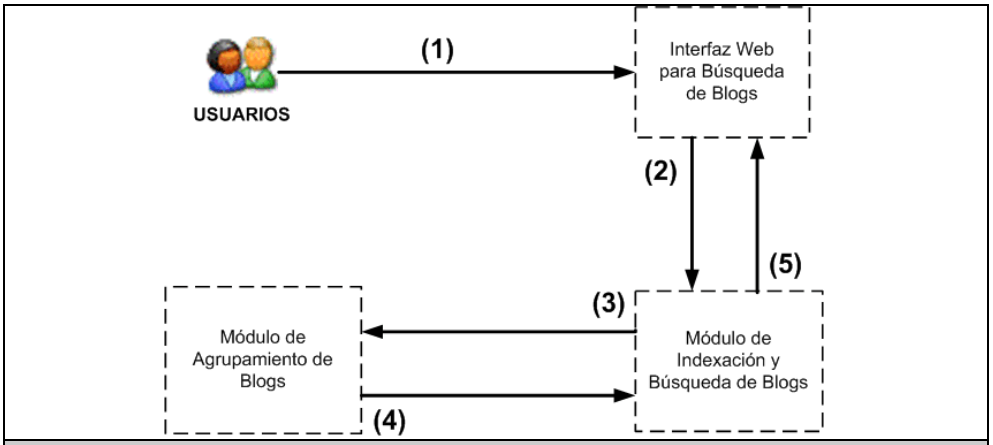

Figura 3.1. Diseño modular del sistema "BONSAI"

En la **figura 3.1**, se describe la interacción de los módulos del sistema: Primero, los usuarios ingresan los términos de búsqueda desde la interfaz Web del sistema **(1)**, dichos términos son enviados al módulo de indexación y búsqueda para obtener los blogs relevantes a dichos términos **(2)**.

Luego, a partir de las entradas resultantes de la búsqueda **(3)** se obtienen los blogs relacionados **(4)**. Finalmente, el resultado es procesado y visualizado en la interfaz Web **(5)**.

En las siguientes secciones se especifica en detalle el diseño e implementación de cada uno de los módulos que componen el sistema “BONSAI”.

3.2.1. Módulo de agrupamiento de contenidos

El propósito de este módulo consiste en obtener grupos de entradas relacionadas entre sí, de acuerdo a la similitud del texto (en lenguaje natural) en su contenido.

A continuación, se describen los pasos diseñados para obtener los grupos de entradas relacionadas entre sí:

1. De cada entrada se extraen un diccionario compuesto por términos (los cuales corresponden a las palabras separadas por espacios, signos de puntuación y caracteres especiales). Además, se eliminan los términos más comunes en el idioma español. Finalmente, se reducen términos de acuerdo a la similitud sintáctica de su raíz (computador, computadora, computar se agrupar por una sola raíz *comput-*).

2. Cada término extraído de la fase anterior es asociado con un factor de relevancia, el cual relaciona la frecuencia con la que aparece dicho término en un determinado documento con la frecuencia del término en la colección completa [30].

3. Cada documento es representado como un *vector de términos*, el cual consta de cada uno de los términos que lo componen, con su respectivo factor de relevancia obtenido en la fase anterior.

4. De la colección de documentos se extrae un diccionario de términos, del cual se elimina el 20% de los términos que se repiten con mayor frecuencia en su léxico, debido a que no son buenos discriminantes durante el proceso de extracción de la información [21].

5. En esta fase, se construye una matriz cuyas filas están compuestas por los vectores de términos (resultado de la fase 3) y las columnas están constituidas por cada uno de los términos de la colección (léxico de la fase 4).

6. La matriz de frecuencias de términos resultante de la fase anterior es ingresada a la implementación del algoritmo de agrupamiento seleccionado.

7. Finalmente, los resultados de la fase anterior son indexados para asegurar el rápido acceso a las recomendaciones de blog relacionados.

En la **figura 3.2** se describen los siete procesos Map/Reduce que implementan las fases de procesamiento de texto diseñadas para el módulo de agrupamiento de contenidos.

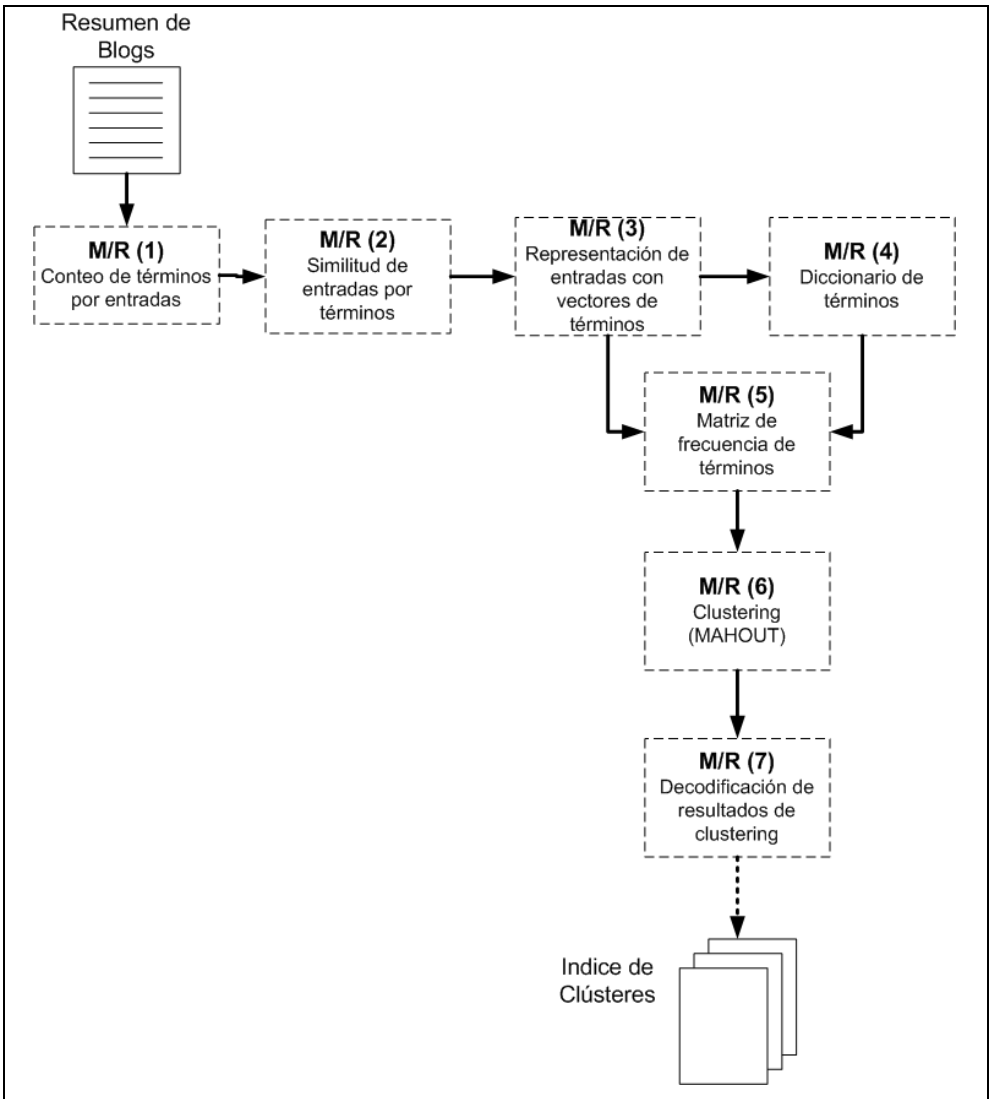

Figura 3.2. Diseño del Módulo de Agrupamiento de Contenidos

A continuación se describen cada unos de los procesos descritos en la **figura 3.2**, los cuales han sido etiquetados desde el M/R1 hasta el M/R7:

1. **Conteo de términos por entradas**

   El proceso M/R1 es una adaptación de un trabajo Map/Reduce propuesto como demostración en la clase CS/Info4300, de la Universidad de Cornell [22] [23]. En cada línea del archivo de entrada para este proceso se encuentra el contenido de cada entrada de blog.

   En la fase mapper, se contabiliza la frecuencia de cada término con la que aparecen en el documento; además de la longitud del futuro vector de términos que representará al mismo.

   Luego, en la fase reducer se contabiliza la frecuencia con la que aparece cada uno de los términos en la colección completa.

   En la **figura 3.3** se muestra el archivo resultante de esta fase, el cual presenta un esquema de índice invertido de cada uno de los términos en la colección de documentos.

Término	Identificador del documento	Frecuencia del término en el documento	Longitud al cuadrado del vector de términos	Frecuencia máxima del documento	Número de documentos en el que aparece	Número total de documentos

Figura 3.3. Estructura del archivo resultante del proceso de Conteo de términos por entradas

2. Similitud de entradas por términos

En el proceso M/R2, en lugar de procesar la colección completa de documentos para obtener grupos de entradas relacionadas entre sí [24], este proceso fue diseñado para crear grupos pequeños de entradas de blogs de acuerdo a la similitud de sus términos. Luego, estos grupos serán procesados minuciosa e independientemente para obtener clústeres de entradas con mayor nivel de cohesión entre ellas. De esta manera, se asegura la confiabilidad en las recomendaciones de entradas del sistema final.

Este proceso está compuesto por tres fases Map/Reduce. En la primera fase, cada término es asociado con su respectivo factor de relevancia, obtenido a partir de la siguiente expresión:

$$tfidf = \frac{\left(\frac{0,5 * f}{maxF}\right) + 0,5}{\sqrt{lSq}} \; x \; \log\left(\frac{nDocs}{df}\right)$$

Donde, *f* = Frecuencia del término en el documento.

maxF = Frecuencia máxima del documento.

lSq = Longitud al cuadrado del vector de términos.

nDocs = # total de documentos.

df = # de documentos en los que el término aparece.

En una segunda fase de este proceso, las entradas son agrupadas de acuerdo a la similitud de sus términos. Primero, se construye un índice invertido y cada término es asociado con una lista de identificadores de documento que lo contienen con su respectivo factor de relevancia (TF-IDF). Luego, iterando sobre la lista de identificadores por término, se obtienen tuplas de documentos las cuales se relacionan con el producto de sus correspondientes factores de relevancia. Estas tuplas representan la contribución individual por término para establecer la similitud entre los documentos (entradas de blogs) [23].

En la tercera fase del presente proceso, se suman las contribuciones individuales por término de la fase anterior para obtener las entradas de relacionados entre sí.

3. Representación de entradas con vectores de términos

En este proceso M/R3 se representan cada uno de los grupos de entradas previamente emparejadas como vectores de términos con su respectivo factor de relevancia (TF-IDF).

De cada documento se extrae la lista de términos representativos con su respectivo factor de relevancia, como se muestra en la **figura 3.4**.

<identificador_documento> <termino**1**>,<tf-idf**1**>;... <termino**N**>,<tf-idf**N**>

Figura 3.4. Estructura para representar los documentos como vector de términos

4. **Diccionario de términos**

El proceso M/R4 es el encargado de extraer los términos de las entradas a procesar, para construir un léxico acorde a los grupos de entradas previamente obtenidos.

5. **Matriz de frecuencia de términos**

En este proceso M/R5 se crea una matriz de frecuencia de términos de *m* filas (número de documentos pertenecientes a un determinado grupo) y *n* columnas (número de términos que componen el léxico del grupo a analizar), cuya intersección es el factor de relevancia correspondiente.

La dispersión que presenta la matriz de frecuencia resultante, se debe a que ciertos términos del léxico no están contenidos en los documentos a analizar.

En la **figura 3.5**, se muestra un ejemplo de la estructura de la matriz de frecuencia de términos.

	Término1	Término2	Término3	...	TérminoN
Documento1	0.0	0.23	0.11		0.0
Documento2	0.234	0.0	0.0		0.15
...					...
DocumentoN	0.0	0.56	0.45	...	0.86

Figura 3.5. Estructura ejemplo de una matriz de frecuencia de términos

6. Clustering (MAHOUT)

El proceso M/R6 es el encargado de realizar el agrupamiento de documentos mediante la implementación del algoritmo de clustering más conocido, K-means, de la librería MAHOUT [31] que implementa diversos algoritmos de aprendizaje de máquina bajo el paradigma Map/Reduce [8][9].

La implementación del algoritmo K-means de esta librería, selecciona aleatoriamente los *k* puntos centrales del espacio de vectores como puntos centrales de los clústeres. Luego, cada uno de los documentos es asignado a los puntos centrales más cercanos. Después de esto, por cada clúster se calcula un nuevo punto central a partir de los promedios de las características analizadas. Este proceso es ejecutado hasta que converge en un número finito de iteraciones.

El diseño para la implementación de este algoritmo, requiere de la matriz de frecuencias resultante de en el proceso anterior.

7. Decodificación de resultados del clustering

El proceso M/R7, luego de procesar el conjunto de vectores de términos con la implementación del algoritmo K-means de la librería MAHOUT, el archivo resultante es procesado en esta fase.

Tal como se muestra en la **figura 3.6**, posterior a cada vector de frecuencias de términos se coloca el índice del clúster al cual fue asignado el documento.

[, 0.0 , 0.15 , 0.34 , 0.0 , ... 0.0, 0.23, 0.1]	1
[, 0.20 , 0.15 , 0.34 , 0.20 , ... 0.0, 0.12, 0.0]	2
[, 0.0 , 0.0 , 0.04 , 0.0 , ... 0.0, 0.23, 0.1]	1
...	
[, 0.10 , 0.25 , 0.34 , 0.0 , ... 0.0, 0.53, 0.0]	1

Figura 3.6. Estructura del archivo final luego del procesamiento de la librería MAHOUT

Luego de decodificar el archivo de vectores de frecuencias, los resultados son indexados de acuerdo a los clústeres creados en la fase anterior.

3.2.2. Módulo de indexación y búsqueda por términos

Este módulo fue diseñado para las realizar las búsquedas de entradas en el índice obtenido de blogosfera politécnica, de acuerdo a los términos provistos por los usuarios en la interfaz del sistema.

Este módulo está compuesto por tres subcomponentes:

1. **Componente de búsqueda de entradas**

   Este componente se encarga de construir la cadena de consulta, compuesta por los términos de búsqueda enlazados por un operador lógico (*AND*).

   Esta cadena de búsqueda es procesada por la interfaz búsqueda de la librería Lucene [19] que realizará la búsqueda en el índice extraído de los blogs politécnicos.

2. **Componente de agrupamiento de resultados de búsquedas**

   En este componente se agrupan las entradas resultantes de la búsqueda por términos, mediante el API que provee la librería Carrot^2[32].

   La más importante característica de los algoritmos implementados por Carrot^2 es que ejecutan el agrupamiento de documentos en memoria.

3. **Componente de recomendaciones**

   El componente de recomendaciones obtiene del Módulo de Agrupamiento de Contenidos las entradas de blogs relacionadas de acuerdo a los resultados de la búsqueda.

3.2.3. Diseño de la interfaz de búsqueda

Para el diseño de la interfaz de búsqueda del sistema, se consideró las siguientes estrategias de interacción con el usuario:

- Los términos de búsqueda son ingresados en el estilo simple, en un formulario por completar.
- Los resultados son mostrados por grupos de entradas de acuerdo al procesamiento en memoria realizado por el módulo de indexación y búsqueda por términos. Para mejorar la navegación, los resultados son paginados en subgrupos de cuatro grupos de entradas etiquetadas.
- Junto a los grupos, se muestran las entradas similares obtenidas del componente de recomendaciones del módulo de indexación y búsqueda por términos.

En la **figura 3.7**, se muestra la interfaz de búsqueda del sistema BONSAI.

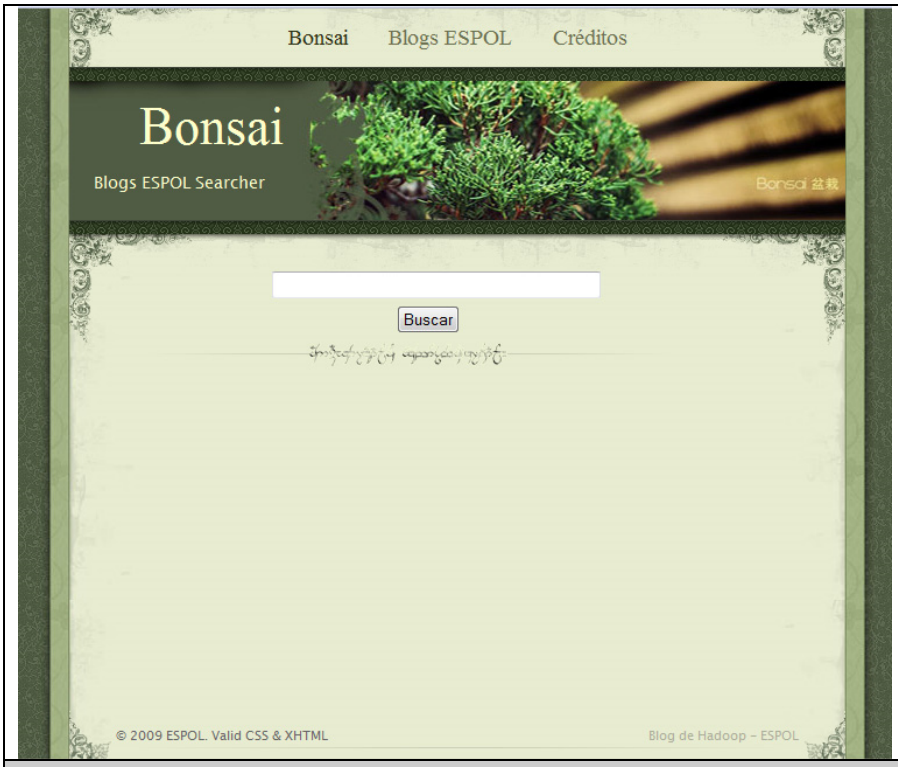

Figura 3.7. Interfaz de búsqueda del Sistema BONSAI

En la **figura 3.8** se muestra el resultado de la búsqueda de por términos, con las respectivas recomendaciones de entradas de blogs.

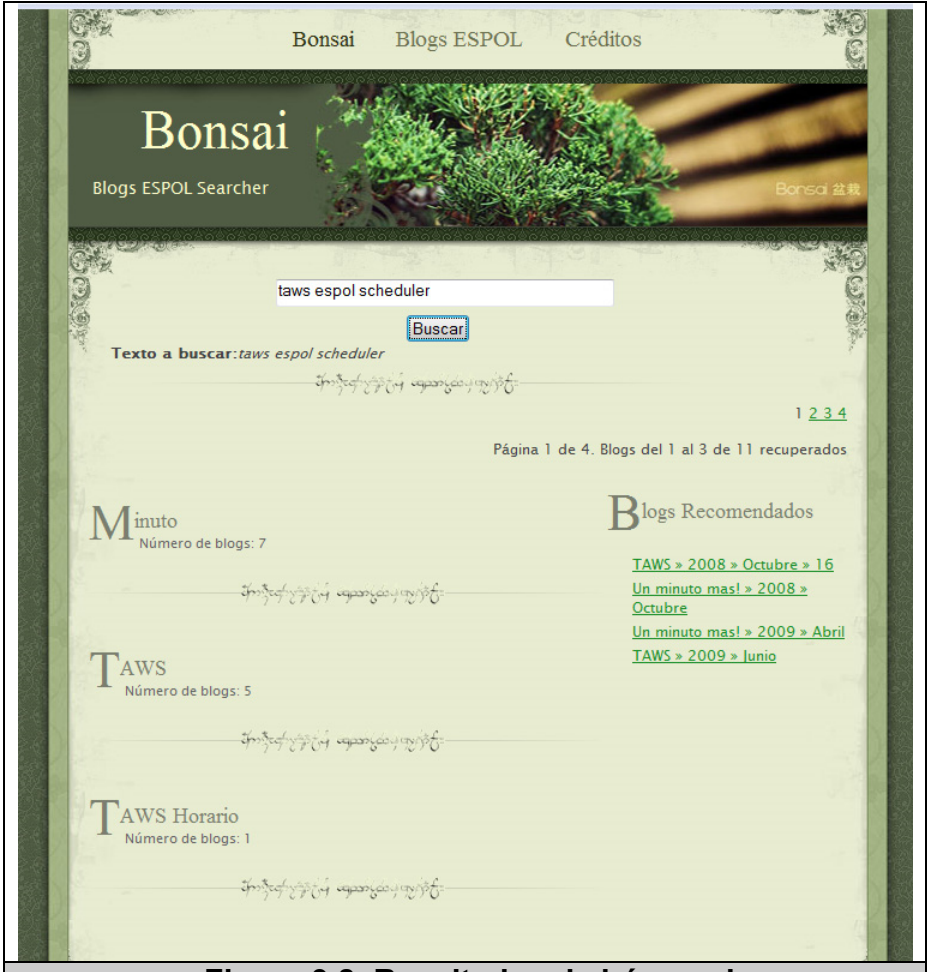

Figura 3.8. Resultados de búsquedas del sistema "BONSAI"

El resultado de las búsquedas se muestra los grupos etiquetados y el tamaño de cada uno de los grupos; junto a estos, se encuentran los grupos de blogs relacionados.

CAPÍTULO 4

4. PRUEBAS DEL SISTEMA

En el presente capítulo, se definen las pruebas de rendimiento a aplicar, para luego analizarlas y finalmente explicar los resultados obtenidos.

4.1. Pruebas de Rendimiento

Debido a que el objetivo principal del sistema consiste en la agrupación de entradas de blogs, se consideró someter a pruebas de rendimiento al Módulo de Agrupamiento de Contenidos.

El objetivo principal de estas pruebas es el de establecer la relación del tamaño de la colección de documentos a agrupar y el tiempo de procesamiento requerido.

4.2. Observaciones de las pruebas

Para la realización de estas pruebas, se utilizaron los recursos bajo demanda de Amazon con los Amazon Web Services, compuestos por diez nodos configurados con la plataforma de Hadoop, instalados con la distribución Fedora de Linux.

Las pruebas se realizaron desde el proceso Map/Reduce de obtención de entradas similares por términos (2) hasta el procesamiento de Clustering-MAHOUT (6)

Procedimiento

1. Se obtienen los grupos de entradas similares por términos (segunda fase del Módulo de Agrupamiento de Contenidos).
2. Cada uno de los grupos es representado como un vector de términos.
3. Se extrae el diccionario de términos del grupo de entradas.
4. A partir de los pasos 2 y 3, se crea la matriz de frecuencia de términos.
5. La matriz de frecuencia del paso anterior pasa al proceso de Clustering de MAHOUT.

Este procedimiento es aplicado por los grupos de entradas similares de tamaño más representativos.

4.3. Resultados

Como se puede observar en la **figura 4.1**, se obtuvieron cinco (5) grupos de entradas de blogs relacionados con el propósito de verificar el rendimiento del algoritmo aplicado en el contenido de las entradas.

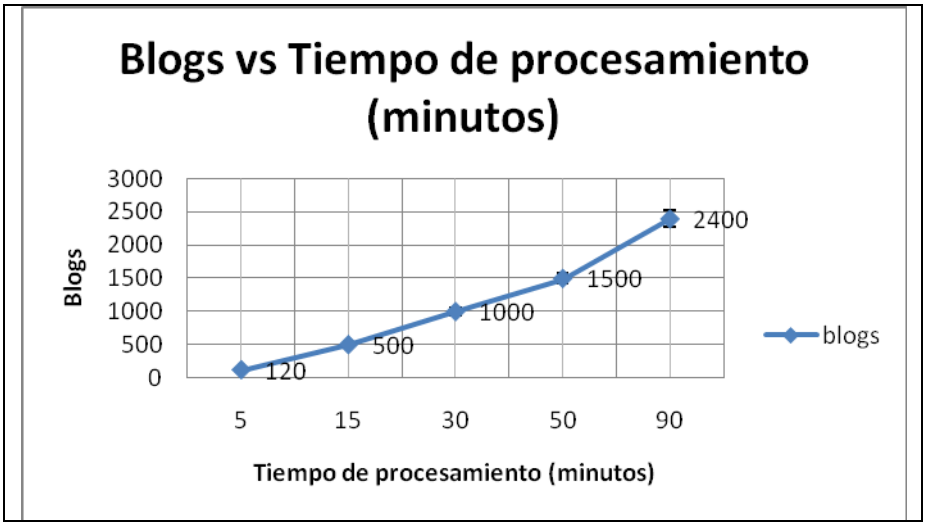

Figura 4.1. Gráfico comparativo de grupos de blogs analizados versus tiempo de procesamiento (minutos)

De acuerdo a los resultados de las pruebas, se puede observar que entre los grupos de entradas y el tiempo de procesamiento existe una relación lineal.

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

1. Debido a que el paradigma Map/Reduce está diseñado para procesar datos con una estructura definida, resulta fácil implementar tareas que permitan extraer información relevante de documentos escritos en lenguaje natural.

2. Implementar las técnicas de extracción de información con el paradigma de procesamiento distribuido Map/Reduce, permite realizar tareas de procesamiento con grandes volúmenes de datos.

3. Debido a que las técnicas de agrupamiento sirven para obtener una visión general de la composición de una colección de objetos, resulta conveniente implementarlas bajo el paradigma Map/Reduce para comprender otras características de las comunidades digitales: cohesión de los miembros, el impacto de tópicos específicos, entre otros.

4. Mediante la visualización de grupos etiquetados de entradas de blogs, es posible realizar inspecciones rápidas de los resultados de búsquedas.

5. El servicio de computación en la nube que ofrece Amazon, a través de sus Amazon Web Services, permite ejecutar tareas de procesamiento de texto en lenguaje natural bajo el esquema de distribución de tareas en equipos de propósito general, en lugar de requerir de equipos costosos de alto rendimiento.

RECOMENDACIONES

1. Debido al tamaño de la colección de datos a procesar, el retardo promedio esperado para visualizar los resultados y los recursos computacionales deben considerarse como factores decisivos al determinar si un proceso Map/Reduce debe ejecutarse por cada requerimiento de los usuarios o como un tarea secundaria que no afecte el funcionamiento del sistema.

2. Para asegurar la confiabilidad de las recomendaciones de entradas de blogs, es aconsejable realizar un procesamiento previo en el que se obtengan grupos de entradas relacionadas de acuerdo a los términos en su contenido tal como se planteó para el presente trabajo.

3. Al utilizar una plataforma de descarga y análisis de contenidos de páginas Web, como Nutch, se deben refinar los algoritmos de análisis de direcciones Web, para evitar extraer aquellas direcciones que contengan código embebido (javascript) y otros tipos de direcciones con información irrelevante de un blog.

4. Previo a la puesta en producción de la presente solución, es recomendable diseñar un plan de actualizaciones del índice construido a partir de las entradas extraídas de la blogosfera politécnica, en la cual se debe considerar el tiempo de vida promedio de los blogs, la frecuencia de publicación de entradas y los recursos computacionales que se le asignen.

REFERENCIAS BIBLIOGRÁFICAS

[1] **PORTER, JOSUA,** Designing for the Social Web, NEW RIDERS, 2008, 97 p.

[2] **"BLOG DE ESPOL DIRECTORIO"**, Blog de Espol, <http://blog.espol.edu.ec/directorio/>, Julio 2009.

[3] **GANTZ**, The Diverse and Exploding Digital Universe, <http://www.emc.com/collateral/analyst-reports/diverse-exploding-digital-universe.pdf>, Marzo 2008.

[4] **SALA DE PRENSA,** Facebook | Estadísticas, <http://www.facebook.com/press/info.php?statistics>, Julio 2009.

[5] **DEAN JEFF Y GHEMAWAT SANJAY**, MapReduce: Simplified Data Processing on Large Clusters, en Sixth Symposium on Operating System Design and Implementation (OSDI'04), San Fransisco CA, Diciembre 2004.

[6] **"WELCOME TO APACHE HADOOP!"**, Hadoop, <http://hadoop.apache.org/>, Julio 2009.

[7] **"INVERTED INDEX - WIKIPEDIA, THE FREE ENCYCLOPEDIA"**, Inverted index, <http://en.wikipedia.org/wiki/Inverted_index>, Julio 2009.

[8] **CHU CHENG-TAO ET AL.**, Map-Reduce for Machine Learning on Multicore, En The Neural Information Processing Systems (NIPS), 2006, pp. 281-288.

[9] **BISCIGLIA CHRISTOPHE, KIMBALL AARON, Y MICHELS-SLETTVET SIERRA ET AL.**, Distributed Computing Seminar: Lecture 5 Graph Algorithms & PageRank, Introduction to Problem Solving on Large Scale Clusters, 2007.

[10] **BARNATT CHRISTOPHER**, "ExplainingComputers.com by Christopher Barnatt", Cloud Computing, <http://www.explainingcomputers.com/cloud.html>, Julio 2009.

[11] **PAUL GRAHAM**, Lisp, <http://www.paulgraham.com/lisp.html>, Julio 2009.

[12] **"HADOOP"**, Notas escritas por Engeneering @ Facebook, <http://www.facebook.com/note.php?note_id=16121578919>, Julio 2009.

[13] **"HIVE – A PETABYTE SCALE DATA WAREHOUSING USING HADOOP"**, Notas escritas por Engeneering @ Facebook, <http://www.facebook.com/note.php?note_id=89508453919#/note.php?note_id=89508453919> Julio 2009.

[14] **WHITE TOM**. Hadoop: The Definitive Guide, O'REILLY, 2009, 11 pp.

[15] **VENNER JASON**. Pro Hadoop, APRESS, 2009, 4-6 pp.

[16] **BORTHAKUR DHRUBA**, "HDFS Architecture", HDFS Architecture, <http://hadoop.apache.org/common/docs/r0.16.4/hdfs_design.html>, Julio 2009.

[17] **MOENS MARIE-FRANCINE**, Automatic Indexing and abstracting of document texts, KLUWER ACADEMIC PUBLISHERS, 2002, 10-15 pp.

[18] **MANNING CHRISTOPHER, RAGHAVAN PRABHAKAR, SCHÜTZE HINRICH**, Introduction to Information Retrieval, CAMBRIDGE UNIVERSITY PRESS, 2008, 1-10 pp.

[19] **"WELCOME TO LUCENE!"**, Welcome to Lucene!, <http://lucene.apache.org/>, Julio 2009.

[20] **MYATT G., JOHNSON W**, Making sense of data II, WILEY, 2009, 67-110 pp.

[21] **BAEZA-YATES R., RIBEIRO-NETO B**, Modern Information Retrieval, ACM PRESS, 1999, 167 p.

[22] **WILLIAM Y. ARMS Y BLAZEJ J. KOT**, Indexer, <http://www.infosci.cornell.edu/courses/info4300/2008fa/Indexer.txt>, Octubre 2008.

[23] **WILLIAM Y. ARMS AND BLAZEJ J. KOT**, WordinDoc, <http://www.coursehero.com/file/1550104/WordinDoc/>, Octubre 2008.

[24] **ELSAYED T., LIN J., DOUGLAS W.**, Pairwise Document Similarity in Large Collections with MapReduce, en *Proceedings of ACL-08: HLT, Short Papers (Companion Volume)*, Columbus, Ohio, USA, Junio 2008, 265-268 pp.

[25] **WEISS DAWID**. Massive Distributed Processing using Map-Reduce, Institute of Computing Science, Pozna University of Technology. Enero 2007.

[26] **BAEZA-YATES R., RIBEIRO-NETO B**, Modern Information Retrieval, ACM PRESS, 1999, 192 p.

[27] **GHEMAWAT SANJAY, GOBIOFF HOWARD Y LEUNG SHUNG-TAK**, The Google File System, en 19th ACM Symposium on Operating Systems Principles, Lake George NY, Octubre 2003.

[28] **DITTUS MARTIN**, "Hadoop usage at Last.fm", Hadoop usage at Last.fm,<http://skillsmatter.com/custom/presentations/martin_dittus_hadoop_at_last.fm_overview.pdf>, Septiembre 2009.

[29] **"SELF-SERVICE, PRORATED SUPER COMPUTING FUN!"**, Self-service, Prorated Super Computing Fun!, <http://open.blogs.nytimes.com/2007/11/01/self-service-prorated-super-computing-fun/>, Septiembre 2009.

[30] **"TF-IDF"**, tf-idf, <http://en.wikipedia.org/wiki/Tf–idf>, Septiembre 2009.

[31] **"MAHOUT"**, Apache Mahout – Overview, <http://lucene.apache.org/mahout/>, Septiembre 2009.

[32] **"CARROT^2 – OPEN SOURCE SEARCH RESULTS CLUSTERING ENGINE"**, Carrot^2, <http://project.carrot2.org/>, Septiembre 2009.

[33] **"AMAZON WEB SERVICES"**, Amazon, <http://aws.amazon.com/>, Septiembre 2009.