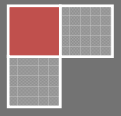
Motores de Búsqueda

Arquitectura de Google

Proyecto de Investigación sobre Motores de Búsqueda Web, específicamente la Arquitectura de Google. Hecho para la clase de Tecnologías Emergentes, cursada el primer trimestre del año 2009.

Fernando David Irías Escher 10611039

3/15/2009



Contents

INTRODUCCIÓN

¿Qué sería él internet si no existieran los Motores de Búsqueda Web? ¿Puede el amable lector imaginarse intentar navegar sin tener una herramienta que le permita buscar la información puntual que necesita en el momento?

Hoy por hoy, las páginas más visitadas en el mundo son los Motores de Búsqueda. Solo tenemos que ver nuestros hábitos cibernéticos para encontrarnos a nosotros mismos escribiendo una cuantas palabras y presionando un botón con etiqueta “Buscar con Google” (“Google Search” en inglés) o “Voy a Tener Suerte” (“I’m Feeling Lucky” en inglés).

No ha sido mera casualidad mostrar esta obvia referencia al Motor de Búsqueda predilecto a nivel mundial, Google. Ha sido tanto el auge y la mística que se ha formado en torno a este sitio, que la palabra “googlear” ya es aceptada como una palabra más del idioma español para muchos usuarios del internet [WIK09].

El presente proyecto consiste en hacer un análisis interno a la estructura de los Motores de Búsqueda de hoy en día. El documento hará énfasis en la estructura de Google, ya que este fue el pionero en la mejora de la estructura para obtener resultados de mayor calidad. Muchos Motores de Búsqueda de bastante popularidad utilizan actualmente varias de las técnicas que se expondrán en las siguientes páginas.

ANTECEDENTES

A principios de los años ’90 el internet comenzó a ganar popularidad entre los usuarios caseros con computadoras personales. El internet se empezó a poblar rápidamente de información y de usuarios. Este crecimiento constante atrajo la necesidad de buscar información en internet y obtener un uso más dinámico de esta. Ideas se comienzan a gestar entorno a esta preocupación.

Varios Motores de Búsqueda (MB) comenzaron a aparecer, pero no fue hasta finales del año 1993 en que se hizo su aparición un MB llamado World Wide Web Worm (WWWW) [MCB94]. Este MB obtenía el título y el encabezado de las páginas web las obtenía haciendo una búsqueda lineal. No muy efectivo, ¿no?

Un año más tarde aparecería lo que es el primer Rastreador Web (Web Crawler en inglés). Este tenía por nombre Web Crawler[1] y su tarea primordial era buscar páginas en internet e indexarlas. Durante este años alcanzó bastante popularidad.

Para 1995 comenzaría a aparecer propuestas bastante interesantes. AltaVista y Yahoo! ya están consolidados como los principales actores en este rubro. Ambos son pioneros en el uso del lenguaje natural para realizar los queries que les aplican sus usuarios.

Durante todos estos años, estos buscadores se enfocaron en la rápida obtención de los resultados, algo que lograron concretar a un nivel admirable, pero no fue hasta el año de 1997 cuando se empieza a forjar el futuro de la navegación web en la Universidad de Stanford. Sergey Brin y Lawrence Page, notan que los MB que existían hasta el momento, no cumplían con el principal objetivo: La calidad de la búsqueda.

“Es un hecho que para Noviembre de 1997, solo uno de los cuatro Motores de Búsqueda comerciales era capaz de encontrarse a sí mismo (retornaba su propia página en respuesta a su nombre en los primeros diez resultados)”. [BRI97]

Este problema tenía que ser solucionado lo antes posible y sería Brin y Page los que llevarían esta bandera. Para el año de 1998 ya tenían un prototipo funcional de lo que ahora conocemos como el Motor de Búsqueda más popular que existe.

En las páginas siguientes, presentaremos la propuesta de Brin y Page en 1998.

[1] http://www.webcrawler.com/

OBJETIVOS DE GOOGLE

La idea de formar Google nace con la necesidad de obtener mejores respuestas de los Motores de Búsqueda. Hasta 1998 ya se había logrado el objetivo de hacer Motores de Búsqueda rápidos, era tiempo de enfocarse en la calidad.

Brin y Page, se dieron cuenta de que solo indexar las páginas no es la solución a una mejor búsqueda. Ocurría muy a menudo que los resultados realmente relevantes en una búsqueda, eran depreciados por resultados mediocres, ¿Por qué este fenómeno?

"Una de las causas principales de este problema es que el número de documentos en los índices se ha incrementado en varios órdenes de magnitud, pero la habilidad de los usuarios para buscar estos documentos ¡No!". [BRI97]

¿Quién de nosotros continua buscando después de los primeros diez resultados de una búsqueda? Realmente muy pocos, damos por fallida una búsqueda si nuestro resultado no está en estos primeros resultados. Dada esta necesidad, el avance en los Motores de Búsqueda se debe enfocar en poner entre las primeras diez páginas lo que el usuario promedio está buscando.

Toda esta necesidad de interacción con el usuario promedio, presenta un nuevo objetivo que debe ser alcanzado: Construir un sistema que la mayoría de las personas puedan utilizar sin problemas.

La arquitectura de Google está diseñada para guardar todos los documentos que se encuentren en el rastreo. Esto tiene un propósito secundario de auxiliar a otros investigadores para que puedan llegar rápido al servidor, procesar datos y obtener resultados interesantes que habría sido difícil de recolectar sin la existencia de esta base de datos.

Características Principales

Para poder cumplir los objetivos que se esperan con Google, se necesita establecer qué es lo que ayudará a toda la estructura a cumplir con estos menesteres. Así que a grandes rasgos, podemos notar dos características que son primordiales para hacer la diferencia entre los motores de búsqueda que existen a la fecha de su nacimiento. Estas características son: PageRank y Texto Anclado.

PageRank

El indexado que implementan varios Motores de Búsqueda pre-Google hace su trabajo bastante bien, el problema reside en que no se le da cierta prioridad a algunas páginas, por lo tanto no tenemos una base para afirmar que nuestra búsqueda es realmente eficaz.

El PageRank viene a solucionar este problema con una audacia admirable. El mismo Lawrence Page tenía la frase "PageRank: Trayendo Orden A La Web" como mensaje de bienvenida en su página de la Universidad de Stanford.

El PageRank utiliza una maraña de mapas para poder calcular la prioridad que debería tener una página especifica, y así, poder saber cuáles deberían estar entre las primeras diez.

Cálculo de PageRank

Podemos ver el PageRank como un gobierno semidemocrático. Si deseamos que la página X gane las elecciones, primero debemos buscar a sus votantes. En este caso los votantes son las páginas que tienen links (que apunta) a esta página X. Si la página X tiene links que apuntan a otras páginas, esto también le dará un poco de peso a su campaña. Pero recordemos que esta no es una democracia total, así que también tomaremos en cuenta el estatus social de cada votante, esto quiere decir que si la calidad (PageRank) de un votante influye en su voto, así de esta manera el voto de una página de alta calidad tendrá más peso que el de una página mediocre.

Ahora bien, veamos la fórmula que se utiliza para hacer el cálculo del PageRank:

$$PR(X) = (1 - d) + d(\frac{PR(V_1)}{C(V_1)} + \ldots + \frac{PR(V_n)}{C(V_n)})$$

En la fórmula anterior, podemos ver como se calcula el PageRank de una página X. Las páginas V_1 hasta V_n son las páginas que apuntan a X (páginas votantes). Tenemos también un valor d que nos dará una mejor estimación del PageRank. Básicamente d nos ayudará a que páginas que solo están citadas por una página de mucho peso, no se queden atrás con respecto a otras páginas que están citadas por muchas páginas de peso medio. $PR(V_i)$ es el PageRank de la página V_i. $C(V_i)$ es una función que indica el número de páginas saliendo de V_i.

"La probabilidad de que un navegador aleatorio visite una página es su PageRank. Y, el factor de damping d es la probabilidad en cada página de que el navegador aleatorio se aburrirá y seleccionará una nueva página." [BRI97]

Uno de los problemas encontrados en Enero de 2007 con este sistema de ranking, fue que era posible burlar al algoritmo con el objeto de poner resultados espurios en los primeros lugares de la búsqueda. Esto se obtenía agregando el enlace de la página que se quiere insertar como resultado espurio, a la mayor cantidad de páginas posible, así de esta manera, se puede llegar a obtener una prioridad lo suficientemente alta como para posicionarse en los primero lugares de las búsquedas. Para más información sobre el Google Bomb se puede referir a [WKP09].

TEXTO ANCLADO

El texto anclado, es el texto al que se le hace clic cuando se hace clic a un hyperlink. No es el texto del link, si no la máscara que hace más descriptivo este link.

"Primero, las anclas a menudo proveen una descripción más acertada de las páginas web que las páginas mismas. Segundo, las anclas pueden existir para documentos que no pueden ser indexados por un Motor de Búsqueda basado en texto, como ser imágenes, programas, y bases de datos". [BRI97]

La cita anterior describe rápidamente la importancia de indexar el texto anclado. El indexar el texto anclado, da pie a una gama mayor de búsqueda ya que no solo se restringe a texto, sino que también a objetos con estructuras binarias.

A pesar de todas las maravillas y bondades que esto presenta, se tienen dos problemas mayores:

1. Es posible que se retorne páginas que no han sido indexadas, por lo tanto es posible que nunca hayan sido revisadas. En este caso, puede suceder que la página

nunca existiera, pero este problema puede llegar a ser solucionado si ordenamos los resultados en base a PageRank.

2. Es muy difícil implementar el uso de las anclas eficientemente ya que pueden apuntar a datos que son bastante grandes. Estos datos deben ser procesados.

ESTRUCTURAS EN GOOGLE

Ahora daremos un rápido vistazo a cada una de las estructuras que conforman la arquitectura del sistema. Más adelante, veremos cómo interactúan estas partes para poder resolver el resultado de una búsqueda.

REPOSITORIO

El repositorio es la estructura de almacenamiento de las páginas web HTML que han sido rastreadas. Estas páginas se comprimen en formato zlib, esto para obtener un balance entre nivel de compresión y velocidad.

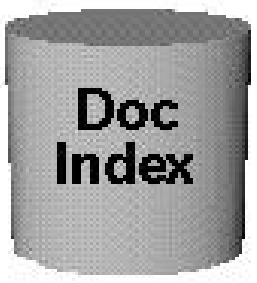
Las páginas HTML se guardan secuencialmente y se les agrega tres campos más: El docID, el tamaño y la URL. En la Figura 1 se presenta una representación de cómo es la estructura de un paquete comprimido.

Paquete (Comprimido en el repositorio)

docid	ecode	urllen	pagelen	url	page

Figura 1

ÍNDICE DE DOCUMENTO



El índice de documento tiene la información sobre cada uno de los documentos. Sus campos son de longitud fija de acceso secuencial. Está ordenado por docID.

Entre la información que contiene cada uno de los campos tenemos: estado del documento, puntero al repositorio, la suma de verificación (checksum), y otras estadísticas. Si el archivo ya ha sido rastreado, también tenemos un puntero a un archivo llamado docinfo que contiene la URL y el titulo de la página. En caso de no haber sido rastreado aún, solo hay un puntero a la URLlist que contiene solamente la URL de la página.

Dentro del índice de documento, también tenemos un archivo que tiene una lista de sumas de verificación junto con su docID correspondiente. Este archivo está ordenado por las sumas de verificación. Cuando obtenemos una URL, se calcula su suma de verificación y se hace una búsqueda binaria de si docID.

LEXICON

El Lexicon contiene un diccionario de palabras. Todas están concatenadas juntas y separadas por nulls. También contiene una tabla hash de punteros. Contiene información extra de las palabras para cualquier uso que se le pueda dar.

HIT LISTS

Esta es una lista de ocurrencias de una palabra en un documento específico. Su estructura conserva información sobre la posición, letra, y la capitalización. Para ahorrar espacio, esta estructura pasa por un proceso de compactación llamado Hand-Optimized Compact. Este tipo de compactación permite dejar una estructura bastante pequeña y con poca manipulación de bits (para mejorar la velocidad). Se utilizan dos bytes para representar cada hit.

Hay dos tipos de Hits: Los Plain y los Fancy. Los Fancy son considerados cuando se encuentra la palabra en un URL, titulo, ancla o metadata. El resto de ocurrencias se toman como Plain hits.

Plain Hit

En este tipo de hit, utilizamos un bit para la capitalización, tres bits para el tamaño letra (solo acepta 7 valores, el octavo es del Fancy Hit) y 12 bits más para la posición. En la Figura 2 mostramos la estructura del Plain Hit.

Estructura del Plain Hit

cap: 1 bit	imp:3 bits	posición: 12 bits

Figura 2

Fancy Hit

Los Fancy Hits contiene el bit de capitalización, el tamaño de la fuente en con el valor 7 (para identificarlo como Fancy Hit), cuatro bits para sabe qué tipo de Fancy Hit es, y los ocho restantes para la posición. La estructura la podemos apreciar en la Figura 3.

Estructura del Fancy Hit

cap: 1 bit	imp = 7	tipo: 4	posicion: 8 bits

Figura 3

Cuando los Fancy Hits se encuentran en un ancla, la estructura cambia un poco. Los ocho bits de la posición se dividen en dos partes: cuatro bits para la posición de la ancla en el documento y cuatro bits para un valor hash que indica el docID donde se encontró el ancla. Vemos la estructura del Anchor Hit en la Figura 4.

Estructura del Anchor Hit

cap: 1 bit	imp = 7	tipo: 4	hash: 4 bits	pos: 4 bits

Figura 4

Índice

Los Índices están parcialmente ordenados en varios barriles. Cada barril acepta un rango de wordID. Así que si un documento tiene ocurrencias de una palabra, el docID de ese documento se grabará en el barril que corresponde a esa palabra. Después del docID se almacena la lista de wordID junto con la Hit List de esa palabra.

Google utiliza 64 barriles para manejar estas wordID. El wordID no se guarda como tal, sino que se almacena con un numeró que indica la diferencia entre el mínimo wordID que cae en ese barril. De esta forma se pueden representar los wordID con solo 24 bits. La estructura del barril está representada en la Figura 5.

Estructura Barril de Índice

docID	wordID: 24 bits	nhits: 8 bits	hit hit hit hit
	wordID: 24 bits	nhits: 8 bits	hit hit hit hit
	null wordID		
docID	wordID: 24 bits	nhits: 8 bits	hit hit hit hit
	wordID: 24 bits	nhits: 8 bits	hit hit hit hit
	wordID: 24 bits	nhits: 8 bits	hit hit hit hit
	null wordID		

Figura 5

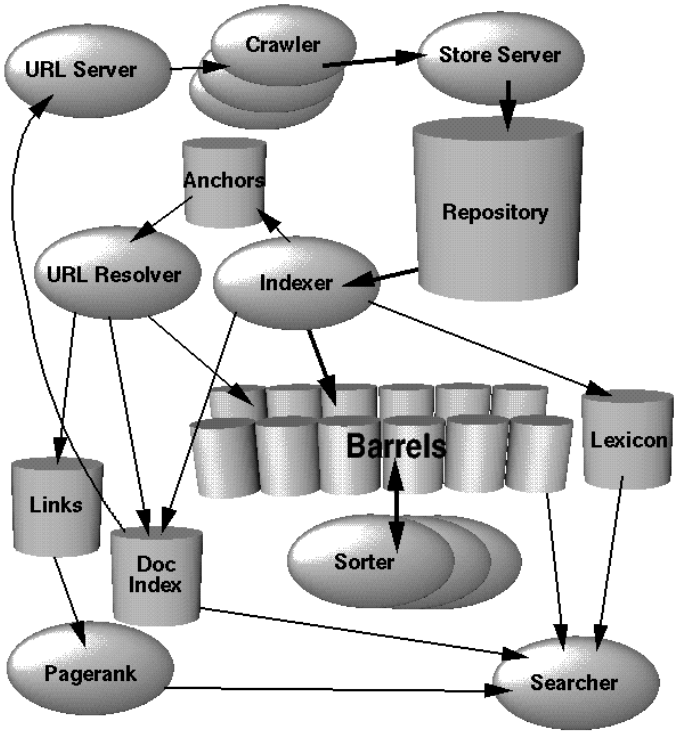
ÍNDICE INVERTIDO

Estos índices son el producto de un barril que ha sido ordenado por el Ordenador. El Lexicon contiene punteros hacia los barriles donde cae cada palabra. Por lo tanto podemos concluir que éste apunta a una lista de docID con su Hit List correspondiente.

Para el fácil y rápido manejo de estos índices invertidos, se mantienen dos sets de barriles: Un set para las Hit Lists que contienen Anchor Hits o Fancy Hits de título, y el otro set es para todas las demás Hit Lists.

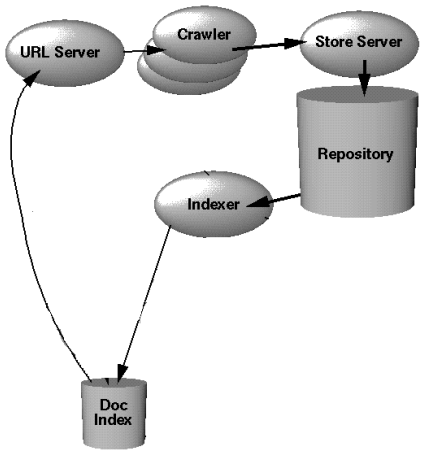
Primero se busca en el primer set de barriles para encontrar las concurrencias más relevantes, si esto no presenta suficientes resultados, se busca en el segundo set.

ARQUITECTURA: EL TODO



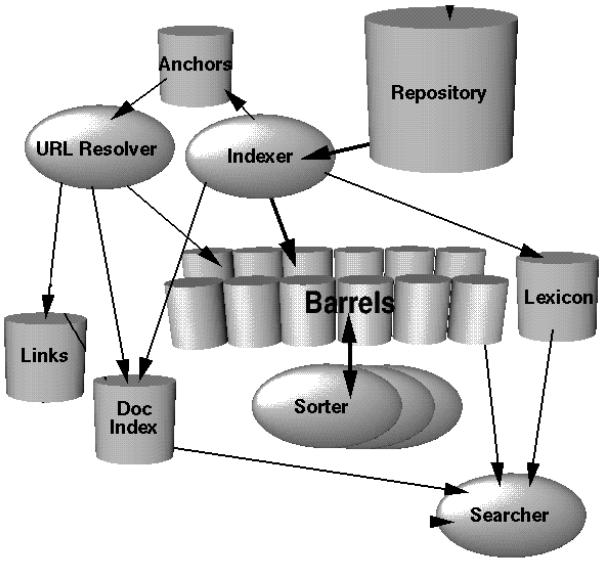
Ahora bien, es tiempo de unir todas las estructuras antes vistas y darles forma. A grandes rasgos, el proceso general se puede dividir en varios subprocesos: El Rastreo, La Indexación, La Clasificación y La Búsqueda.

EL RASTREO



El proceso de rastreo comienza por la captura las primeras URL por parte de los Rastreadores. Google normalmente utiliza tres rastreadores que mantiene abiertas más de 300 conexiones. Estos Rastreadores llegan hasta los servidores y obtiene las páginas web en HTML. Luego, mandan estas páginas al Servidor de Guardado (Store Server), el cual las comprime y las guarda en el repositorio. Una vez que el indexador ha analizado estás páginas (se explicará a profundidad más adelante) y generado el docID del documento, el URL Server se encarga de obtener las nuevas URL del Índice de Documento. Estas nuevas URL son adheridas los Rastreadores para luego se rastreadas.

LA INDEXACIÓN



El Indexador es el proceso más pesado en el sistema de Google. Empieza por leer y descomprimir los archivos dentro del Repositorio. Luego parsea el HTML y comienza a convertir cada palabra en Hits. Estos Hits guardan toda la información que necesita la estructura, y las palabras son mandadas al Lexicon.

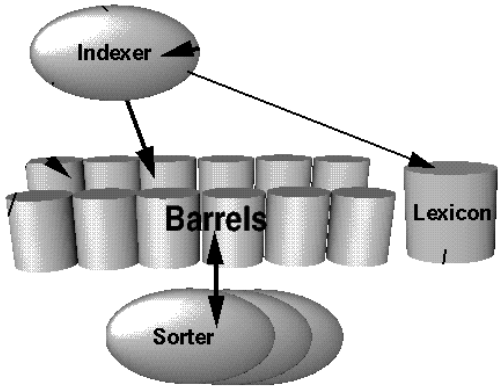
Para hacer el conteo de las palabras, el Indexador se vale de un conjunto de funciones que llevan por nombre MapReduce. La Función Map se encarga de tomar las palabras y emitirlas junto con una inicialización de una (1) ocurrencia. La función Reduce toma cada aparición de una palabra y la reduce sumando todas sus apariciones en el mapa. Lo bueno de estas funciones es que se pueden ejecutar en paralelo, así que el tiempo se recorta a un poco más de la mitad. Para más información sobre esta función puede consultar a [DEA08].

Luego de haber barrido todo el documento, el Indexador distribuye todos los hits en los Barriles (Barrels). En paralelo, el Indexador parsea todos los links en la página que procesó y guarda toda la información necesaria en el archivo de Anclas.

El URLresolver lee el archivo de anclas y convierte cada link en un link absoluto (el link definitivo que apunta a la página HTML). A estos URL absolutos, les aplica una técnica de hash para encontrar su docID (como se explicó en la estructura del Índice de Documento). A continuación, se encarga de distribuir la información recolectada en los sitios pertinentes: Pone el Texto Anclado en el Índice asociándolo con el docID al que éste apunta, y genera una base de datos de links la cual será usada para calcular el PageRank para cada documento.

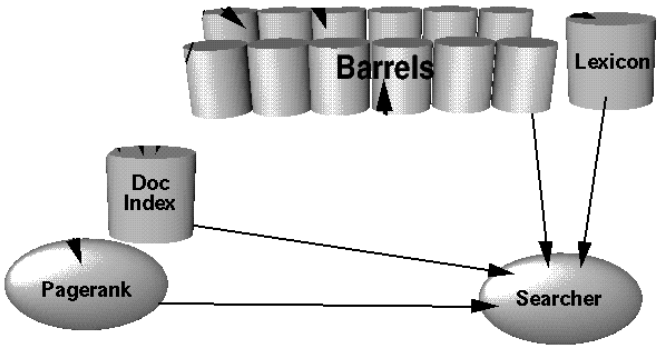
Para conocer a profundidad la matemática de vectores en el indexado, se recomienda leer [SAL75].

LA CLASIFICACIÓN



La Clasificación es la parte donde se genera el Índice Invertido. El Ordenador (sorter) ordena lo barriles por wordID (originalmente semiordenados por docID). Luego, se produce una lista de wordID y offsets en la lista invertida. Un programa llamado DumpLexicon toma el léxico generado por el Indexador, y las cruza con estas listas, generando un nuevo léxico que podrá usar el Buscador.

LA BÚSQUEDA



En Brin y Page proponen los siguientes pasos para la búsqueda en [BRI98]:

1. Se parsea el query.
2. Se convierten cada palabra en un wordID.
3. Se busca en el inicio de cada docList en los barriles pequeños cada una de las palabras.
4. Se sigue buscando por toda la lista hasta que se encuentre un documento que contenga todas las palabras.
5. Se computa la prioridad para ese documento según el query.
6. Si estamos en el barril pequeño y estamos al final de la docList, se busca en los barriles grandes por cada palabra y se retorna al paso 4.
7. Si no hemos llegado al final de ninguna docList, regresar al paso 4.
8. Se sortean los resultados por prioridad y se muestran los primeros k elementos.

El cálculo de la prioridad se vuelve un problema matemático bastante complejo. Se pueden dar dos casos con diferente complejidad al momento de parsear el query:

1. Si consideramos el caso más sencillo, una sola palabra en el query, Google buscará en la Hit List de cada documento por esa palabra. Cada hit contiene su propio tipo, y cada tipo tiene su valor. El Tipo-Peso hace un vector indexado por tipo. Google contará el número de hits del tipo en la Hit List y así obtendremos otro vector Cantidad-Peso. Si hacemos el producto punto de ambos vectores, habremos computado un puntaje llamado IR para este documento. Finalmente este IR se combina con el PageRank para dar la prioridad final al documento.

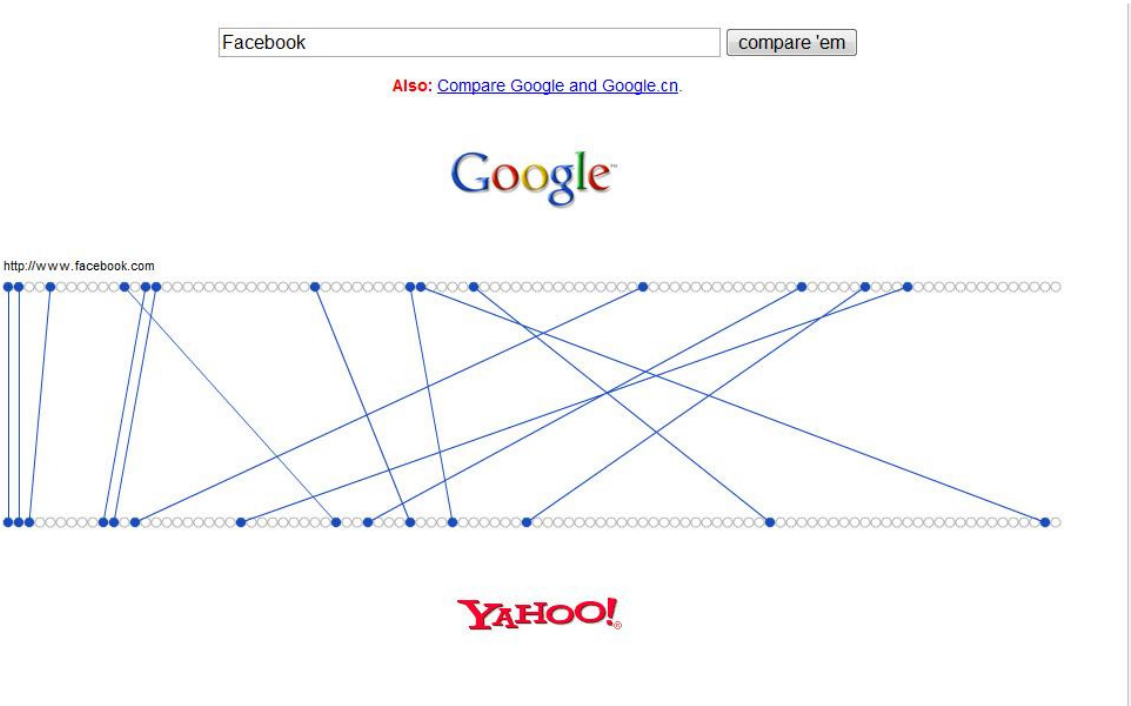
2. Cuando son dos o más palabras, la cosa se complica aún más. Ahora se deben revisar múltiples Hit List al mismo tiempo para que cuando se encuentren dos hits cercanos, se les pueda dar prioridad mayor. Una vez que se han encontrado incidencias de palabras cercanas, se empieza a computar una aproximación. Ahora el conteo no se hace solo por cada tipo de hit, sino que también por cada tipo de proximidad. Con esto obtendremos los vectores Tipo-Proximidad-Peso y Conteo-Peso. Y luego se computa el IR haciendo el producto cruz entre ambos vectores.

Al lector que le interese profundizar sobre la matemática y la metodología seguida para desarrollar este algoritmo de priorización, puede consultar [SAL75] y [SIN01].

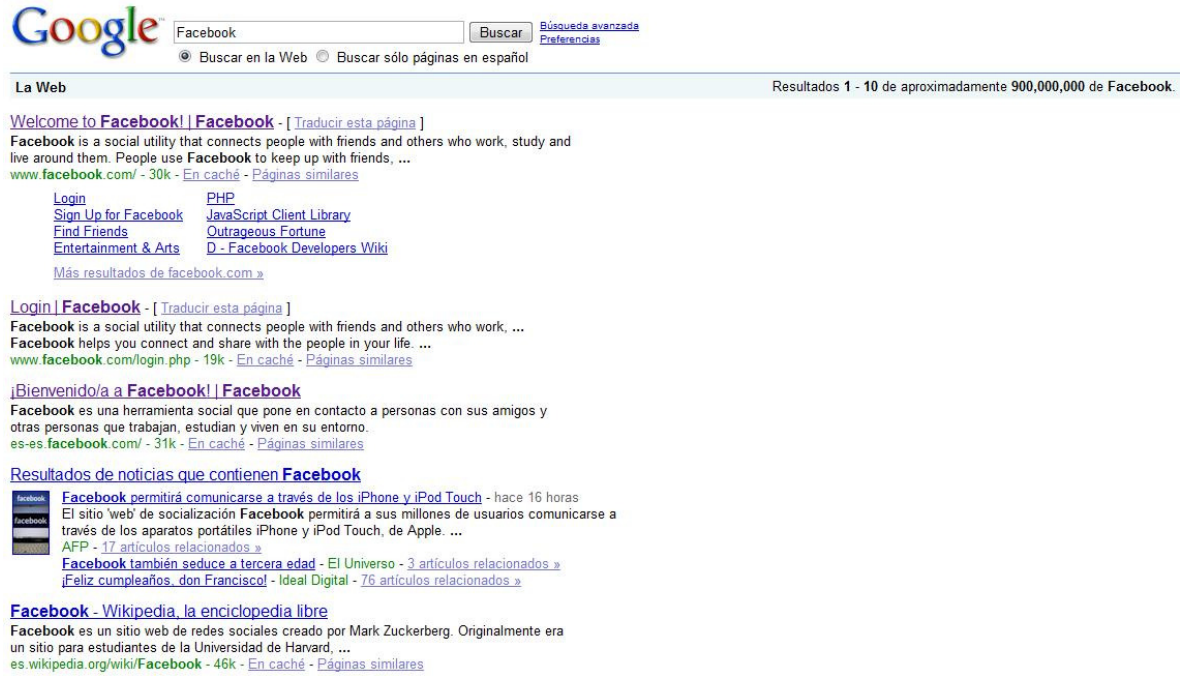
COMPARACIÓN: YAHOO! VS GOOGLE

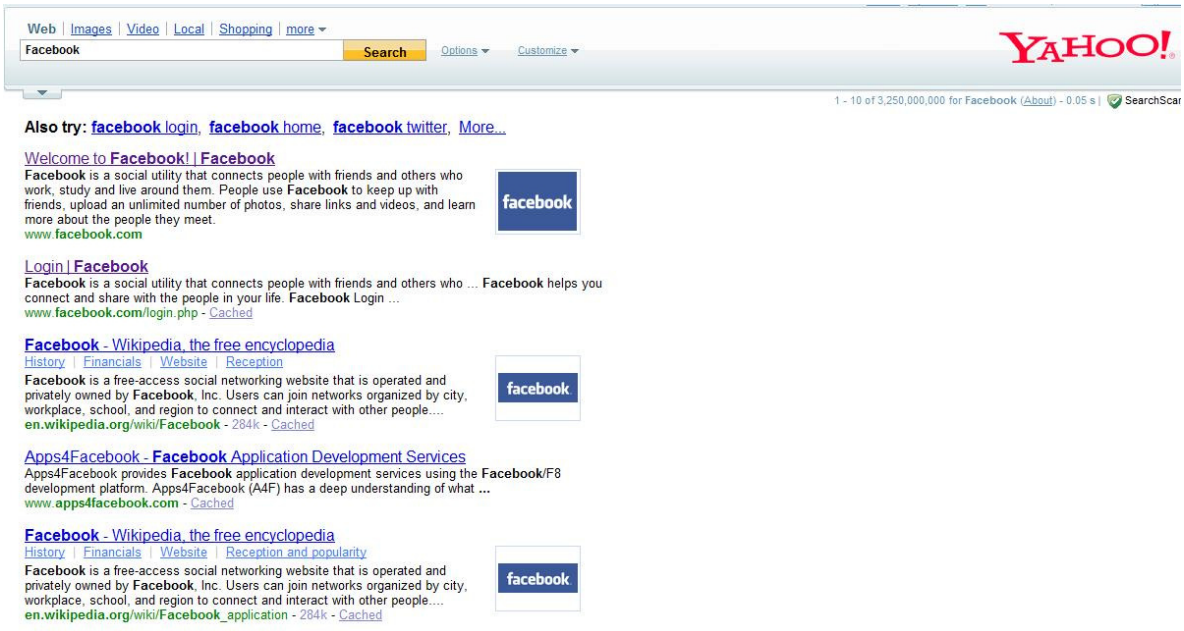
Ahora haremos una rápida comparación entre los resultados de una búsqueda en Yahoo! y en Google. Para hacer valer esta comparación, se hará uso de la interfaz que presenta la página web: http://www.langreiter.com/exec/yahoo-vs-google.html

Veamos qué es lo que obtenemos al buscar la palabra "Facebook" en ambos Motores de Búsqueda:



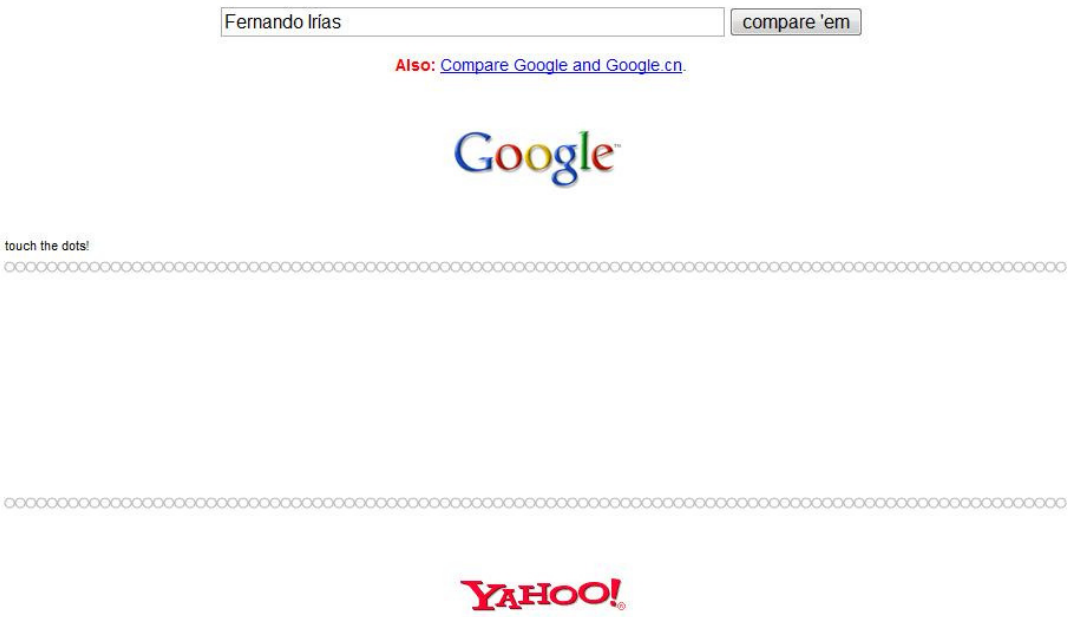
Al parecer ambos buscadores presentan en los primeros lugares las mismas páginas que para ellos eran relevantes. Ahora veamos los resultados concretos:



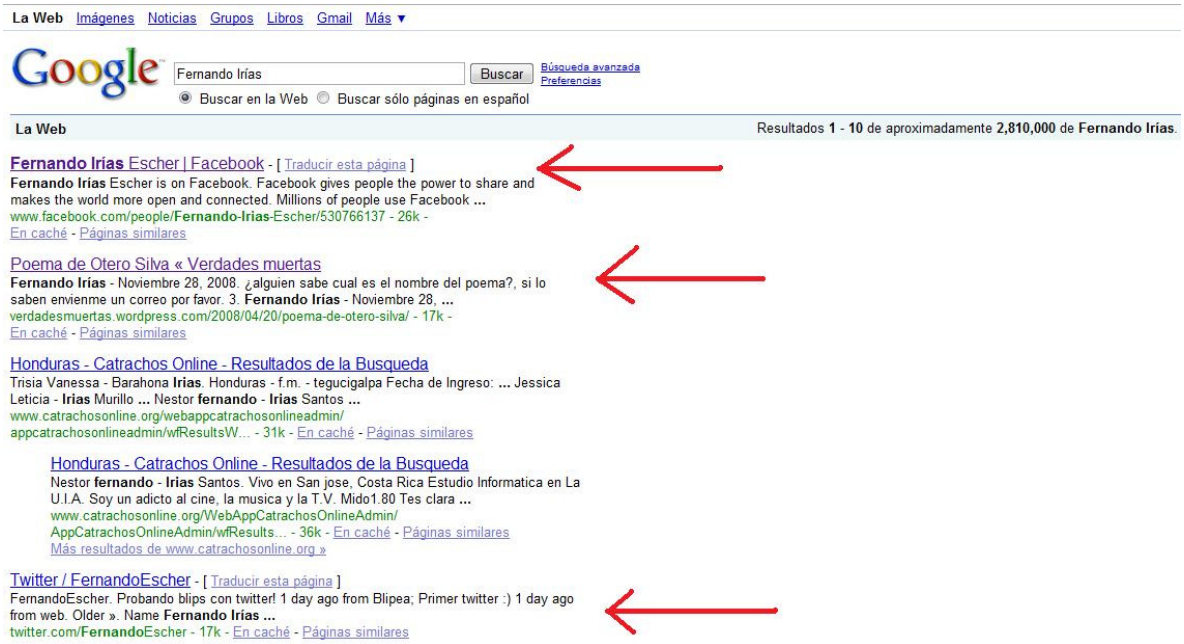


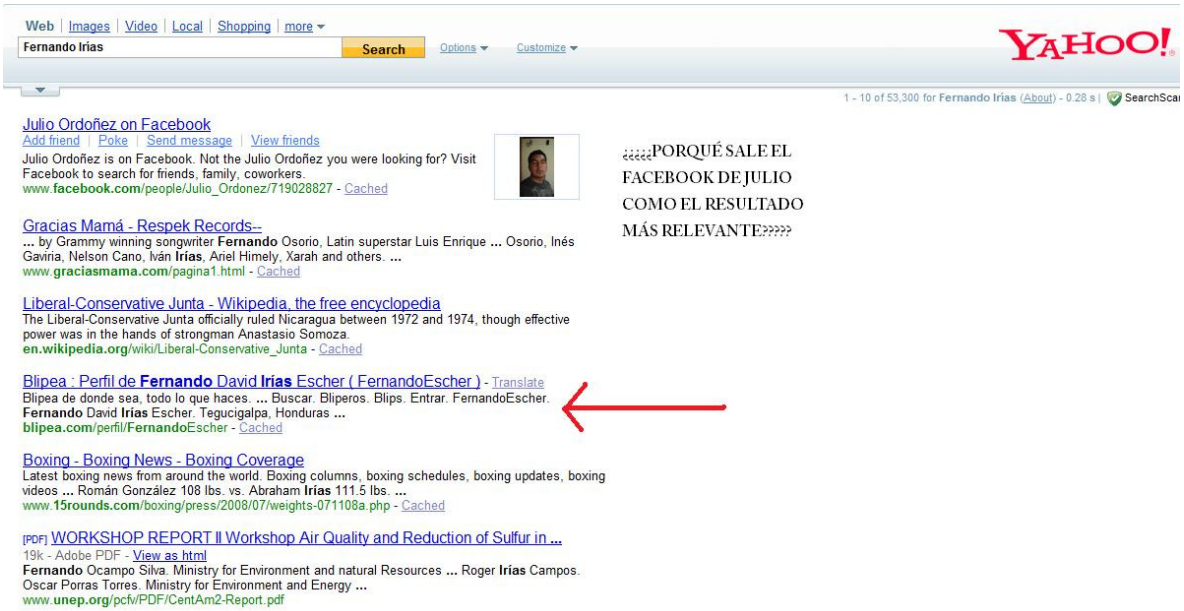
Hasta el momento, ambos se han comportado bien al mostrarme lo que para ellos son las páginas más relevantes. Yo daría como aceptables los resultados de las dos.

Ahora pongamos un poco más interesante la situación. Compliquemos el trabajo de estos Motores de Búsqueda y hagamos una búsqueda de mi nombre, ¿seré una persona difícil de encontrar en internet?



Bueno, está claro que ambos buscadores dieron respuestas totalmente distintas. Veamos los resultados concretos para ver si alguno de ellos tiene la razón:





En ambos resultados, señalé con una flecha roja las páginas que si tienen que ver con mi persona. Lo chistoso es que para Yahoo! yo no soy Fernando Irías, soy Julio Ordoñez, mi compañero de clases. Lo que me lleva a la conclusión de que si uso mucho Yahoo! voy a tener problemas de identidad y cosas así. Así que declaro a ¡GOOGLE! ¡Ganador de este duelo!

¿Qué opinan los ingenieros de Honduras?

Para poder tener una idea de lo que opinan los ingenieros en nuestro país, mandé un correo a unos cuantos ingenieros con una serie de preguntas sobre motores de búsqueda. Curiosamente solo dos personas respondieron. Aún así, esto me permite hacer una comparación un tanto interesante, ya que uno de los que me respondió es el ingeniero Edgares Futch, una persona muy experimentada en el área de la informática. Por el otro lado, tengo una respuesta de mi amigo que está por graduarse Daniel Martínez, una persona que empieza a descubrirse a sí mismo en el ámbito laboral.

Edgares Futch Responde

Fernando Irías: ¿Cuál es su nombre y su ocupación?

Edgares Futch: *Egdares Futch H., ingeniero de sistemas*

FI: ¿Podría dar un primer pensamiento general sobre Motores de Búsqueda?

EF: *Los motores de búsqueda son el punto de partida para obtener contenido del Internet, como tal cumplen una función vital debido al crecimiento continuo y explosivo del Internet. ¿Cuántos nuevos sites habrán al día? Sin motores de búsqueda sería imposible acceder a ese contenido.*

FI: ¿Recuerda como era utilizar Motores de Búsqueda antes de que apareciera Google? En caso de ser así, ¿Nota alguna diferencia entre los Motores de Búsqueda Pre-Google y Post-Google?

EF: *Anteriormente, se usaban índices como Yahoo!, en la cual personas clasificaban y mantenían la lista de sitios del índice. Eran buenos recursos, pero tenían el problema que un nuevo sitio tardaba mucho tiempo en ser colocado, por el proceso de revisión manual. Luego comenzaron a salir motores como AltaVista, los cuales ya incorporaban tecnología de crawlers. La diferencia que se nota, es que ahora todos los SE son basados en crawlers, ya que no se puede mantener un esquema de indización manual.*

FI: En su vida profesional ¿Qué papel han jugado los Motores de Búsqueda? ¿Han sido importantes para que usted se desempeñe en su trabajo?

EF: *Son muy importantes ya que permiten encontrar la información necesaria. Encuentro que es importante aplicar el criterio profesional con la información que se ubica, ya que no todo lo que aparece en Internet es cierto o está correcto.*

FI: ¿Con qué aspectos de los Motores de Búsqueda se encuentra satisfecho?

EF: *La velocidad y calidad de los resultados obtenidos.*

FI: ¿Con qué aspectos de los Motores de Búsqueda NO se encuentra satisfecho?

EF: *Qué es posible alterar o jugar con los resultados obtenidos, por ejemplo el caso del famoso Google Bomb "Miserable Failure".*

FI: ¿Cuál cree usted que ha sido el impacto de estos Motores de Búsqueda en su área de trabajo/estudio?

EF: *Es muy grande, porque permite acceder a una gran cantidad de información relevante.*

FI: ¿Cree usted que estos Motores de Búsqueda son de vital importancia para nuestra sociedad? En caso de creerlo, ¿Cuál es el factor que usted considera de mayor importancia?

EF: *Bueno, depende de qué tanto esté "conectada" la sociedad y cómo la sociedad aprovecha la existencia de los motores de búsqueda. Me he dado cuenta que muchas empresas tienen página web, pero no ponen ni el teléfono ni la dirección, entonces el motor de búsqueda no me sirve para uso local por ejemplo, pero puedo encontrar la forma de llamar a una librería en Miami por ejemplo. El factor más importante para la sociedad considero entonces que sería la disponibilidad de información relevante, pero con contenido local.*

FI: ¿Qué piensa usted que sería el internet si no existieran estos Motores de Búsqueda?

EF: *Sitios más puntuales, tal vez con el esquema de portales reconocidos, y muy específicos (verticales), tal vez perfeccionándose los agregadores de contenido estilo Digg.*

FI: ¿Cuál es el Motor de Búsqueda que más utiliza?

EF: *Google*

FI: Estimando subjetivamente el uso que usted le da a estos Motores de Búsqueda, ¿En qué porcentaje diría usted que encuentra la información que estaba buscando?

EF: *En general siempre encuentro lo que busco, pero creo también que para usar efectivamente un motor de búsqueda, la persona debe tener un buen vocabulario para manejo de sinónimos y un manejo de lenguaje inglés porque la mayoría del contenido está en este lenguaje. Tal vez cuando tengamos la Web Semántica esto será más fácil.*

FI: ¿Tiene algún comentario extra que quiera compartir sobre el tema?

EF: *Es importante ver que por ejemplo Google tiene mecanismos avanzados de acceso, que es importante conocer para lograr más efectividad en las búsquedas. Además hay mucho contenido que no es libre, y que aunque el SE lo encuentre, probablemente la persona que busca no tenga acceso al mismo, por estar en un portal pagado.*

DANIEL MARTÍNEZ RESPONDE

Fernando Irías: ¿Cuál es su nombre y su ocupación?

Daniel Martínez: *Daniel Martínez, Analista Programador de Data Warehouse.*

FI: ¿Podría dar un primer pensamiento general sobre Motores de Búsqueda?

DM: *Una Aplicación de sistema de búsqueda por palabras clave.*

FI: ¿Recuerda como era utilizar Motores de Búsqueda antes de que apareciera Google? En caso de ser así, ¿Nota alguna diferencia entre los Motores de Búsqueda Pre-Google y Post-Google?

DM: *Sí, recuerdo utilizar AltaVista o Yahoo. Con respecto a la diferencia es bastante notable, antes de Google la búsqueda era más complicada en el sentido de los resultados presentados, con Google se encuentran las cosas más rápido.*

FI: En su vida profesional ¿Qué papel han jugado los Motores de Búsqueda? ¿Han sido importantes para que usted se desempeñe en su trabajo?

DM: *Claro que sí, cuando tengo alguna duda o quiero resolver un problema, acudo a Google y encuentro la información que necesito de manera rápida.*

FI: ¿Con qué aspectos de los Motores de Búsqueda se encuentra satisfecho?

DM: *Velocidad, variedad de opciones de búsqueda particulares (imágenes, noticias, grupos,etc), "exactitud" en los resultados.*

FI: ¿Con qué aspectos de los Motores de Búsqueda NO se encuentra satisfecho?

DM: *Ninguno, estoy conforme con su rendimiento.*

FI: ¿Cuál cree usted que ha sido el impacto de estos Motores de Búsqueda en su área de trabajo/estudio?

DM: *Efectividad. Una buena herramienta para el estudio.*

FI: ¿Cree usted que estos Motores de Búsqueda son de vital importancia para nuestra sociedad? En caso de creerlo, ¿Cuál es el factor que usted considera de mayor importancia?

DM: *Son importantes, pero no los consideraría vitales.*

FI: ¿Qué piensa usted que sería el internet si no existieran estos Motores de Búsqueda?

DM: *Un lugar muy difícil de comprender, navegar y entender.*

FI: ¿Cuál es el Motor de Búsqueda que más utiliza?

DM: Google

FI: Estimando subjetivamente el uso que usted le da a estos Motores de Búsqueda, ¿En qué porcentaje diría usted que encuentra la información que estaba buscando?

DM: *87.765%*

FI: ¿Tiene algún comentario extra que quiera compartir sobre el tema?

DM: *Ninguno*

CONCLUSIONES

- A medida que evoluciona y crece el internet, los Motores de Búsqueda se hacen más rápidos y precisos. No es de extrañar que ahora veamos Motores de Búsqueda Verticales que permiten buscar cierto tipo de información específica en internet, es más, quizás el próximo paso sea que los mismos MB sepan de que se trata lo que ando buscando.

- Es de vital importancia para el profesionalismo del ingeniero en sistemas, el conocer las estructuras básicas de programación y de análisis de algoritmos. Estas son las cosas que se pueden lograr cuando se tiene la dedicación en aprender a utilizar las mejores prácticas para hacer el mejor software.

- Google ha sido un hito en la historia del internet. Muy pocas empresas han tenido el auge con el que hoy cuenta esta empresa. Sin duda ha sido una de las pioneras en varias ramas de la ciencia y la tecnología. No podemos dejar de decir que nuestras vidas como ingenieros se han facilitado un poco más gracias a Google.

- A pesar de que el cálculo del PageRank para poder priorizar páginas al momento de hacer una búsqueda en muy ingenioso y preciso, siempre existe la probabilidad de que alguien encuentre la manera de explotar debilidades que no estén cubiertas en cierto momento. Ejemplo de esto es el caso del Google Bomb.

- Entre la comparación entre Google y Yahoo!, pudimos ver que aunque Yahoo! presenta resultados muy relevantes, al momento de obligarlos a ir más allá y buscar información que tal vez nunca nadie había buscado antes en la historia de esos buscadores, se presenta una notable madurez en el arquitectura de Google para manejar nuevos queries.

- Las encuestas reflejan claramente el nivel de madures de un ingeniero con experiencia en el rubro, y uno que acaba de abrirse camino a través de este. El ingeniero Edgares Futch fue un poco más profundo en el análisis de sus preguntas, mientras que Daniel Martínez fue un tanto más puntual.

- Otra observación interesante es que el ingeniero Edgares Futch siempre encuentra lo que busca en los Motores de Búsqueda, mientras que Daniel Martínez solo un 88% de las veces lo logra.

BIBLIOGRAFÍA

- [BRI98] Brin, S., y Page, L., *The Anatomy of a Large-Scale Hypertextual Web Search Engine*, Stanford University, 1998

- [SAL75] Salton, G., Wong, A., y Yang, C. S., *A Vector Space Model for Automatic Indexing*, Cornell University, 1975

- [SIN01] Singhal, A., *"Modern Information Retrieval: A Brief Overview"*, Bulletin of the IEEE, 2001

- [WIK09] *"Definición de Googlear",* http://es.wiktionary.org/wiki/googlear

- [MCB94] McBryan, O. A., *GENVL and WWW: Tools for Taming the Web*, CERN, 1994

- [COO97] Coombs, T., y DeLeon, R., *Google Power Tools Bible*, John Wiley and Sons, 2007

- [DEA08] Dean, J., y Ghemawat, S., *MapReduce: Simplified Data Processing on Large Clusters*, Communications of the ACM, 2008

- [WKP09] *"Google Bomb",* http://es.wikipedia.org/wiki/Google_bomb