TESIS

"Análisis de criterios de relevancia en motores de búsqueda de la web"

DIRECTORES

Dr. Daniel Yankelevich
Departamento de Computación
FCEN – UBA
dany@dc.uba.ar

Dr. Luis Gravano
Department of Computer Science
Columbia University
gravano@cs.columbia.edu

TESISTA

A. U. Eduardo F. Chao
L.U. N° 426/92
Departamento de Computación
FCEN – UBA
efchao@dc.uba.ar

2002

Capítulo 1. Introducción y motivación

En este capítulo se hace hincapié en las motivaciones para el estudio de las nociones de relevancia que se aplican en los motores de búsqueda de la World Wide Web (WWW).

También se presenta la estructura de la tesis en cuanto al resto de los capítulos y cómo éstos se relacionan entre sí.

Es evidente que la Web se ha transformado en un reservorio de información enorme, cuya utilidad sólo estamos comenzando a comprender y aprovechar. La cantidad de documentos visibles por los Motores de Búsqueda (MB) ronda los 4000 millones, que significan 40 TB de texto. Históricamente se ha venido observando una duplicación de tamaño en períodos de 4 a 9 meses. La web invisible o no indexable se calcula que es de una magnitud entre 400 y 550 veces superior. En lo que respecta a los MB, la Web debe interpretarse como la indexable o de "superficie"; y no como la "escondida", dinámica o profunda.

Si bien es cierto que todo documento es accesible directamente mediante la navegación, especificando una dirección URL, este procedimiento no es efectivo para realizar una búsqueda de información en una gran colección de documentos como la Web.

El acceso concreto y rápido que una persona tiene a tal volumen de datos debe hacerse a través de interfaces de portales que recolectan parte de la Web, y proveen el servicio de búsqueda al usuario. Estos sistemas pueden realizar la recolección mediante el criterio de operadores humanos o de forma automática.

De acuerdo a ello, se los denomina **directorios** o **motores de búsqueda**, respectivamente.

En el primer caso, proveen al usuario una estructura jerárquica por temas que es navegable. Los documentos que ofrecen son de buena calidad y, aunque la profundidad y amplitud del árbol de temas sea importante, no alcanzan a abarcar todos los temas contemplados en la web. De hecho, la mejor cobertura es menor al 0,08% de todos los documentos de la web (menos de 3 millones). El ejemplo más conocido de este tipo de portales es Yahoo!.

Además de la jerarquía temática, brindan al usuario la posibilidad de ingresar un requerimiento de información o consulta. Como respuesta a la misma, pueden devolver algunos nodos del árbol de temas si corresponde con las páginas relevantes. En caso de no encontrar en su propia base de datos ningún recurso relacionado, transfieren la consulta a un servicio de búsqueda que se contrata a tal fin. Este MB consulta, a su vez, la propia base de datos (diferente a la del directorio) y devuelve los documentos más relevantes.

Yahoo! utiliza actualmente a Google para este fin, y anteriormente usó a Inktomi.

Otro tipo de **sistema de recuperación de información** (SRI) que puede utilizar un usuario para realizar una búsqueda de información en la web, es el de los MBs. Como se mencionó, la base de documentos sobre la que actúa un MB se recolecta automáticamente. Ningún MB logra indexar una magnitud superior al 20% de la web, aunque con Google se pueda acceder a cerca del 33% de la web por particularidades de su algoritmo de indexación. Además, la intersección entre colecciones de los MBs es llamativamente baja: el 36% de páginas muestreadas en consultas de 14 MBs fueron indexadas por un solo MB, y el 76% por menos de 3 MBs; sólo el 3% por 9 a 12 MBs.

En este caso, también se ofrece al usuario un formulario donde ingresar su consulta. Ante la misma, se suele devolver un *ranking*, es decir una lista de páginas ordenadas por relevancia no creciente. También es posible, en algunos casos como Google, que se retornen nodos de la jerarquía temática que mantienen o de otros directorios.

Vemos entonces que los directorios se apoyan en MBs, y éstos a su vez mantienen su propio árbol de temas o recurren a un servicio equivalente de un tercero. Esta complementación se puede explicar por la suposición habitual de que las jerarquías de temas son usadas para buscar recursos de calidad en temas suficientemente populares; y los MBs sirven para obtener resultados en asuntos no tan reconocidos, o cuando se necesita cubrir más extensamente a la web.

Pero si lo que se requiere es el acceso al mayor número posible de recursos en la web, se debe recurrir a la combinación de varios de estos buscadores. El escaso solapamiento entre sus bases de datos torna interesante la posibilidad de buscar simultáneamente en ellos, y obtener una lista de documentos resultante de la integración de los *rankings* individuales.

Este servicio de multibúsqueda lo brindan los metabuscadores. Estos sitios ofrecen una interfaz similar a la de un MB, pero al recibir una consulta lo que realmente hacen es efectuarla en varios MBs y luego integrar los resultados en un ranking único. Los MBs no cooperan con estos esfuerzos de integración.

Cada consulta individual que el metabuscador realiza en cada MB utilizado ocurre automáticamente a través de la misma interfaz pública que se brinda a los usuarios. De hecho, un MB no puede diferenciar si la consulta la realiza un agente de software o un usuario humano, a menos que el programa cliente se identifique explícitamente.

El problema de la integración de los resultados de varios MBs es muy complejo, ya que se deben mezclar nociones de relevancia diversas y, más importante aún, desconocidas. Estas nociones se implementan basándose a su vez en un modelo de recuperación de información (RI), que representa a documentos y consultas con diversas estructuras, y calcula similitudes entre ellos también según diferentes variantes de asignación de importancia a los términos representativos.

A pesar de que no hay demasiada información pública acerca de los algoritmos o modelos utilizados por los actuales MB de la web, se puede llegar a afirmar [BAE/99] que muchos de ellos usan variantes del modelo Booleano o vectorial. Este último modelo de RI, asegura simplicidad de implementación; y excelente performance en tiempo y en calidad de respuesta con colecciones grandes y de temas generales.

Los algoritmos de asignación de relevancia son secretos comerciales resguardados por su fundamental importancia competitiva. Es comprensible desde el punto de vista de los MBs, ya que la publicación de dichos algoritmos o técnicas derrumbaría el negocio sobre el que se asientan. Sin embargo, este hecho impide que se pueda aprovechar del todo la sumatoria de las coberturas que varios MBs pueden brindar.

Resulta complejo evaluar con certeza el grado de calidad o precisión de cada MB. Si bien hay estudios de satisfacción del usuario promedio de un buscador, no se puede evaluar la performance del MB con colecciones específicas sobre determinados temas. Este método es el que se aplica normalmente a un SRI para comparar su desempeño, por ejemplo en las Conferencias sobre Recuperación de Texto (*Text Retrieval Conferences – TREC*).

Otro punto clave del que se carece de datos válidos, es saber qué parte de la web tiene indexada un MB, lo que depende directamente de la manera en que recolecta las páginas de la web.

Se conocen técnicas de recolección o *crawling* de páginas, sobre las que seguramente se basan las de los MBs. Pero es claro que las variaciones entre los diferentes algoritmos adoptados provoca que los MBs recorran partes diferentes de la web, y de esta manera recolecten páginas de diferentes sitios. No obstante, se observa, cada vez más, una propensión a indexar páginas populares, o sea aquéllas con mayor cantidad de enlaces entrantes. Otra

tendencia es a utilizar el algoritmo de relevancia para decidir qué páginas recorrer; aumentando la probabilidad de brindar una mayor calidad de respuesta ante las consultas. Esto quizás redunde en un futuro hacia una mayor superposición de las colecciones de los MBs.

Vemos entonces que son múltiples las dificultades para conocer con certeza los detalles de implementación de los algoritmos de asignación de relevancia, de recolección de páginas; y por lo tanto, de la colección de documentos, y de la performance en calidad de las respuestas brindadas.

Nos concentraremos en esta tesis en el punto concerniente a la relevancia, que es el asunto central en el diseño de cualquier SRI, sea de la web o no.

Los MBs más importantes son sitios extremadamente populares, con decenas de millones de consultas por día. Google dice recibir 100 millones; AltaVista, por su parte, 50 millones. Los sitios web reciben cada vez más tráfico proveniente de los buscadores, y los usuarios ya incluyen a la búsqueda como una de sus actividades predilectas después de la lectura del email. Por otra parte, los estudios indican que la búsqueda es la principal forma de conocer sitios web, seguido por los enlaces entre páginas.

Con semejante audiencia, es natural que despierten el interés de las empresas por ubicar a sus sitios en posiciones relevantes ante consultas realizadas por un volumen suficiente de dichas consultas. Esta presión del mercado, esta necesidad de obtener una exposición importante y competitiva ante una potencial multitud de clientes, genera un mayor interés por conocer los criterios de relevancia de los MBs. Pero el objetivo en este caso, no es el de integrar múltiples nociones de relevancia o el de evaluar la calidad de los MBs, sino el de manipular el servicio de búsqueda en provecho propio.

Este proceso de manipulación, se ha extendido tanto que merece ser analizado y detallado en sus múltiples técnicas. Se lo suele denominar *web spamming* o *search engine persuasion* (*SEP*) –persuasión de motores de búsqueda (PMB)– en la bibliografía. También se lo conoce en el ámbito comercial de empresas que brindan el servicio de posicionamiento web como optimización de MB.

Consiste en realizar un relevamiento de las posibles consultas que los potenciales grupos de clientes pueden llegar a realizar, diseñar las páginas destinadas a cada grupo de clientes y consultas, y enviar dichas páginas para su agregado a la base de datos de los MBs. Este envío o *submission* de páginas es un servicio que ofrecen los MBs a los autores de las páginas. A éstos les brinda una opción alternativa a quedarse esperando que los *crawlers* o *robots* del MB lleguen a sus sitios, algo que como vimos puede no suceder en un corto plazo. Lamentablemente, la experiencia indica que el procesamiento de las *submissions* puede durar meses.

Es importante destacar que según dicen los mismos MBs, las páginas enviadas manualmente por los *webmasters* mediante los formularios destinados a tal efecto, pasan por el mismo tamiz que las páginas obtenidas por los programas recolectores o *crawlers*. Es decir, no reciben un trato o una inspección especial, más allá de la que puede recibir cualquier página obtenida por los MBs por otro medio, con el fin de indexarla.

Pero ¿qué tamiz o filtro puede implementar un MB, además del consabido proceso de indexación de documentos o páginas? Es claramente previsible que los MBs estén al tanto de la intención de manipulación de sus servicios, pues ello atenta contra la calidad que intentan brindar. Es por eso que ante la PMB o *SEP*, los MBs responden con medidas de defensa y filtrado de páginas.

Estas técnicas de seguridad defensivas, no sólo se verifican en el momento en que se decide si agregar o no una página a la base de datos, sino también en el momento de calcular la

función de *ranking* o relevancia. Como veremos, se implementan formas de amortiguar el efecto PMB, para brindar resultados acordes con las consultas, y también para evitar brindar publicidad indebida a empresas que, de otro modo, tendrían que pagar por obtenerla.

Sintetizando lo expuesto hasta aquí:

- Los MBs son los sistemas que mayor cobertura de la web obtienen, no superando cada uno el 20% del total de la web indexable.
- Son también los sistemas mejor preparados para afrontar el crecimiento de la web.
- Buscan eficientemente en ese gran volumen de datos. Reciben decenas de millones de consultas por día.
- Las colecciones de los MBs se solapan poco, 76% de la web por menos de 3 MBs.
- Por ello, es interesante integrar las colecciones, para obtener mayor cobertura de la web. Las colecciones no son conocidas, como tampoco la forma de recolectarlas.
- Lo más unificable entonces son los *rankings* de varios MBs, que resultan de las diversas nociones de relevancia, que a su vez dependen del modelo de RI implementado.
- Los metabuscadores intentan realizar esta integración, aunque el ideal es conocer más en detalle los algoritmos de asignación de relevancia, para mezclar convenientemente los *rankings* devueltos en sus interfaces públicas.
- La importancia de los MBs provoca el mecanismo de *web spamming*, que influye en la calidad de los MBs, y los obliga a tomar medidas anti-spam que pueden adoptar en sus algoritmos de aceptacion de *submissions*, de crawling, de indexacion y de relevancia.

Estas consideraciones muestran la importancia de intentar comprender o descubrir qué nociones o criterios se consideran para asignar relevancia a documentos web respecto de consultas.

La estructura de la presente tesis responde a la discusión precedente de la siguiente manera:

- Capítulo 2: se introducen los sistemas de recuperación de información (SRI). Siendo los MBs, ejemplos de este tipo de sistemas, es atinado hacer un panorama de los modelos clásicos y los conceptos que se manejan.
- Capítulo 3: se describe cómo se adaptaron y aplicaron los SRI a la web, con énfasis en la utilización de los hiperenlaces entre documentos para mejorar el cálculo de las relevancias.
- Capítulo 4: se describen las técnicas de ataque y defensa PMB.
- Capítulo 5: referente al análisis experimental de las nociones de relevancia, se reportan y analizan los resultados de algunos experimentos diseñados en el marco de la tesis que responden a dos técnicas diferentes. La primera consiste en la validación de hipótesis acerca de la ubicación de páginas en un *ranking;* y la restante, en un sistema de evaluación o aproximación a *rankings* de un SRI, en particular de AltaVista y AlltheWeb (FAST). Este último desarrollo es el aporte más importante de la presente tesis, ya que presenta una manera sistemática de comparar un *ranking* de páginas de un SRI de la web, contra múltiples *rankings* locales obtenidos a partir de diversas variantes de asignación de relevancia.

- Capítulo 6: se arriba a algunas conclusiones y se señala los posibles trabajos futuros en la línea de investigación de esta tesis.
- Finalmente, se presenta un glosario con explicación y traducción al inglés de los términos utilizados; y se detalla la bibiliografía y otras fuentes de información utilizadas.

Capítulo 2. Sistemas de recuperación de información

En este capítulo se brindará una introducción al campo de la Recuperación de la Información (RI), haciendo hincapié en los modelos clásicos de RI y los conceptos allí manejados.

Sobre todo, el interés recaerá en el modelo de espacio vectorial y en las posibles maneras de calcular la relevancia de documentos. Se intentará graficar y ejemplificar cuando se desee fijar alguna idea de importancia, o que requiera de un correlato más práctico.

La estructura del capítulo se apoya fuertemente en el enfoque de [BAE/99], aunque se amplía y profundiza con otras fuentes cuando resulta conveniente brindar otra visión, alguna variante o un aporte útil.

2.1. Presentación

2.1.1. Información versus datos

La disciplina de Recuperación de la Información (*Information Retrieval –IR)* abarca la representación, almacenamiento, organización y acceso a elementos de información.

Estos procedimientos deben asegurar al usuario un fácil acceso a los items de su interés.

Lamentablemente, la caracterización del **requerimiento de información del usuario** no es un problema simple. Considérese una consulta del estilo en el contexto de la WWW:

> "Encontrar los documentos que contengan información acerca de instituciones educativas que sean objeto de partidas presupuestarias del Estado Argentino y participan de algún tipo de financiamiento privado".

Si ingresamos esta descripción completa de la necesidad del usuario, en las interfaces actuales de los MBs de la web, el sistema receptor debe traducir el pedido a una consulta (*query*) que sea procesable por el buscador o sistema de RI.

Comúnmente, la traducción deriva en un conjunto de términos clave (*keywords*) o términos índice (*index terms*) que resumen la descripción de lo que se quiere obtener.

Dada una consulta del usuario, el objetivo central de un SRI es recuperar información que pueda ser útil o relevante para el usuario. Se pone énfasis sobre la recuperación de la **información** en oposición a la recuperación de **datos**.

Recuperar datos en el contexto de un SRI consiste básicamente en determinar qué documentos de una colección contienen las palabras de la consulta. Esto, en la mayoría de los casos, no alcanza para satisfacer el requerimiento del usuario, quien se encuentra interesado más en obtener información sobre un tema que datos que correspondan a la consulta.

Un sistema de recuperación de datos (SRD) apunta a encontrar objetos que satisfacen condiciones claramente definidas, como las de una expresión regular o del álgebra relacional. Por ello, un objeto erróneo entre miles significa una falla total. En un SRI, los objetos recuperados pueden ser imprecisos, y pequeños errores suelen pasar desapercibidos.

La principal razón para estas diferencias es que los SRI se manejan con texto del lenguaje natural, que no siempre está bien estructurado y puede ser ambiguo semánticamente. En el otro extremo, los SRD manejan datos con estructura y semántica bien definidas.

Para ser efectivo en su intento de satisfacer la necesidad de información del usuario, el SRI debe "interpretar" de alguna manera el contenido de los elementos de información (documentos) en una colección y ordenarlos (*rank*) de acuerdo al grado de relevancia a la consulta del usuario. Esta interpretación del contenido de un documento involucra la extracción de la información sintáctica y semántica del texto del documento, y el uso de dicha información para satisfacer o corresponder a la consulta realizada.

La dificultad no sólo reside en saber cómo extraer esta información, sino también en saber cómo usarla para decidir la relevancia. Por lo tanto, la noción de **relevancia** está en el centro del proceso de RI. De hecho, el objetivo principal de un SRI es recuperar todos los documentos que son relevantes a una consulta de usuario, procurando recuperar la menor cantidad posible de documentos no relevantes.

2.1.2. La tarea del usuario y la visión lógica de los documentos

La recuperación efectiva de información relevante está directamente afectada tanto por la **tarea del usuario** como por la **visión lógica de los documentos** adoptada por el SRI.

Tanto al especificar un conjunto de palabras que cargan la semántica del requerimiento de información ante un SRI, como al usar una expresión para definir las restricciones que deben satisfacer objetos del conjunto respuesta, podemos afirmar que el usuario está buscando información útil mediante una tarea de **recuperación**.

Si el interés de un usuario está pobremente definido o es intrínsecamente amplio, esto lo puede llevar a cambiar el alcance o el foco de su búsqueda a medida que interactúa con la interfaz del sistema. En [BAE/99] se afirma que, al hacer esto, el usuario está recorriendo o navegando (*browsing*), y no buscando (*searching*). En este caso, los objetivos del proceso de recuperación no están claros desde el principio y el propósito del mismo puede cambiar.

Sin embargo, muchas veces el proceso de búsqueda puede involucrar refinamientos de las consultas iniciales, que hacen que el límite entre estas dos tareas del usuario se vuelva difuso.

Los SRI modernos, como las interfaces web, proveen formas de recuperación de información y datos, pero limitados recursos para navegar por la colección. Esta combinación puede darse en el futuro.

Por razones históricas, los documentos de una colección se encuentran representados a menudo por conjuntos de términos índice o clave. Tales términos pueden extraerse directamente de los documentos o ser especificados por un humano, como suele suceder en el ámbito de las ciencias de la información.

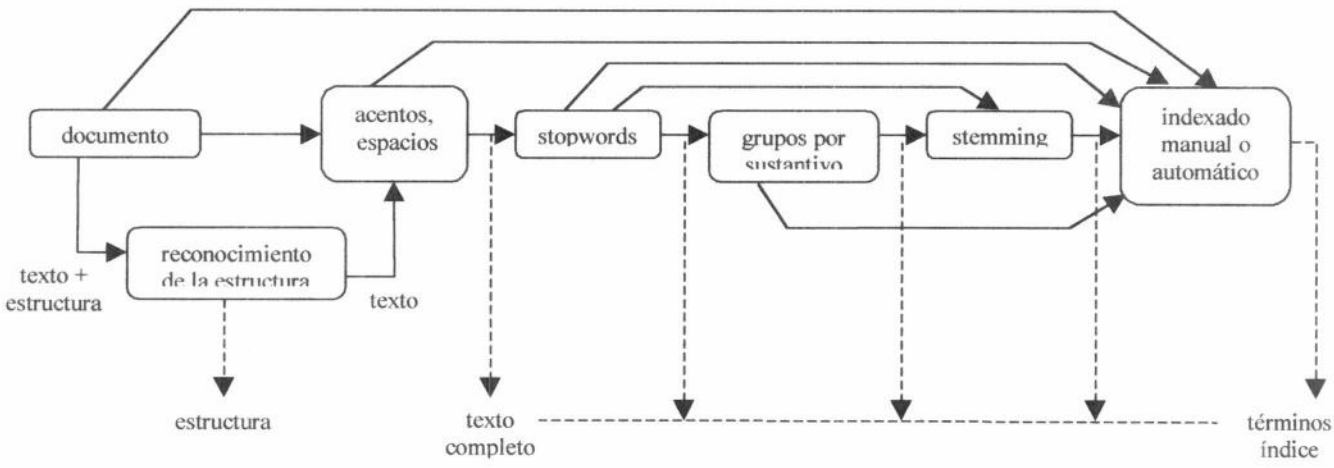
Sin perjuicio de la manera en que se obtienen estos *keywords*, proveen una visión lógica de los documentos.

Los avances en materia de tecnología de computadoras están haciendo posible la representación de un documento mediante el conjunto de todas sus palabras. En ese caso, se dice que el SRI adopta la visión (o representación) lógica de **texto completo**. Con colecciones muy grandes, sin embargo, inclusive las modernas computadoras pueden tener que reducir el conjunto de términos representativos. Esto puede llevarse a cabo mediante la eliminación de palabras muy frecuentes (*stopwords*) como conectivos o artículos, la reducción de varias palabras a su raíz gramatical común (*stemming*), y la identificación de grupos por sustantivo (*noun groups*), que elimina adverbios, adjetivos y verbos. Además se puede aplicar la compresión.

A estas operaciones se las llama transformaciones u **operaciones textuales**. Reducen la complejidad de la representación de documentos y permiten pasar de una visión lógica de texto completo a otra de conjunto de términos índice.

El texto completo es claramente la visión lógica más completa de un documento; pero su uso implica generalmente costos computacionales más altos. Un conjunto pequeño de categorías, generadas por un especialista humano, proveen la visión lógica más concisa de un documento; pero su uso puede redundar en una recuperación de calidad pobre.

Un SRI puede adoptar diversas visiones lógicas intermedias de un documento:



Además de adoptar alguna de las representaciones intermedias, el SRI puede reconocer la estructura interna presente normalmente en un documento (e.g. capítulos, secciones, subsecciones). Esta información sobre la estructura del documento puede ser muy útil y es necesaria para los modelos de recuperación de texto estructurado.

Como se observa en la figura, la representación lógica de un documento es un continuo que puede variar gradualmente desde una representación de texto completo hacia una representación de un nivel superior, especificada por un humano.

2.1.3. Panorama histórico de los SRI

La humanidad siempre estuvo interesado en organizar la información para su posterior obtención y uso. Un ejemplo típico es la tabla de contenidos de un libro.

Al crecer la cantidad de libros, se tornó necesario construir estructuras de datos para asegurar el acceso rápido a la información almacenada. Una estructura añeja y popular para acelerar la recuperación es la colección de palabras o conceptos con los cuales se asocian apuntadores a la información o los documentos relacionados: el **índice**.

Los índices están en el núcleo de todos los SRI modernos. Brindan un acceso más rápido a los datos y aceleran el procesamiento de las consultas. Por siglos se utilizaron índices creados manualmente como jerarquías de categorización. Esto es práctica habitual en muchas bibliotecas actuales.

La llegada de las computadoras ha hecho posible la construcción automática de índices enormes.

La Web continúa usando índices que son muy similares a los usados por los bibliotecarios de hace un siglo. Pero se produjeron cambios dramáticos:

- es más barato acceder a múltiples fuentes de información;
- los avances en las comunicaciones digitales permitieron acceder a redes remotas;
- la libertad de publicar lo que cualquier persona considere útil: por primera vez en la historia, mucha gente tiene libre acceso a un gran medio de publicación.

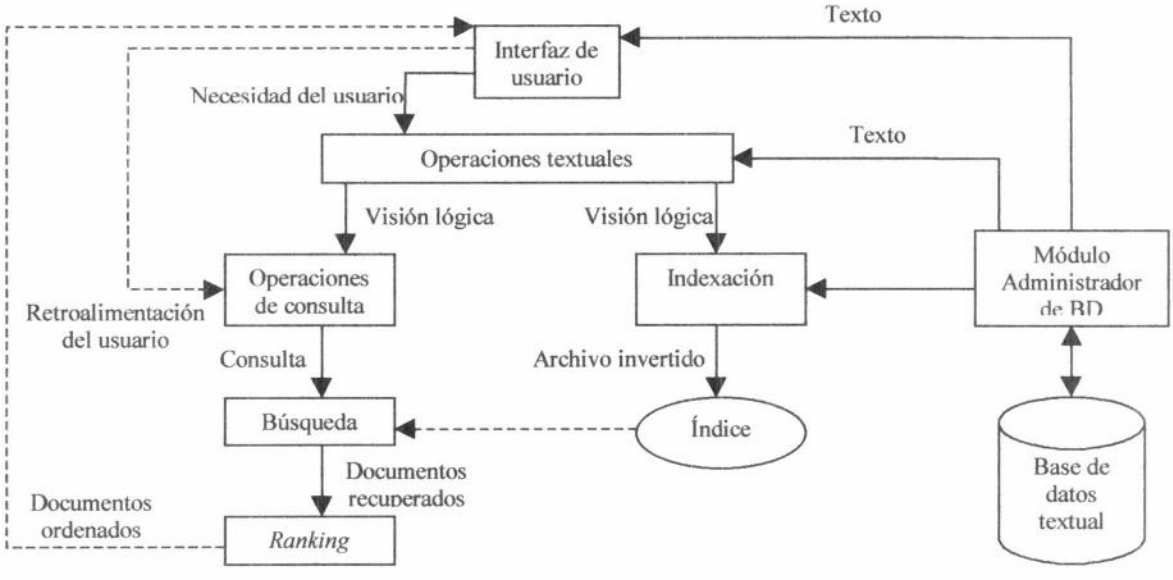
Esta facilidad de acceso y de publicación, permitió que la gente usara la Web como un medio altamente **interactivo**. La posibilidad de intercambiar información instantáneamente y en el momento que se prefiera aumenta la conveniencia del servicio.

Son tres las cuestiones que emergen con gran importancia en el futuro de los SRI:

- Qué técnicas lograrán una recuperación de mejor calidad. Todavía es dificultoso sino imposible obtener información relevante a un requerimiento dado de información.
- La rapidez de respuesta se convierte en un imperativo ante la demanda creciente.
- Cómo incluirán los sistemas de RI al perfil o a los patrones de comportamiento del usuario en sus estrategias de recuperacion.

2.1.4. El proceso de RI

El proceso de recuperación se puede describir por medio de la arquitectura de *software* genérica de la siguiente figura:



Antes de comenzar siquiera el proceso de recuperación, se debe definir la base de datos de texto. Consiste en especificar los documentos que se usarán, las operaciones que se aplicarán sobre el texto y el modelo textual, es decir la estructura del texto y los elementos que se podrán recuperar.

Las operaciones textuales transforman los documentos y generan una visión lógica de los mismos. Con esta representación lógica, el administrador de la Base de Datos (BD) construye un índice del texto, para permitir una rápida búsqueda en un gran volumen de datos. La estructura de índice más usada es la de **archivo invertido**. El esfuerzo y los recursos invertidos en construir el índice serán amortizados consultando el sistema repetidas veces.

Una vez que la colección de documentos ha sido indexada, puede iniciarse el proceso de recuperación. El usuario especifica una **necesidad del usuario**, que es analizada

sintácticamente (*parsing*) y transformada por las mismas operaciones textuales que fueron aplicadas al texto de la colección. Luego, pueden aplicarse **operaciones de consulta**, antes de generar la **consulta** real, que proporciona una representación de la necesidad del usuario entendible por el sistema. La consulta es entonces procesada para obtener los **documentos recuperados**. La estructura de índice construida previamente permite un rápido procesamiento de la consulta.

Antes de enviarlos al usuario, los documentos recuperados son ordenados según un **criterio de relevancia**. El usuario ahora examina el conjunto de documentos ordenados en la búsqueda de información útil. Llegado a este punto, puede reconocer y señalar un subconjunto de los documentos vistos como interesantes, y dar comienzo a un ciclo de **retroalimentación del usuario** (*user feedback*). En tal ciclo, el sistema utiliza los documentos seleccionados por el usuario para cambiar la formulación de la consulta. Es esperable que esta nueva consulta sea una mejor representación del requerimiento real del usuario.

Si consideramos las interfaces de los SRIs actuales (incluyendo motores de búsqueda y directorios de la Web), notaremos que al usuario casi nunca se le pide que declare su requerimiento de información. Por el contrario, se le exige una representación directa de la consulta que el sistema ejecutará. Como muchos usuarios ignoran las operaciones textuales y de consultas, frecuentemente formulan consultas que son inadecuadas. No es sorprendente pues encontrarnos con que las consultas formuladas de una manera pobre resultan en una **calidad de recuperación** pobre, como ocurre a menudo en la Web.

2.2. Modelando los SRI

Los SRI tradicionales utilizan términos índice para indexar y recuperar documentos. En un sentido estricto, un término índice es una *keyword* o un grupo de palabras relacionadas que tienen un significado propio, normalmente la semántica de un sustantivo.

En su forma más general, un término índice es simplemente cualquier palabra que aparece en el texto de un documento de la colección. La recuperación que se basa en los términos índice es sencilla, aunque impone preguntas respecto del proceso de recuperación de información. Por caso, recuperar usando términos índice tiene como supuesto fundamental la idea de que la semántica de los documentos y de la necesidad de información del usuario puede ser expresada naturalmente por medio de conjuntos de términos índice.

Claramente, esto es una simplificación extrema del problema porque mucha de la semántica presente en un documento o en un pedido de usuario se pierde al reemplazar el texto por el conjunto de palabras. Más aún, la correspondencia entre cada documento y la consulta de usuario se lleva a cabo en este espacio sumamente impreciso de términos índice.

Por ello, no es sorprendente que los documentos recuperados en respuesta a una consulta expresada como conjunto de términos sean, a menudo, irrelevantes. Si se tiene en cuenta que la mayoría de los usuarios carecen del entrenamiento para formar apropiadamente las consultas, el problema se agudiza con resultados potencialmente desastrosos.

Un problema central en lo que respecta a los SRI es el de predecir cuáles son documentos relevantes, y cuáles no lo son. Tal decisión depende usualmente del **algoritmo de *ranking*** que intenta establecer un ordenamiento simple de los documentos recuperados. Los documentos que aparecen al tope de este ordenamiento son los más factibles de ser relevantes. Esto nos muestra que los algoritmos de *ranking* están en el núcleo de los SRI.

Un algoritmo de *ranking* opera de acuerdo a premisas básicas en cuanto a la noción de relevancia de documentos. Los distintos supuestos o premisas a este respecto llevan a modelos

de RI diferentes. Es decir, el modelo RI elegido determina las predicciones de lo que se considera relevante o no, y que derivará en la implementación de la noción de relevancia en el sistema.

2.2.1. Modelos clásicos de RI

Los tres modelos clásicos en RI se denominan Booleano, vectorial y probabilístico. En el modelo Booleano, los documentos y las consultas son representados por conjuntos de términos índice. Por ello, podemos decir que es un modelo de **teoría de conjuntos**.

En el modelo vectorial, los documentos y las consultas están representados como vectores en un espacio t-dimensional. Entonces decimos que este es un modelo **algebraico**.

En el modelo probabilístico, el marco en el que se modela a documentos y consultas se basa en la teoría de probabilidad, por lo que podemos hablar de un modelo **probabilístico**.

Se han propuesto modelos alternativos para cada modelo clásico. También hay modelos que hacen referencia no sólo al contenido textual, sino también a la estructura presente en el texto escrito. La profundización en el estudio de los mismos está más allá del objetivo de esta tesis.

Es importante enfatizar que el modelo RI (Booleano, vectorial, probabilístico, etc.), la visión lógica de los documentos (texto completo, conjunto de términos índice, etc.), y la tarea del usuario (recuperación, navegación) son aspectos ortogonales de un SRI. De esto se desprende que un mismo modelo RI puede usarse con diferentes visiones de los documentos para realizar diversas tareas de usuario.

Los modelos clásicos que veremos utilizan la visión de términos índice, en la tarea de recuperación.

Comencemos primero por definir precisamente de manera formal que un modelo de RI es una cuadrupla $[\mathbf{D}, \mathbf{Q}, \mathfrak{I}, \mathbf{R}(q_i, d_j)]$ donde

i. **D** es el conjunto de las visiones lógicas o representaciones de los documentos.

ii. **Q** es el conjunto de visiones lógicas o representaciones de los requerimientos de información del usuario o consultas.

iii. $\mathfrak{I}$ es el marco (*framework*) para modelado de representaciones de documentos, consultas y sus relaciones.

iv. $\mathbf{R}(q_i, d_j)$ es una función de *ranking* que asocia un número real con una consulta $q_i \in \mathbf{Q}$ y una representación de documento $d_j \in \mathbf{D}$. Tal función define un ordenamiento entre los documentos respecto de la consulta q_i. $\mathbf{R}$: $\mathbf{D} \times \mathbf{Q} \rightarrow \Re$ (donde $\Re$ son los números reales).

En la construcción de un modelo, se piensa primero en cómo representar a documentos y consultas. Una vez hecho esto, se concibe el marco de modelado. En el modelo clásico Booleano, por ejemplo, dicho marco se compone de conjuntos de documentos y de las operaciones de conjuntos conocidas. Para el modelo vectorial, el marco está compuesto por el espacio vectorial t-dimensional y las operaciones del álgebra lineal sobre vectores. En el modelo probabilístico, lo componen conjuntos, operaciones probabilísticas y el teorema de Bayes.

Como mencionamos, los modelos clásicos de RI suponen que cada documento es descripto por un conjunto de palabras clave o *keywords* representativas llamadas términos índice. Un **término índice** es una palabra (de un documento) cuya semántica nos ayuda a recordar los temas más importantes del documento. Entonces, los términos índice se usan para

indexar y resumir el contenido del documento. Por lo general, los términos índice son sustantivos porque acarrean un significado por fuerza propia y por ello la semántica asociada a ellos es más fácilmente identificable. Los adjetivos, los adverbios y los conectivos son menos útiles porque su rol es complementario. Sin embargo, puede resultar provechoso considerar todas las palabras distintas en una colección de documentos como términos índice. Algunos motores de búsqueda de la Web adoptan este enfoque, en cuyo caso la visión lógica de los documentos es de **texto completo**.

No todos los términos índice son igualmente importantes o útiles a la hora de describir el contenido de un documento. Algunos son más amplios o vagos que otros. Decidir qué importancia tiene un término para resumir el contenido de un documento no es una tarea trivial. Aun así, hay algunas propiedades de un término índice que nos dan una idea cuantitativa de su potencial representatividad.

Consideremos una colección con cien mil documentos. Una palabra que aparece en todos y cada uno de los documentos es absolutamente inútil como término índice pues no expresa nada acerca de en qué documentos puede estar interesado el usuario. En el otro extremo, una palabra que se encuentra en sólo cinco documentos es sumamente útil porque acota considerablemente el espacio de documentos que pueden interesar al usuario. De lo expuesto, debe quedar claro que términos índice distintos tienen **diferente relevancia** como descriptores del contenido de los documentos. Este efecto es asimilado mediante la asignación de **pesos** numéricos a cada término índice de un documento.

Supongamos que t es la cantidad de términos en el sistema, y k_i es un término índice genérico.

$K = \{k_1, k_2, ..., k_t\}$ es el conjunto de todos los términos índice.

Un peso $w_{ij} \geq 0$ se asocia a cada término índice k_i de un documento d_j. El peso cuantifica la importancia de k_i para describir el contenido semántico de d_j. Si un término no aparece en un documento su peso $w_{ij} = 0$. Con cada documento d_j se asocia un vector $\vec{d}_j = (w_{1j}, w_{2j}, ..., w_{tj})$. La función g_i nos da el peso asociado al término k_i en cualquier vector t-dimensional, i.e. $g_i(\vec{d}_j) = w_{ij}$.

Como se verá más adelante, normalmente asumiremos que los pesos de los términos índice son **mutuamente independientes**. Hacer esta suposición significa que conocer el peso w_{ij} asociado con el par (k_i,d_j) no nos dice nada acerca del peso $w_{i+1,j}$ asociado al par (k_{i+1},d_j). Esto es una simplificación ya que es claro que las ocurrencias de los términos en un documento no carecen de correlación.

Consideremos, por caso, que los términos *real* y *academia* son representativos de un documento que habla sobre la Real Academia. Frecuentemente en este documento, la aparición de uno de los términos atrae la aparición del otro. Entonces, sus pesos podrían reflejar esta correlación. La razón de asumir la fuerte suposición de mutua independencia es que simplifica el cálculo de los pesos de términos índice, y permite una computación más rápida del *ranking*. De todas maneras, no se pudo demostrar que el uso de correlaciones entre términos sea ventajoso con colecciones generales.

2.2.1.1. Modelo Booleano

Es un modelo simple que se basa en la teoría de conjuntos y en el álgebra Booleana. La idea de conjunto es sencilla y permite que un usuario se dé una idea de cómo funciona este tipo

de sistemas RI. Las consultas expresadas como expresiones Booleanas tienen una semántica precisa. Vemos que el formalismo subyacente al modelo es nítido.

Tiene, sin embargo, algunas contras. El **criterio binario** para decidir si un documento es o no relevante impide la noción de una escala de graduación, lo que atenta contra la buena *performance* o desempeño del sistema. Por ello, se puede pensar que el modelo Booleano es más un sistema de recuperación de datos (SRD) que uno de información (SRI).

Muchas veces es difícil y tortuoso traducir una necesidad de información a una expresión Booleana. Por ello, los usuarios tienden a usar expresiones simples.

El modelo Booleano considera que un término índice está presente o ausente en un documento, por lo que también los **pesos son binarios**, i.e. $w_{ij} \in \{0,1\}$. Una consulta q se compone de términos índice conectados por tres conectivos: *not*, *and*, *or*.

Para arribar al cálculo de **R**, la similitud entre documentos y consultas, debemos incluir la idea de que cualquier expresión Booleana puede ser transformada a la forma normal disyuntiva (FND). Por ejemplo, $q = k_a \wedge (k_b \vee k_c)$ se escribe $q_{FND} = (1,1,1) \vee (1,1,0) \vee (1,0,0)$, donde cada componente es un vector asociado a la tupla (k_a, k_b, k_c). Así tenemos que

$$\text{sim}(d_j, q) = \begin{cases} 1 & \text{si } \exists\, q_{cc} \mid (q_{cc} \in q_{FND}) \wedge (\forall\, k_i : g_i(d_j) = g_i(q_{cc})) \\ 0 & \text{si no} \end{cases}$$

Si la similitud es 1, el modelo predice que el documento d_j es relevante a la consulta q (aunque pueda no serlo). De otro modo, la predicción es que el documento no es relevante.

No hay una idea de correspondencia (*matching*) o **similitud parcial** a las condiciones de la consulta. Un documento que contenga 9 de 10 los términos conectados con *and* en una consulta, no será considerado relevante.

Otra desventaja del modelo es que la exigencia de una similitud total puede redundar en resultados muy numerosos o muy escasos, medidos en cantidad de documentos. Se sabe además que el pesaje de términos puede producir una mejora sustancial del desempeño de la recuperación (*retrieval performance*). Esta certeza nos lleva al modelo vectorial.

2.2.1.2. Modelo Vectorial

El modelo vectorial reconoce la limitación que imponen los pesos binarios, y propone un marco de modelado en donde la correspondencia parcial es posible. Se asignan entonces **pesos no binarios** a los términos índice en consultas y documentos. Estos pesos son usados para calcular el **grado de similitud** entre cada documento almacenado en el sistema y la consulta de usuario.

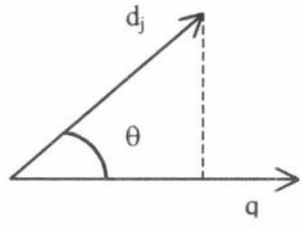
Mediante el ordenamiento de los documentos recuperados en orden decreciente según su grado de similitud, el modelo vectorial tiene en cuenta a los documentos que se corresponden con los términos de la consulta de manera parcial.

El efecto resultante es que el conjunto respuesta de documentos ordenados es mucho más preciso (en el sentido de que se acerca mejor a la necesidad de información del usuario) que el conjunto de documentos respuesta obtenido por el modelo Booleano.

Los w_{ij} son positivos y no binarios. Los términos índice de la consulta también son pesados. Tenemos entonces que $w_{i,q}$ es el peso asociado al término k_i en la consulta q. El vector

consulta se define por $\vec{q} = (w_{1,q}, w_{2,q}, ..., w_{t,q})$ donde t nuevamente es la cantidad de términos distintos en toda la colección indexada por el sistema. El vector $\vec{d}_j = (w_{1j}, w_{2j}, ..., w_{tj})$ se representa como hasta ahora.

Podemos representar entonces a la consulta q y al documento d_j como dos vectores en el espacio t-dimensional, como se ve en la figura.



El modelo vectorial propone evaluar el grado de similitud del documento d_j con respecto a la consulta q, como la correlación entre los vectores $\vec{q}$ y $\vec{d}_j$. Dicha correlación puede cuantificarse, por ejemplo, por el **coseno del ángulo** entre estos dos vectores. Es decir,

$$sim(d_j, q) = \cos\theta = \frac{<\vec{d}_j, \vec{q}>}{|\vec{d}_j| . |\vec{q}|}$$

donde el denominador es el producto de las normas de los vectores. El efecto que tiene la norma es el de medir la longitud de un vector en el espacio t-dimensional. En este caso, dividir a $\vec{d}_j$ por su norma resulta en un vector con la misma dirección y sentido, pero de longitud 1. El mismo efecto produce la división del vector consulta $\vec{q}$ por su norma.

$$\left|\vec{d}_j\right| = \sqrt{\sum_{k \in K} (w_{k,j})^2}$$

El numerador es el producto interno entre los vectores, luego normalizados por el denominador, lo que nos da el coseno del ángulo entre ellos.

$$<\vec{d}_j, \vec{q}> = \sum_{i=1}^{t} w_{ij} \cdot w_{iq}$$

El factor $\left|\vec{q}\right|$ no afecta al *ranking*, es decir el orden de los documentos, porque es el mismo para todos los documentos.

Como $w_{ij} \geq 0$ y $w_{iq} \geq 0$, la función sim (q, d_j) varía desde 0 hasta +1. Entonces, en lugar de intentar predecir si un documento es relevante o no lo es, el modelo vectorial ordena los documentos de acuerdo con su **grado de similitud** con la consulta. Un documento puede ser recuperado aun cuando su correspondencia con la consulta sea sólo parcial. Esto permite

establecer, si uno lo desea, un umbral para sim (q, dj), de manera de retornar sólo los documentos con un grado de similitud por encima de dicho umbral.

Veamos ahora cómo se computan los pesos de los términos, para poder calcular los *rankings*.

Existen muchas maneras distintas de calcular los pesos de los términos índice. Sin entrar en un estudio exhaustivo de dichas técnicas, nos concentraremos más bien en puntualizar las ideas más efectivas para calcular los pesos.

La idea fundamental está relacionada con el principio básico de las técnicas de *clustering*.

Dada una colección C de objetos y una descripción **vaga** de un conjunto A, el objetivo de un algoritmo simple de *clustering* puede ser separar la colección C de objetos en dos conjuntos: uno con objetos relacionados con A, y otro con objetos no relacionados con A. Pero una descripción vaga nos impide disponer de suficiente información como para decidir con precisión qué objetos están o no en el conjunto A.

Por ejemplo, un usuario podría buscar casas cuyo precio es **comparable** a la casa en donde vive. Al no estar claro el significado del término comparable, no hay una descripción precisa del conjunto A.

Lo que necesitamos es la decisión de qué documentos predecimos relevantes, y qué otros documentos predecimos como no relevantes, siempre respecto a la consulta del usuario.

Para observar el problema de RI como uno de *clustering*, podemos pensar a los documentos como una colección C de objetos y pensar a la consulta del usuario como una (vaga) especificación del conjunto A de objetos. Hemos reducido el problema de RI a determinar qué documentos están en A y cuáles no, es decir a un problema de *clustering*.

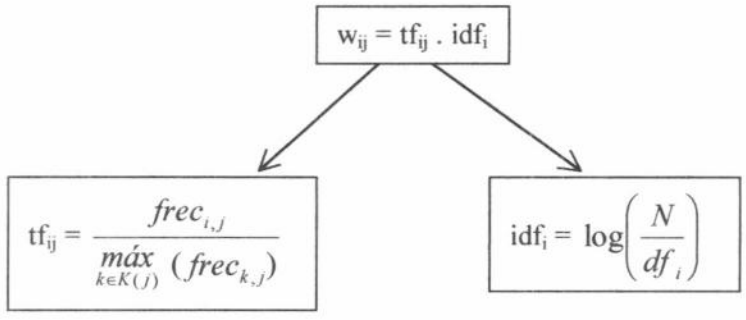
Debemos resolver ahora las dos cuestiones que caracterizan este tipo de problemas. Una es qué atributos describen mejor los objetos de A. La otra, qué atributos distinguen los objetos en el conjunto A del resto de los objetos en la colección C.

En el modelo vectorial, la primera medida (análoga a la similitud *intracluster*) se cuantifica midiendo la **frecuencia** absoluta de un término k_i dentro de un documento d_j. Es decir, la cantidad de veces que se menciona al término en el texto del documento. Se suele referir a tal frecuencia del término como el **factor tf** (*term frequency*) y provee una medida de cuán bien describe ese término al contenido del documento.

La segunda medida, que puede analogarse a la disimilitud *intercluster*, se cuantifica midiendo la frecuencia inversa de un término k_i entre todos los documentos de la colección. Es decir, la cantidad de documentos en los que ese término aparece. Este factor es llamado también **frecuencia inversa en documentos** o **factor idf** (*inverse document frequency*).

Como ya ha sido mencionado, la motivación por el uso del factor idf es que los términos que aparecen en muchos documentos no son muy útiles a la hora de distinguir un documento relevante de uno no relevante.

Los esquemas de pesaje de términos más conocidos utilizan pesos dados por:



Donde:

- N es la cantidad de documentos en la colección.
- $frec_{ij}$ es la cantidad de veces que el término k_i aparece en el documento d_j;
- El valor tf_{ij} es igual a $frec_{ij}$ normalizado en el documento d_j.
- $K(j)$ es el conjunto de términos en el documento d_j.
- df_i es la cantidad de documentos de la colección que contienen el término k_i.

Para el pesaje de los términos de la consulta, se sugiere:

$$w_{iq} = (0.5 + 0.5\ tf_{iq}).idf_i$$

Donde tf_{iq} se define para q de manera análoga a como se usó para un documento d_j.

A este tipo de estrategias se las conoce como esquemas tf-idf.

Tenemos entonces que las **ventajas** principales del modelo vectorial son:

- El esquema de pesaje de términos mejora el desempeño de la recuperación.
- La estrategia de correspondencia parcial permite recuperar documentos que se aproximan a las condiciones de la consulta.
- La fórmula de cálculo del coseno para calcular puntajes ordena a los documentos de acuerdo a su grado de similitud respecto a la consulta.

Teóricamente, el modelo vectorial tiene la **desventaja** de que se asume que los términos índice son mutuamente independientes (las ecuaciones anteriores no contemplan dependencias entre términos). Sin embargo, en la práctica, considerar las dependencias entre los términos puede ser contraproducente. A raíz de la localidad de muchas dependencias entre términos, su aplicación indiscriminada a todos los documentos de la colección puede ir en detrimento del desempeño general del sistema.

A pesar de su simpleza, el modelo vectorial es una estrategia de *ranking* sobresaliente con colecciones (sobre temas) generales. Produce conjuntos respuesta ordenados que son difíciles de superar sin expansión de la consulta o retroalimentación del usuario en el contexto del modelo vectorial. Una gran variedad de métodos han sido comparados al modelo vectorial y el consenso es que, en general, este modelo es superior o casi tan bueno como las alternativas conocidas. Además, es simple y rápido.

Por estas razones, el modelo vectorial es un modelo de recuperación muy utilizado en nuestros días.

2.2.1.3. Modelo Probabilístico

El último modelo que presentaremos es el probabilístico.

Es un modelo clásico que intenta capturar el problema de RI dentro de un marco probabilístico.

La idea fundamental es la siguiente. Dada una consulta de usuario, existe un conjunto de documentos que contiene exactamente los documentos relevantes y ningún otro. Refirámonos a este conjunto como el conjunto respuesta **ideal** (CRI). Si tuviéramos la descripción de este conjunto, recuperaríamos sus documentos sin problemas.

Así, interpretamos al proceso de consulta como especificando las propiedades de un CRI (lo que es análogo a interpretar el problema de RI como uno de *clustering*). El problema es que no sabemos exactamente cuáles son estas propiedades. Todo lo que sabemos es que hay términos índice cuya semántica nos puede ayudar a caracterizar estas propiedades. Como en el momento de la consulta no son conocidas estas propiedades, debe realizarse una aproximación a lo que podrían ser. Esta aproximación inicial nos permite generar una descripción probabilística preliminar del CRI que es usada para recuperar el primer conjunto de documentos.

Se inicia entonces una interacción con el usuario con el propósito de mejorar la descripción probabilística del CRI. El usuario observa los documentos recuperados y decide cuáles son relevantes y cuáles no lo son. El sistema usa esta información para refinar la descripción del CRI. Repitiendo este proceso varias veces, se espera que tal descripción evolucione y se vaya acercando a la descripción real del CRI.

La suposición fundamental del modelo probabilístico es la siguiente.

Principio probabilístico. Dada una consulta q y un documento d_j de la colección, el modelo probabilístico intenta estimar la probabilidad de que el usuario encuentre interesante (relevante) al documento d_j. El modelo asume que esta probabilidad de relevancia depende sólo de las representaciones de la consulta y el documento. También supone que hay un subconjunto de todos los documentos que el usuario prefiere como respuesta a su consulta q. Nombraremos R a tal CRI que debe maximizar la probabilidad de relevancia para el usuario. Se predice que los documentos del conjunto R son **relevantes** a la consulta. Los documentos que no están en este conjunto son **no relevantes**.

Ante una consulta q, el modelo probabilístico asigna a cada documento, como medida de similitud hacia la consulta, la razón $P(d_j$ relevante para q) / $P(d_j$ no relevante para q) que computa las chances de que d_j sea relevante para la consulta q.

Los pesos de los términos índice son binarios, i.e. $w_{ij} \in \{0,1\}$, $w_{iq} \in \{0,1\}$ como en el modelo Booleano, pero distinto al modelo vectorial. Una consulta q es un subconjunto de términos índice.

Supongamos que R es el conjunto conocido (o aproximado inicialmente) como relevante. Notamos al complemento de R como $\overline{R}$ (el conjunto de los documentos no relevantes).

Sea $P(R \mid \vec{d}_j)$ la probabilidad de que el documento d_j sea relevante a la consulta q.

Sea $P(\overline{R} \mid \vec{d}_j)$ la probabilidad de que el documento d_j sea no relevante para q.

La similitud sim (d_j, q) del documento d_j a la consulta q se define como la razón

$$\text{sim}(d_j, q) = \frac{P(R \mid \vec{d_j})}{P(\overline{R} \mid \vec{d_j})}$$

Por el Teorema de Bayes

$$\text{sim}(d_j, q) = \frac{P(\vec{d_j} \mid R) \times P(R)}{P(\vec{d_j} \mid \overline{R}) \times P(\overline{R})}$$

En esta fórmula, $P(\vec{d}_j \mid R)$ es la probabilidad de seleccionar al azar al documento d_j de entre los del conjunto R de documentos relevantes. Además, P(R) es la probabilidad de que un documento seleccionado al azar entre los de la colección completa sea relevante. Los significados de $P(\vec{d}_j \mid \overline{R})$ y $P(\overline{R})$ son análogamente complementarios.

Como P(R) y $P(\overline{R})$ son iguales para todos los documentos de la colección, podemos decir

$$\text{sim}(d_j, q) \sim \frac{P(\vec{d_j} \mid R)}{P(\vec{d_j} \mid \overline{R})}$$

Asumiendo independencia de los términos índice, e ignorando factores que son constantes en el contexto de una misma consulta se llega finalmente a

$$\text{sim}(d_j, q) \sim \sum_{i=1}^{t} w_{iq} \times w_{ij} \times \left(\log\left(\frac{P(k_i \mid R)}{1 - P(k_i \mid R)} \right) + \log\left(\frac{1 - P(k_i \mid \overline{R})}{P(k_i \mid \overline{R})} \right) \right)$$

que es una expresión clave para computar los puntajes en el modelo probabilístico.

Aquí, $P(k_i \mid R)$ es la probabilidad de que el término índice k_i esté presente en un documento seleccionado al azar del conjunto R. Siendo $P(\overline{k_i} \mid R) + P(k_i \mid R) = 1$ pues el primer término es la probabilidad de que esta vez el término k_i no esté presente en R.

Como al principio no conocemos al conjunto R, es menester divisar un método para calcular las probabilidades $P(k_i \mid R)$ y $P(k_i \mid \overline{R})$. Hay muchas alternativas posibles para encarar tal computación. No las discutiremos en profundidad, sino que daremos la idea general.

Se puede asumir que al comienzo, cuando aún no se recuperaron documentos:

- la $P(k_i \mid R)$ es constante para todos los términos k_i, típicamente igual a 0.5.
- la distribución de los términos índice en los documentos no relevantes puede aproximarse por la distribución de los términos en toda la colección, o sea n_i / N.

Entonces:

$$P(k_i \mid R) = 0.5$$

y

$$P(k_i \mid \overline{R}) = \frac{n_i}{N}$$

donde, como se definió oportunamente, n_i es la cantidad de documentos que contienen el término k_i y N es la cantidad total de documentos en la colección.

Con esta aproximación inicial, podemos recuperar documentos que contengan los términos de la consulta y asignarles un puntaje probabilístico inicial. Luego se puede mejorar este *ranking* inicial, pero está más allá del alcance de este trabajo.

La principal **ventaja** del modelo probabilístico, en teoría, es que los documentos están ordenados en orden decreciente de su **probabilidad de ser relevantes**.

Las **desventajas** incluyen:

- la necesidad de adivinar la separación inicial entre conjuntos de documentos relevantes y no relevantes.
- no se tiene en cuenta la frecuencia con que un término ocurre dentro de un documento (todos los pesos son binarios).
- la adopción de la suposición de independencia entre términos índice. Aunque, como se vio en el modelo vectorial, no está claro que esto sea del todo malo en la práctica.

2.3. Operaciones sobre el texto

Como hemos discutido en secciones anteriores, no todas las palabras son igualmente significativas para representar la semántica de un documento. En el lenguaje escrito, algunas palabras poseen mayor **significado** que otras. En general, los **sustantivos** (o grupos de sustantivos) son los que mejor representan el contenido de un documento. En consecuencia, se considera redituable el preprocesamiento del texto de los documentos de la colección para identificar los términos que serán usados como **términos índice** (***index terms***).

Durante esta fase de preprocesamiento, se pueden realizar otras operaciones útiles sobre el texto, tales como la eliminación de palabras frecuentes o poco representativas (*stopwords*), la reducción de una palabra a su raíz gramatical (*stemming*), la construcción de un *thesaurus*, y el uso de la compresión.

Ya sabemos que representar documentos mediante conjuntos de términos índice nos brinda una representación un tanto imprecisa de la semántica de los documentos presentes en la colección. Por caso, un término como 'el' no tiene un significado por sí mismo y puede llevar a la recuperación de algunos documentos que no tienen relación con la presente consulta.

El uso del conjunto de todas las palabras presentes en la colección para indexar sus documentos genera demasiado **ruido** en la tarea de recuperación. Una manera de reducir este ruido es reducir, a su vez, el conjunto de palabras que pueden ser usadas para referirse (i.e. indexar) a los documentos. Es por eso que puede verse al preprocesamiento de los documentos de la colección como un proceso de control del tamaño del vocabulario (i.e. la cantidad de palabras diferentes que son usadas como términos índice). Se aspira a que el uso de un vocabulario controlado produzca una mejora en el desempeño de la recuperación.

Si bien el uso del control del tamaño del vocabulario es una técnica común en los sistemas comerciales, introduce un paso adicional en el proceso de indexación que, a menudo, no es percibido por los usuarios. Como resultado, un usuario común puede sorprenderse con la aparición de algunos documentos recuperados y la ausencia de otros que esperaba ver. Por ejemplo, puede recordar que cierto documento contiene la frase 'ved el trono a la noble' y notar que dicho documento no está presente en los primeros 20 documentos recuperados en respuesta a su consulta (porque el vocabulario controlado no contiene ni a 'el' ni a 'la').

Debe quedar claro, entonces, que a pesar del potencial mejoramiento del desempeño en la recuperación, las transformaciones del texto realizadas en tiempo de preprocesamiento pueden dificultar la interpretación que hace el usuario de la tarea de recuperación. Reconociendo este problema, algunos motores de búsqueda de la Web evitan por completo las operaciones textuales y simplemente indexan todas las palabras del texto. La idea es que, si bien se tiene un índice más ruidoso, la tarea de recuperación es más sencilla y más intuitiva para un usuario común, ya que puede ser interpretada como una búsqueda en la totalidad del texto.

Además del preprocesamiento de documentos, pueden intentarse otros tipos de operaciones sobre los documentos, con el objetivo de mejorar el desempeño en la recuperación. Entre ellos distinguimos la construcción de un *thesaurus* que representa las relaciones entre conceptos y el agrupamiento (*clustering*) de documentos relacionados entre sí.

La normalización del texto y la construcción de un *thesaurus* son estrategias que apuntan a mejorar la precisión de los documentos recuperados. Pero con el advenimiento de las colecciones digitales gigantescas, mejorar la eficiencia (en términos de tiempo) del proceso de recuperación se ha transformado en algo crítico. De hecho, los sistemas de RI de la Web están actualmente más preocupados por reducir el tiempo de respuesta que en mejorar los valores de precisión y *recall*. La compresión del texto puede ser una alternativa en estos casos.

Un buen algoritmo de compresión es capaz de reducir el texto al 30-35% de su tamaño original. Se necesita menos espacio para almacenarlo y menos tiempo para transmitirlo por una línea de comunicación.

El preprocesamiento de un documento es un procedimiento que se puede dividir fundamentalmente en cinco operaciones o transformaciones sobre el texto:

1) El análisis léxico del texto con el propósito de tratar a dígitos, guiones, signos de puntuación y la capitalización de las letras (mayúsculas / minúsculas).

2) La eliminación de *stopwords* con el fin de descartar palabras con muy bajo poder discriminatorio en lo que se refiere a la recuperación de documentos.

3) La radicación gramatical (*stemming*) de las palabras restantes, con el objetivo de eliminar afijos (i.e. prefijos y sufijos) y permitir la recuperación de documentos que contengan variaciones de los términos presentes en la consulta (e.g. conecta, conectar, conectado, conectando, etc.).

4) La selección de términos índice para determinar qué palabras / raíces (*stems*) o grupos de palabras serán de utilidad como elementos de indexación. A menudo, la elección de un término índice se relaciona con la naturaleza sintáctica de la palabra. En realidad, los sustantivos transportan más semántica que los adjetivos, adverbios o verbos.

5) La construcción de estructuras de categorización de términos como los *thesauri*, o la extracción de la estructura representada directamente en el texto, para permitir la expansión de la consulta original mediante términos relacionados (un procedimiento útil).

Como se vio en la primera sección de este capítulo, la visión lógica de los documentos que resulta de cada una de las fases mencionadas arriba evoluciona desde el texto completo a un conjunto de términos índice de alto nivel.

2.4. Evaluación de la Recuperación

Cualquier sistema programado en una computadora debe proporcionar la funcionalidad para la que fue concebido. Por lo tanto, el primer paso de evaluación a ser considerado es un análisis funcional en el que las funcionalidades especificadas del sistema son probadas (*test*) una por una. Luego de pasar esta fase, se puede proceder a evaluar el desempeño (*performance*) del sistema.

Las medidas más comunes de desempeño de un sistema son tiempo y espacio. A menor tiempo de respuesta, y menor espacio utilizado, mejor será considerado el sistema. Hay un balance inherente entre complejidad en espacio y en tiempo que implica decidir por uno en detrimento del otro.

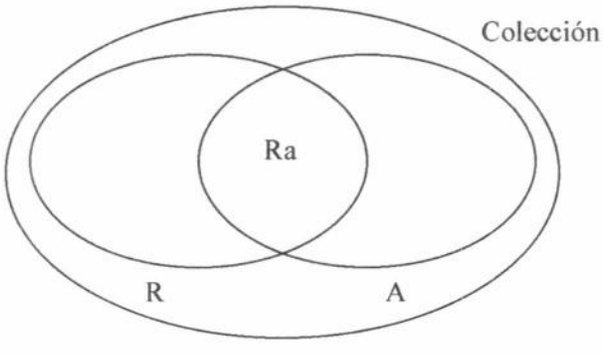
En un sistema diseñado para recuperar datos, el tiempo de respuesta y el espacio requerido son las métricas más interesantes y las que se usan normalmente para evaluar al sistema. Se analiza el desempeño de las estructuras de índice, la interacción con el sistema operativo y otros aspectos. Solemos referirnos a esta forma de evaluación como de **evaluación de desempeño**.

En un sistema diseñado para recuperar información, son también interesantes otras métricas, además del espacio y el tiempo. Como la consulta del usuario es intrínsecamente ambigua, los documentos recuperados no son respuestas exactas y deben ordenarse de acuerdo a su relevancia a la consulta. Este orden por relevancia introduce un componente que no está presente en los sistemas de recuperación de datos y juega un rol central en la recuperación de información (RI). Es por ello que los sistemas de RI imponen la necesidad de evaluar cuán preciso es el conjunto respuesta. A este tipo de evaluación se la denomina **evaluación del desempeño en la recuperación**.

Tal evaluación se basa generalmente en una colección referencia de prueba y en una medida de evaluación. La colección referencia de prueba consta de una colección de documentos, un conjunto de requerimientos de información ejemplo, y un conjunto de documentos relevantes (provistos por especialistas) para cada requerimiento de información o consulta.

Dada una estrategia de recuperación S, la medida de evaluación cuantifica (para cada consulta) la **similitud** entre el conjunto de documentos recuperados por S y el conjunto de documentos relevantes provistos por los especialistas. Esto nos da una estimación de la **bondad** de la estrategia de recuperación S.

Consideremos ahora una consulta ejemplo I y su conjunto R de documentos relevantes. Sea |R| la cantidad de elementos en dicho conjunto. Asumamos que una determinada estrategia de recuperación (que estamos evaluando) procesa la consulta I y genera el conjunto de documentos respuesta A. Tomemos |Ra| la cantidad de documentos en la intersección de R y A.



Las medidas de **precisión** y **recuperación** (***recall***) se definen como sigue:

- Recuperación es la fracción de documentos relevantes (R) que han sido recuperados.

$$\text{Recuperación} = \frac{|Ra|}{|R|}$$

- Precisión es la fracción de documentos recuperados (A) que son relevantes.

$$\text{Precisión} = \frac{|Ra|}{|A|}$$

Tal como están definidas, la recuperación y la precisión suponen que todos los documentos del conjunto respuesta A han sido examinados (o al menos vistos). Sin embargo, no es común que el usuario pueda ver todos los documentos de A a la vez. Por el contrario, los documentos de A están ordenados según algún grado de relevancia (i.e. se genera un ordenamiento o *ranking*).

El usuario entonces examina esta lista ordenada comenzando por el primer documento, ubicado al tope de la lista. En esta situación, las medidas de precisión y recuperación varían según cuánto ha examinado el usuario al conjunto respuesta A. Por lo tanto, la evaluación correcta consiste en dibujar una curva de precisión contra recuperación de la siguiente manera.

Concentrémonos en un requerimiento de información para la cual se formula una consulta q. Asumamos que R_q contiene los documentos relevantes para q. De acuerdo a los especialistas, son 10 los documentos relevantes para q:

$$R_q = \{d3, d5, d9, d25, d39, d44, d56, d71, d89, d123\}$$

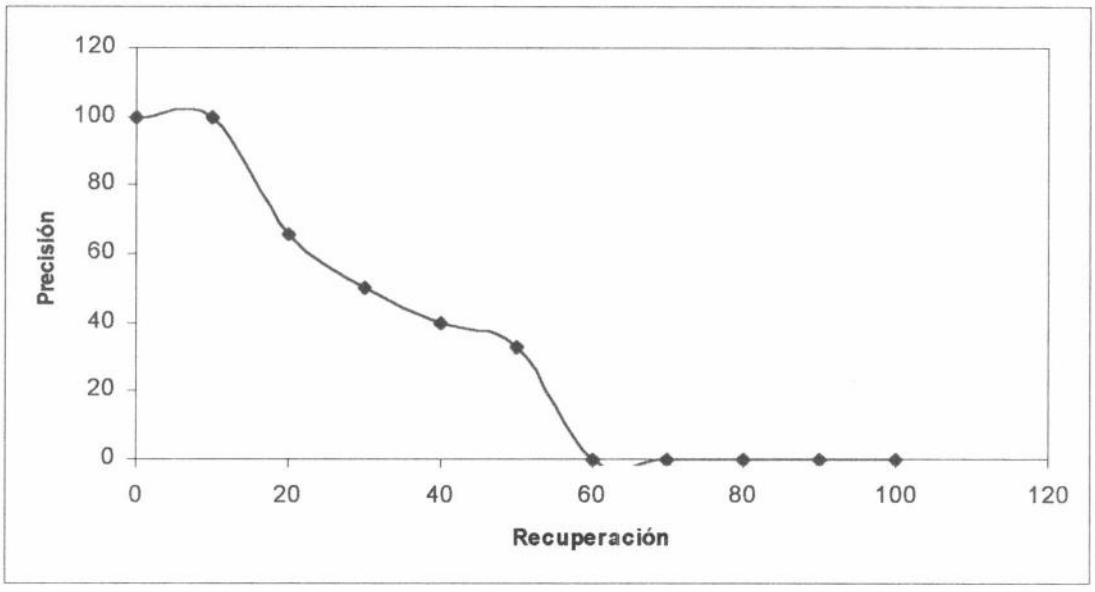
Consideremos un nuevo algoritmo de recuperación que ha sido diseñado recientemente. Supongamos que este algoritmo devuelve, para la consulta q, el siguiente ordenamiento de los documentos en el conjunto respuesta.

Ranking para la consulta q:

1. d123 •	6. d9 •	11. d38
2. d84	7. d511	12. d48
3. d56 •	8. d129	13. d250
4. d6	9. d187	14. d113
5. d8	10. d25 •	15. d3 •

Los documentos que son relevantes a la consulta son marcados con una viñeta, a continuación del número de documento. Observando este *ranking*, notamos que el primer documento d123 es relevante y que corresponde al 10% de los documentos relevantes en el conjunto R_q. Por lo que podemos decir que tenemos una precisión del 100% al 10% de recuperación.

El documento d56 está en el lugar 3, por lo que tenemos una precisión de 66% (2 documentos de 3 son relevantes) en el 20% de recuperación (2 documentos de 10 relevantes han sido observados). Podemos dibujar la curva de precisión vs. recuperación.



Como vemos, la precisión más allá del 50% de *recall* cae a 0 porque no todos los documentos relevantes han sido recuperados. La curva de precisión se basa generalmente en 11 niveles **estándar** de recuperación que son 0%, 10%, 20%, ..., hasta 100%.

En el ejemplo anterior, los valores de precisión y recuperación son para una sola consulta.

Lo que se hace generalmente, sin embargo, es evaluar a los algoritmos de *ranking* en su desempeño para varias consultas. Para cada una de ellas, se genera una curva de precisión-recuperación diferente. Para evaluar el desempeño en la recuperación de un algoritmo sobre todas las consultas de prueba, promediamos las precisiones en cada nivel de recuperación:

$$\overline{P}(r) = \frac{1}{N_q} \times \sum_{i=1}^{N_q} P_i(r)$$

donde $\overline{P}(r)$ es la precisión promedio en el nivel r de recuperación, N_q es la cantidad de consultas utilizadas, y $P_i(r)$ es la precisión en el nivel r de recuperación para la i-ésima consulta.

Cuando los niveles de recuperación para distintas consultas no coinciden, se interpolan los valores en los 11 niveles estándar de recuperación para así poder promediar las precisiones en esos niveles.

Otro enfoque posible es calcular la precisión promedio a determinados **valores de corte de documentos**. Por ejemplo, se podría calcular la precisión promedio cuando se han visto 5, 10, 15, 20, 50 y 100 documentos relevantes.

Si bien las medidas de precisión y recuperación son muy valiosas para evaluar la calidad de un sistema de recuperación de información, también tienen sus **problemas**:

- La estimación del máximo nivel de recuperación requiere conocer detalladamente a todos los documentos de la colección. Con **colecciones grandes**, no es posible disponer de este conocimiento, lo que implica que las medidas de recuperación no pueden estimarse con exactitud.

- Precisión y recuperación son dos medidas que capturan aspectos diferentes de un conjunto de documentos recuperados. En algunas situaciones, combinar ambas en una sola medida puede ser más apropiado.
- Son medidas de la efectividad sobre un conjunto de consultas que se procesan en lote (*batch*). Pero en los modernos sistemas, la interactividad es un aspecto clave del proceso de recuperación. Se prefieren entonces métricas de **informatividad** de dicho proceso.
- Son medidas inadecuadas cuando no hay un orden bien definido entre los documentos recuperados.

Existen, como se aprecia, otras medidas de evaluación de desempeño que pueden ser más apropiadas, más sintéticas o incluso orientar más el análisis hacia la percepción o satisfacción del usuario. El propósito de esta sección no es el de hacer un reporte exhaustivo de estas técnicas, sino el de presentar el problema de evaluar un sistema de recuperación de información. Se expusieron entonces las dos medidas más utilizadas en la literatura, haciendo hincapié tanto en su importancia como en sus limitaciones.

Capítulo 3. Sistemas de recuperación de información en la WWW

En los últimos 20 años, el área de recuperación de la información creció más allá de los objetivos iniciales de indexación de texto o búsqueda de documentos útiles en una colección.

Hoy en día, la investigación en RI abarca el modelado, categorización, clasificación y filtrado de documentos, arquitectura de los sistemas, interfaces de usuario, visualización de datos, lenguajes, etc. A pesar de su madurez, hasta no hace mucho la RI era vista como un área lateral de interés sólo para bibliotecarios o expertos en información. Esa visión permaneció hasta los comienzos de los 1990s, donde un solo hecho cambió de una vez y para siempre todas esas percepciones: la aparición de la World Wide Web.

La Web se ha convertido en el repositorio universal del conocimiento y la cultura humana, lo que ha permitido compartir ideas e información en una escala sin precedentes y nunca antes vista. Su éxito se basa en la concepción de una interfaz de usuario estándar que es siempre la misma, más allá del entorno computacional o plataforma utilizada para ejecutar dicha interfaz.

Como consecuencia, el usuario está ajeno a los detalles de los protocolos de comunicación, la ubicación de las máquinas, y los sistemas operativos subyacentes. Inclusive cualquier usuario puede crear sus propios documentos Web y hacerlos apuntar a cualesquiera otros documentos Web sin restricciones. Éste es un aspecto clave porque transforma a la Web en un medio de publicación accesible para todos, casi sin costos.

Este universo sin fronteras atrajo la atención de millones de personas de todas partes del mundo desde el principio. Sin dudas, está provocando una revolución en la manera en que la gente usa las computadoras y realiza sus tareas diarias. Por caso, la banca electrónica y el comercio electrónico se han popularizado y han generado ganancias por miles de millones de dólares.

Pero a pesar de este éxito fulminante, la Web trajo consigo nuevos problemas. El encontrar información útil en la Web es un proceso a menudo tedioso y difícil. Por ejemplo, para satisfacer sus necesidades de información, el usuario puede navegar por el espacio de los enlaces Web (i.e. el **hiperespacio**) buscando información de interés. Pero como el hiperespacio es vasto y muchas veces desconocido, tal esfuerzo navigacional es con frecuencia ineficiente. Con usuarios ingenuos o no preparados, el problema resulta más agudo, lo que puede frustrar por completo todos sus esfuerzos.

El mayor obstáculo es la ausencia de un modelo de datos subyacente bien definido para la Web, lo que implica que la definición y estructura de la información es a menudo de baja calidad. Estas dificultades han atraído un interés renovado en la RI y sus promisorias técnicas. De la noche a la mañana, la disciplina de RI se ha ganado un lugar en el centro del escenario junto a las demás tecnologías.

El crecimiento exponencial que experimenta la Web es bien conocido, por lo que se ha venido duplicando en períodos desde 4 hasta 9 meses, llegando a tener alrededor de 4000 millones de páginas (5/2001) la Web de superficie o indexable por los motores de búsqueda (MB), lo que representa 40 TB de texto, del cual cerca del 60% es texto útil (sin código HTML), lo que nos da 24 TB de texto útil en la Web superficial [Archive].

Por supuesto que también hay otros formatos o tipos de archivo que acrecientan estos números como lo son las imágenes, videos, audio, otros formatos de texto (PDF, DOC, RTF), etc. Podemos ver a la Web como una inmensa base de datos sin estructura, lo que promueve la necesidad de herramientas eficientes para manejar, recuperar y filtrar la información proveniente de este reservorio de datos.

Sin embargo, como aclaramos en el capítulo anterior, lo que nos interesa es la recuperación de la información, y no la recuperación de los datos, que pareció ganar terreno en las intranets dada la necesidad de extraer o inferir información a partir de datos para dar soporte a los procesos de decisión, una tarea denominada minería de datos (*data mining*).

La otra Web, la Web no indexable, la que no fue ni puede ser por ahora utilizada ni aprovechada por los MBs, es mucho más grande que la Web visible. Se calculó en el año 2000 [Deepweb] que era entre 400 y 550 veces más grande que la Web superficial (1000 millones en el estudio), que el 95% era de acceso público e irrestricto, lo que pasado a las cifras de este año nos daría alrededor de 2 millones de millones, es decir 2 billones de páginas representando 20.000 TB de texto, finalmente 12.000 TB de texto útil. Suponiendo siempre que el tamaño de una página es de 10 KB, una cifra por demás conservadora.

Otro análisis que predecía en 7/2000 los valores actuales de la Web es [Cyveil].

Nos concentraremos en el **texto** porque, si bien hay técnicas para buscar imágenes y otros datos no textuales, no se las puede aplicar aún a gran escala. Se enfatiza también en la búsqueda sintáctica, es decir, buscamos documentos Web que tienen en su texto palabras o patrones especificados por el usuario. Como se aclaró, tales palabras o patrones pueden no reflejar la semántica intrínseca del texto.

Un enfoque alternativo a la búsqueda sintáctica es realizar un análisis del texto basado en el lenguaje natural. A pesar de que las técnicas de preprocesamiento de lenguaje natural y extracción de la semántica textual no son nuevas, en verdad no son todavía muy efectivas y son sumamente costosas con grandes volúmenes de datos. Además, en muchos casos sólo son efectivas con textos bien estructurados, un *thesaurus*, e información de contexto.

Hay básicamente tres maneras de buscar en la Web. Dos de ellas son bien conocidas y usadas frecuentemente. La primera es la utilización de un MB para indexar una porción de los documentos de la Web en una base de datos de **texto completo**. La segunda es el uso de directorios Web, que clasifican en categorías temáticas a documentos Web seleccionados. La tercera y no del todo disponible aún, es buscar en la Web explotando su estructura de hiperenlaces.

Nos interesará sobre todo la primera de estas variantes, aunque muchas de las aseveraciones y dificultades expuestas pueden ser aplicadas para describir los otros dos mecanismos.

Expondremos entonces los desafíos que trae aparejados la búsqueda de información en la Web, junto con algunas otras estadísticas de la Web y de los buscadores.

3.1. Desafíos y características de la Web

Mencionaremos los principales inconvenientes planteados por la Web. Los podemos dividir en dos clases: los problemas con los datos en sí y los problemas relacionados con el usuario y su interacción con el sistema de recuperación. Los problemas relacionados con los datos son:

- **Datos distribuidos**: debido a la naturaleza esencial de la Web, los datos se diseminan entre muchas computadoras y plataformas. Estas computadoras, a su vez, están interconectadas sin una topología predeterminada y el ancho de banda disponible y la confiabilidad de las interconexiones de las redes varían enormemente.

- **Alto porcentaje de datos volátiles**: debido a la dinámica de Internet, se agregan y quitan nuevas computadoras y datos fácilmente (se estima que el 40% de la Web cambia cada mes). Tenemos también enlaces colgantes (sin destino válido) y problemas de reubicación cuando cambian los nombres de dominio, de archivos o simplemente son borrados.
- **Gran volumen**: el remanido pero real crecimiento exponencial de la Web plantea cuestiones de escalabilidad que son difíciles de afrontar y manejar.
- **Datos redundantes y sin estructura**: mucha gente dice que la Web es un hipertexto distribuido. Sin embargo, no es exactamente así. Cualquier hipertexto tiene un modelo conceptual detrás, que organiza y da consistencia a los datos y los hiperenlaces. Ello es difícilmente cierto en la Web, inclusive con documentos individuales. Asimismo, cada página HTML no está bien estructurada y algunos prefieren usar la frase **datos semiestructurados** para referirse a ellas. Y hay más, una gran porción de la Web está repetida (espejada (*mirrored*) o copiada) o es muy similar. Cerca del 30% de las páginas Web son duplicados cercanos [GAR/98]. La redundancia semántica es claramente superior.
- **Calidad de los datos**: la Web puede ser considerada un nuevo medio de publicación. Pero no existe, en la mayoría de los casos, un proceso editorial. Es por eso que los datos pueden ser falsos, inválidos (por ejemplo, por ser viejos), escritos pobremente o, típicamente, con errores de diferentes orígenes (ortográficos, gramaticales, de *OCR*, etc). Estudios preliminares demuestran que el número de palabras mal escritas puede ir de 1 en 200 para palabras comunes a 1 en 3 para apellidos extranjeros.
- **Datos heterogéneos**: aparte de tener que lidiar con múltiples tipos de medios y por lo tanto con múltiples formatos, tenemos también lenguajes diferentes y, lo que es peor, diversos alfabetos, algunos de ellos muy grandes (por caso, el chino o el Kanji japonés).

Muchos de estos problemas, como la variedad de tipos de datos y la calidad de datos pobre, no son resolubles simplemente mediante mejoras de *software*. En realidad, muchos de ellos no cambiarán (y no deberían, como es el caso de la diversidad idiomática) porque son problemas o propiedades inherentes a la naturaleza humana.

La segunda clase de problemas son aquéllos que el usuario debe enfrentar durante la interacción con el sistema de recuperación. Son básicamente dos problemas:

1) **Cómo especificar una consulta**: sin tomar en cuenta el contenido semántico de un documento, no es sencillo especificar con precisión una consulta, a menos que sea un requerimiento de información muy simple.
2) **Cómo interpretar la respuesta provista por el sistema**: aunque el usuario sea capaz de formular la consulta, la respuesta puede consistir en mil páginas Web ¿Cómo manejar una respuesta numerosa? ¿Cómo ordenar (*rank*) los documentos? ¿Cómo seleccionar los documentos que realmente interesan al usuario? Puede que haya documentos grandes. ¿Cómo recorrerlos eficientemente?

Entonces, vemos que el desafío general, sin desmedro de los problemas intrínsecos que trae aparejados la Web, es el de introducir una buena consulta en el sistema de búsqueda, y obtener una respuesta manejable y relevante. Es más, en la práctica se debe intentar alcanzar este último objetivo aunque las consultas estén pobremente formuladas.

Para completar la visión cuantitativa de la Web, podemos decir que de acuerdo a varios estudios coincidentes, la cantidad de hosts conectados a Internet creció de 80 millones en 6/2000 a 120 millones en 6/2001 [Netsizer] [ISC].

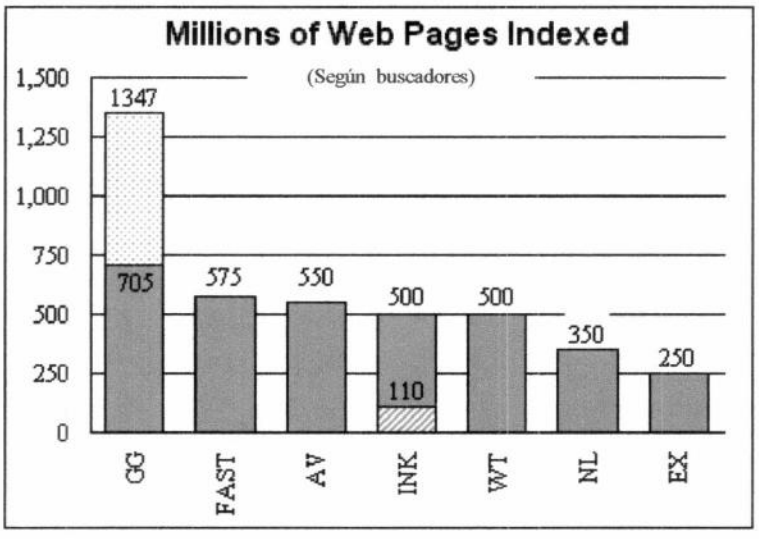
De estos hosts, en 2000 había 7,4 millones de sitios web [OCLC], lo que significaba en ese año que cada sitio tenía en promedio 135 páginas (1000 / 7,4). Sin embargo, desde 1997 [OCLC] se sabe que muy pocos sitios (1%) concentran una gran cantidad de páginas (50%).

3.2. Caracterizando los servicios de búsqueda

Continuaremos nuestro análisis cuantitativo y cualitativo, esta vez en lo concerniente a los motores de búsqueda (MB), los directorios y servicios de búsqueda en general.

Sabemos que el 80% de las páginas tienen menos de 10 enlaces apuntándoles, lo que es decir que se conforman islas de sitios, hecho que dificulta la recolección automática que se basa en seguir los enlaces.

Si nos detenemos a observar la cobertura de la Web indexable por parte de los principales MB, podemos considerar a [SEW] que nos muestra lo que cada MB decía indexar en 4/2001:



Referencias: GG = Google.com; FAST = Alltheweb.com; AV = AltaVista; INK = Inktomi; WT = WebTop.com; NL = NorthernLight.com; EX = Excite.com.

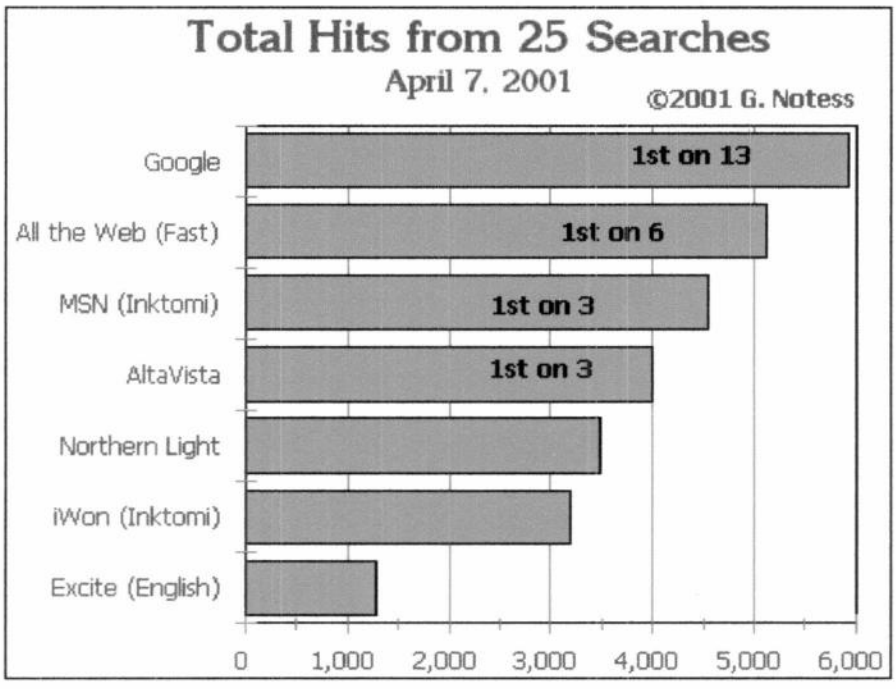
Recordemos que se calcula que 4000 millones de páginas conforman la Web indexable en 6/2001. Vemos que Google es el que más cobertura alcanza, ha indexado 705 millones de páginas (17% de la Web), pero por la manera en que utiliza los datos de los enlaces, puede devolver sitios que en realidad nunca ha visitado, llegando a 1347 millones (33%).

La separación en Inktomi diferencia los 110 millones de páginas de "lo mejor de la Web" y los 390 millones restantes para "resto de la web", que conforman su nuevo índice GEN3. Éste es usado por varios otros MB socios, entre ellos HotBot.com y NBCi.

El buscador con la colección de documentos más grande no es necesariamente el mejor MB. Si lo que se desea es encontrar información inusual u oscura, recurrir a un MB basado en el *crawling* que disponga de un índice enorme es, a menudo, la forma de proceder. El uso de la

automatización, en lugar de los editores humanos, significa que los rastreadores o *robots* tenderán a encontrar cosas que los editores nunca contemplarán. Cuanto más grande es el índice, mayor será la chance de encontrar información que satisfaga una búsqueda inusual.

Determinar qué buscador tiene el índice más abarcador no es una tarea simple. Todos los MB proveerán datos que dicen ser verdaderos, pero ello no significa que sean correctos. El sitio [SESH] realiza pruebas periódicas para validar las cantidades pretendidas por los MB.



En este estudio se realizan 25 búsquedas de una sola palabra, no muy populares (con alrededor de 250 páginas como resultado). Los valores de las abcisas son la cantidad total de resultados accesibles por su URL, en respuesta a las 25 consultas en cuestión.

Vemos que Google devolvió más páginas que los demás en 13 de las 25 consultas, All the Web en 6 y tanto MSN (usa Inktomi) como AltaVista en 3.

Si bien estos números nos dan una idea relativa de los tamaños de los índices, los valores absolutos fueron estimados también en [SESH], tomando como absolutos algunos valores devueltos por FAST y Northern Light.

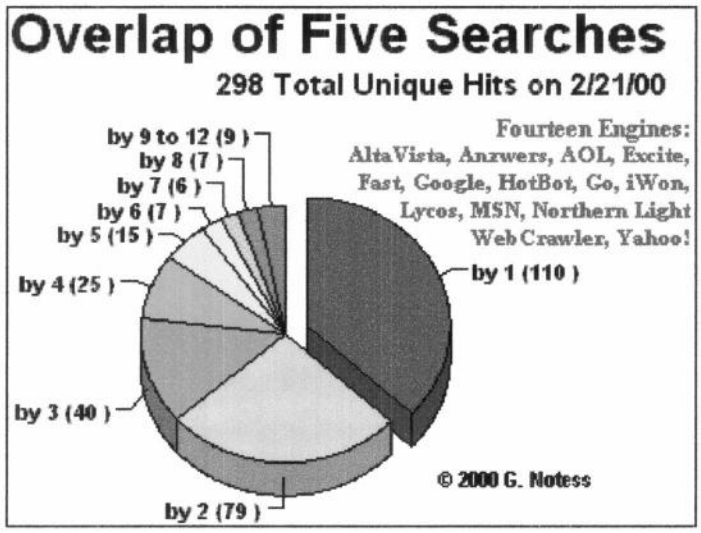
Motor de Búsqueda	Showdown Estimado (millones)	Dice tener (millones)
Google	625	700
Fast	539	607
MSN Search	480	500
AltaVista	423	500
Northern Light	368	322
iWon	337	500

Se observó en Google que menos del 1% de las respuestas a sus consultas son páginas no indexadas, por lo que no sería tan importante el "extra" que dice acceder. Así y todo, sigue primero en tamaño del índice o cobertura.

Se aclara que estos resultados no deben tomarse como definitivos ya que son estimaciones muy imprecisas, aunque basadas en resultados reales a consultas reales.

Consideremos ahora el análisis de la superposición o solapamiento (*overlap*) entre los índices de los MB.

En un estudio realizado en 2/2000, [SESH] analizó 5 consultas pequeñas en 14 buscadores para observar cuántas páginas aparecían repetidas en cuántos MB.



Del total de 795 páginas devueltas en todos los MB para las 5 consultas, en realidad 298 representan páginas únicas y el resto son repeticiones de las mismas. Es decir, el 62.5% de las páginas devueltas son repeticiones.

De esas 298 páginas, 110 fueron encontradas por sólo 1 MB, y 79 por sólo 2 MB.

Cada porción de la torta representa el número de aciertos (*hits*) o páginas encontrados por el número de MBs dado. Por ejemplo, la porción *by 1(110)* representa 110 aciertos únicos encontrados por uno (y sólo uno) de los buscadores.

Ni siquiera con 6 MBs que usan la base de datos de Inktomi, se verificó un solapamiento total, aunque en 3 de las 5 consultas fue casi completo.

Más del 76% de las páginas (229=110+79+40) sobre las 298 fueron devueltas por 3 MBs o menos.

Se puede concluir entonces que, a pesar del crecimiento sostenido de los índices, no se verifica una mayor superposición, quizás porque la Web crece más rápido que los índices.

Para culminar con esta descripción estadística del estado de la búsqueda de información en la Web, nos ocuparemos ahora de los directorios por temas y de algunas medidas de satisfacción del usuario.

Los directorios son, por lo general, guías de la Web compiladas por humanos, donde los sitios son ordenados por categorías. La tabla siguiente compara el tamaño de los directorios en varios servicios, además de datos clave para comprenderlos [SEW].

Servicio	Tipo	Editores	Categorías	Enlaces	Fecha
Open Directory	D	36,000	361,000	2.6 million	4/01
LookSmart	D	200	200,000	2 million	8/00
Yahoo	D	100+	n/a	1.5 to 1.8 million	8/00
NBCi (Snap)	D	30	80,000	1.5 million	12/00
AskJeeves	AS	150	n/a	128 million	3/01

Nota: D es directorio, AS es *answer service* (servicio de respuesta).

Primero observamos que el tamaño de los índices es muchísimo menor (de 5 a 350 veces menor) que el de los MB analizados previamente, de recolección automática. Por eso mismo, vemos que la cobertura de la Web que logran es mucho menor, siendo del 3,2% para AskJeeves y de 0,065% para el que lo sigue Open Directory.

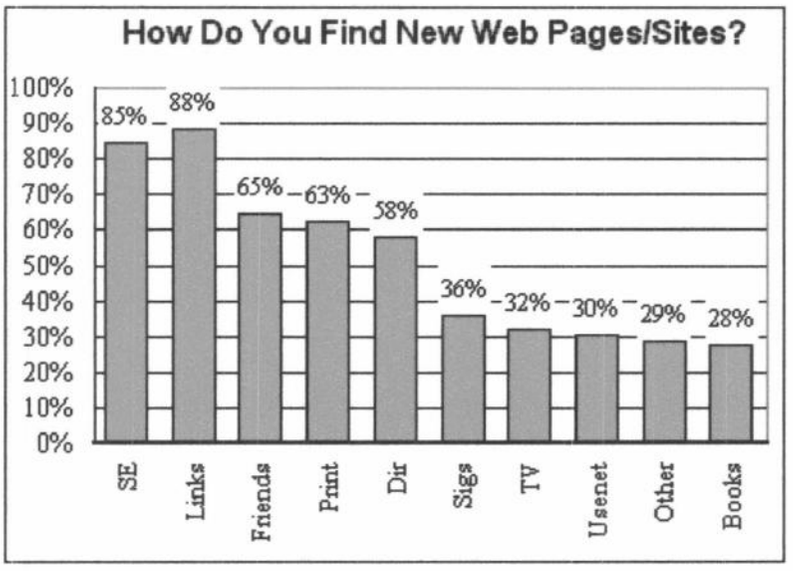
La vigencia e importancia de estos portales evidentemente no reside en la cobertura que logran sino en la calidad de los "pocos" sitios (relativamente) que ofrecen, y en el enorme avance que significa para el usuario poder navegar en categorías ordenadas en una jerarquía temática que se corresponde la mayoría de las veces a la categorización natural que haría el propio usuario.

Es por esto que las consultas de usuario sobre temas populares o conocidos seguramente encontrará recursos sumamente pertinentes a dichas necesidades.

De hecho, como se observa en varios estudios acerca de la popularidad de los servicios de búsqueda (MB y directorios) vemos que según [StatMarket] en 4/2000 Yahoo! iba a la cabeza con 53,4%, seguido de AltaVista con 18%. Similarmente, Yahoo! lidera en 12/2000 según [Nielsen] con el 48%, seguido por MSN con el 38,5% de los sitios visitados por los navegantes hogareños. Por último, [MedMet] en 2/2001 nos da también el porcentaje de usuarios que visitaron cada buscador en ese mes: Yahoo! ostenta el liderazgo con 63%, seguido por MSN con 56% y AOL con 41%.

En estos dos últimos estudios, los porcentajes combinados superan 100 porque un navegante puede haber visitado varios servicios. En el primero se utiliza HitBox, un servicio contador de visitantes, para medir cuánto tráfico proveniente de los buscadores se generó en los sitios, por lo que el porcentaje totaliza 100.

Finalmente, en un estudio llevado a cabo entre octubre y diciembre de 1998 por [GVU], vemos que ante la pregunta "¿Cómo encuentra usted a los nuevos sitios/páginas web?" se utilizan los MB en un 85%, sólo superado por el uso de enlaces con un 88%. El "boca a boca", o comentarios de conocidos sigue con el 65%, seguido de material impreso con 63% y por los directorios con el 58% (nuevamente se supera 100% porque un usuario puede usar varios métodos).



En cuanto a la satisfacción del usuario con sus búsquedas, se condujo una encuesta en verano/2000 (hemisferio sur) por [NPD], donde sólo el 2,6% dijo nunca encontrar lo que buscaba, y el 81% dijo encontrarlo siempre o casi siempre, en el nivel más alto desde 9/1997.

En el mismo estudio, se verificó que el 44,8% son búsquedas de varias palabras, y el 28,6% de una sola palabra, seguidos por el uso de opciones predefinidas y preguntas, un orden que se mantiene desde 9/1997 por lo menos.

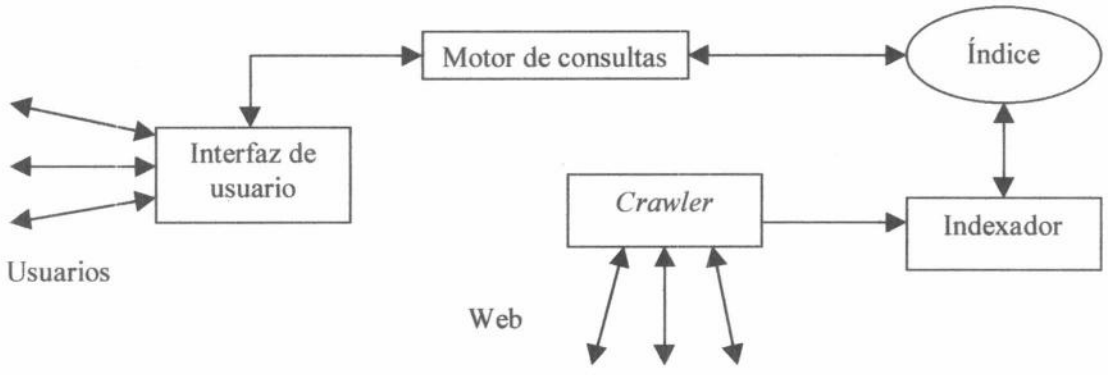
3.3. Arquitecturas de los motores de búsqueda

Podemos pensar en dos posibles arquitecturas para los sistemas de recuperación de información que modelan la Web como una base de datos de texto completo. Una diferencia sustancial entre los SRI estándares o tradicionales y los de la Web, es que en ésta todas las consultas deben realizarse sin acceder al texto, para mejorar la eficiencia en tiempo de ejecución. Ello no significa que sólo estén disponibles los índices, ya que por lo menos Google almacena las páginas que indexa (*cached pages*). Acceder a la copia local de la Web recolectada o a las páginas remotas a través de la red en tiempo de consulta sería demasiado lento. Esta diferencia tiene un impacto sobre los algoritmos de indexación y búsqueda, así como en los lenguajes de consulta disponibles.

3.3.1. Arquitectura centralizada

La mayoría de los motores de búsqueda utilizan una arquitectura conocida como de rastreador-indexador (*crawler-indexer*). Los rastreadores son programas o agentes de *software* que recorren la Web enviando páginas nuevas o actualizadas a un servidor principal donde son indexadas. A los rastreadores (*crawlers*) se los conoce también como *robots*, *spiders*, *wanderers*, y algunos términos más. A pesar de su nombre, un *crawler* no se mueve realmente y se ejecuta en las máquinas remotas, sino que corre sobre un sistema local y envía pedidos a servidores Web remotos.

El índice es utilizado de manera centralizada para responder las consultas enviadas desde diferentes lugares de la Web. La figura siguiente muestra la arquitectura de programa de un motor de búsqueda basado en la arquitectura de AltaVista.



Tiene dos partes: una que trata con los usuarios, consistente en la interfaz de usuario y el motor de consultas (*query engine*); y la otra que consta del *crawler* y los módulos indexadores. En 1998, el sistema de AltaVista en su conjunto corría sobre 20 computadoras multiprocesador, cada una disponiendo de más de 130 Gb de RAM y más de 500 Gb de espacio en disco. Sólo el motor de consultas usaba más del 75% de los recursos.

El problema fundamental que enfrenta esta arquitectura es la recolección de datos, a raíz de la naturaleza altamente dinámica de la Web, los enlaces de comunicación saturados y la carga pesada en los servidores Web. Otro problema crucial es el volumen de los datos. En los hechos, la arquitectura de recolector-indexador puede no ser capaz de lidiar con el crecimiento de la Web en un futuro cercano. De particular importancia es un buen balance de carga entre las diferentes actividades del motor de búsqueda, internamente (respondiendo consultas e indexando) y externamente (recolectando).

3.3.2. Arquitectura distribuida

Hay diversas variantes a la arquitectura recolector-indexador. Entre ellas, la más importante es la representada por el sistema Harvest (cosechar), que aunque ha quedado obsoleto, nos sirve para ejemplificar este tipo de arquitectura.

Harvest usa una arquitectura distribuida para reunir y distribuir los datos, lo cual lo hace más eficiente que la arquitectura de recolección. La principal desventaja de este enfoque es que Harvest requiere la coordinación de varios servidores Web.

La aproximación distribuida de Harvest aborda varios de los problemas de la arquitectura recolector-indexador, como por ejemplo:

1) Los servidores Web incrementan su carga al recibir pedidos de los diferentes *crawlers*.
2) El tráfico en la Web crece porque los *crawlers* obtienen objetos completos, para luego descartar la mayor parte del contenido de los mismos al indexarlos.
3) La información es recolectada independientemente por cada *crawler*, sin coordinación entre todos los motores de búsqueda.

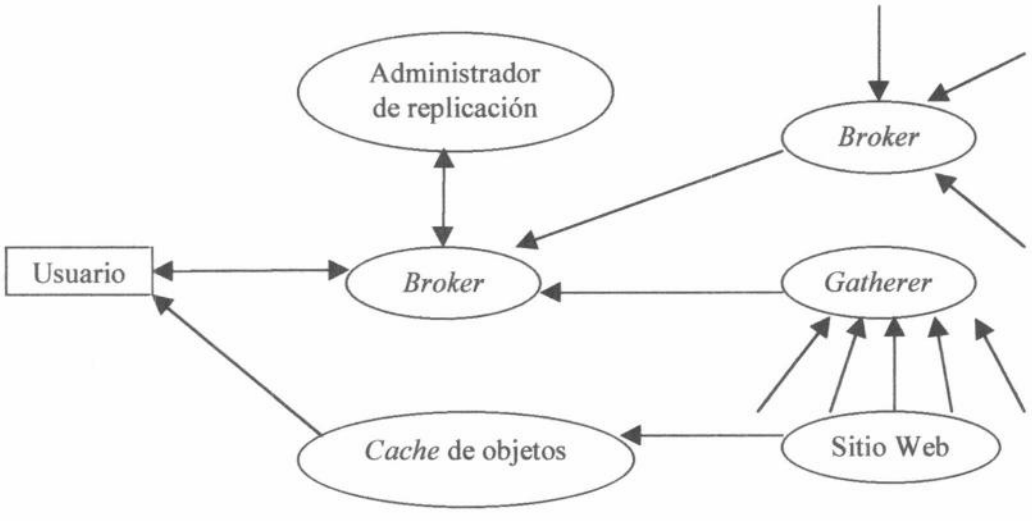
Para resolver estos problemas, Harvest introduce dos elementos principales: *gatherers* (acopiadores) y *brokers*. El *gatherer* recolecta y extrae información de indexación de uno o más servidores Web, periódicamente (los tiempos de cosecha, como sugiere el nombre del sistema).

El *broker* provee el mecanismo de indexación y la interfaz de consultas a los datos acopiados. Los *brokers* obtienen información de uno o más *gatherers* o de otros *brokers*, actualizando sus propios índices de manera incremental.

Dependiendo de la configuración, se pueden lograr diversas mejoras en cuanto a la carga en el servidor y al tráfico en la red.

Por ejemplo, un *gatherer* puede correr en un servidor Web, sin generar tráfico externo a dicho servidor. También puede enviar información a varios *brokers*, evitando la repetición del trabajo. Los *brokers*, a su vez, pueden filtrar la información y enviarla a otros *brokers*.

La idea de este diseño es la de compartir y distribuir la información y el trabajo de una manera flexible y genérica. Vemos en la figura un ejemplo de la arquitectura Harvest.



Uno de los objetivos de Harvest es construir *brokers* para temas específicos, focalizando el contenido del índice y evitando muchos de los problemas de vocabulario y de escala que presentan los índices genéricos.

Se incluye en el sistema a un *broker* distinguido que permite que los demás *brokers* y *gatherers* registren información relativa al conjunto de *gatherers* y *brokers* que conocen. Esto es útil para buscar un *broker* o *gatherer* apropiado al construir un sistema nuevo.

La arquitectura Harvest también dispone de replicadores y *caches* (memorias temporarias) de objetos. Los replicadores se usan para replicar servidores, ampliando la escalabilidad de la base de usuarios. Por ejemplo, el *broker* de registración puede estar replicado en diferentes regiones geográficas para permitir un acceso más rápido. Finalmente, la *cache* de objetos reduce la carga de la red y del servidor, así como la latencia de respuesta al acceder a las páginas Web.

Al ser el sistema Harvest de dominio público, se realizaron cientos de aplicaciones en la Web, siendo ejemplos importantes la CIA o la NASA.

3.4. Interfaz de usuario

Hay dos aspectos importantes en la interfaz de usuario de los motores de búsqueda: la interfaz de consultas y la interfaz de respuesta.

La interfaz de consultas básica es una caja de texto en donde pueden tipearse una o más palabras. Aunque el usuario pueda suponer que una secuencia de palabras dada representa la

misma consulta en todos los MB, no es así. Por ejemplo, en AltaVista una secuencia de palabras es una referencia a la unión de todas las páginas Web que tengan al menos una de las palabras, mientras que en HotBot es una referencia a las páginas Web que contengan a todas las palabras.

Otro problema es que la visión lógica del texto no es conocida, puesto que algunos MB usan *stopwords*, *stemming*, o no son sensibles a la capitalización de las letras (*case sensitive*). Ya hemos hablado oportunamente de estas operaciones.

Todos los MB proveen también una interfaz para consultas complejas así como un lenguaje de comandos que incluyen los operadores Booleanos y otras facilidades, como búsqueda por frases, búsqueda por proximidad, y el uso de comodines.

Adicionalmente, se ofrecen varias funciones de filtrado. Los resultados pueden ser filtrados mediante palabras que deben o no estar presentes en la respuesta o en algún campo particular, como la URL o el título; por lenguaje, región geográfica, dominio de Internet, rango de fechas o por la inclusión de tipos de datos específicos como audio o imágenes.

La respuesta consiste generalmente en una lista de las 10 páginas Web más relevantes a la consulta. Cada entrada de esta lista incluye información acerca del documento al que representa. Típicamente, esta información se puede componer de la URL, tamaño, fecha en que la página fue indexada, y unas líneas describiendo el contenido de la misma (título más las primeras líneas o encabezados u oraciones seleccionadas).

Algunos MB permiten al usuario cambiar la cantidad de páginas devueltas en la lista y la cantidad de información por página, pero en la mayoría de los casos esto está fijo o limitado a unas pocas opciones. El orden de la lista es típicamente por relevancia, pero también puede estar disponible el ordenamiento por URL o fecha en algunos MB.

La mayor parte de los MB tienen una opción para encontrar documentos similares a cada página Web de la respuesta. En algunos casos, se agrupan las páginas del mismo dominio y se dispone de una opción para ver todas las páginas del dominio que no se muestran en el resultado, y que son relevantes a la consulta. Google inclusive filtra páginas casi duplicadas en contenido a las que muestra como respuesta, y ofrece la posibilidad de incluir o no dichos duplicados.

El usuario puede asimismo refinar la consulta inicial, construyendo consultas más complejas basadas en las respuestas anteriores.

Las páginas devueltas por un motor de búsqueda en respuesta a una consulta de usuario están ordenadas según el puntaje asignado por el algoritmo de *ranking*, usualmente basado en estadísticas relacionadas con los términos de la consulta. En algunos casos esto puede no tener sentido, porque la relevancia no está totalmente correlacionada con las estadísticas concernientes a las apariciones de términos en la colección.

Algunos MB están teniendo en cuenta además los términos incluidos en los tags *meta* o *title* del código HTML, o la popularidad de una página Web para mejorar su puntaje. Por último, se incluyen cada vez más las técnicas de análisis de enlaces en el grafo de hipervínculos de la Web, pero dedicaremos una sección completa a detallar los algoritmos y las ideas que subyacen o motivan su aplicación en los MB actuales.

3.5. Recolección de las páginas Web

En esta sección, discutimos cómo se puede recorrer (*crawl*) la Web, puesto que existen diversas técnicas. La más simple consiste en comenzar con un conjunto de URLs y desde allí

extraer otras URLs que son recorridas recursivamente siguiendo el algoritmo de primero en profundidad (*depth-first*) o primero en amplitud (*breadth-first*).

Una variante es comenzar con un conjunto de URLs populares, porque podemos suponer que tienen información que se requerirá con frecuencia.

Ambos casos sirven para un *crawler*, pero el proceso se complica al tener que coordinar varios *crawlers* para que no visiten la misma página más de una vez.

Otra técnica es particionar la Web usando códigos de país o nombres de dominio de Internet, y asignar un grupo de *robots* a cada partición, y explorar cada partición de manera exhaustiva.

Al considerar cómo se suele recorrer a la Web, podemos pensar al índice de un MB como análogo a un cielo lleno de estrellas. Lo que observamos en él, no ha existido jamás, puesto que la luz ha recorrido diferentes distancias para llegar a nuestro ojo. De manera similar, las páginas Web presentes en un índice fueron exploradas también en diferentes fechas, y pueden no existir más.

¿Cuánto hace que están las páginas en el índice? Normalmente pueden tener desde un día hasta algunos meses, por lo que muchos MB muestran, en la respuesta, la fecha en que la página fue indexada. El porcentaje de enlaces inválidos encontrados en los MB varía del 2% al 9%.

Las páginas enviadas al MB por los usuarios son recolectadas en un período de unos pocos días a algunos meses. Partiendo de ellas, algunos MB recopilan todo el sitio Web, mientras otros seleccionan una muestra de las páginas o sólo hasta cierta profundidad del sitio. Ciertos MB aprenden la frecuencia de actualización de una página y la visitan de acuerdo a ello. También pueden recabar con mayor frecuencia las páginas populares o las más relevantes.

En líneas generales, los *crawlers* más rápidos son capaces de procesar hasta 10 millones de páginas por día.

El orden en que se visitan las páginas es importante. Como ya se ha mencionado, los enlaces de las páginas Web pueden recorrerse en profundidad o amplitud. Si la política es esta última, miraremos en primer lugar las páginas apuntadas por la página en curso, y así siguiendo. La cobertura será amplia pero superficial y un servidor Web puede ser bombardeado con muchos pedidos rápidos.

En el caso de profundidad, seguimos el primer enlace de una página y hacemos lo mismo en la siguiente hasta no poder ir más profundo, retornando recursivamente. El recorrido esta vez es angosto y profundo.

Algunas investigaciones han demostrado que los esquemas de ordenamiento en que se visitan las mejores páginas primero pueden hacer la diferencia (usando la medida PageRank explicada luego).

Debido al hecho descripto de que los *robots* pueden sobrecargar un servidor mediante pedidos rápidos y a la vez utilizar un ancho de banda significativo de Internet, se desarrolló una serie de pautas (protocolo) para limitar el comportamiento de los *robots*.

Puede haber problemas con páginas que usan marcos y con mapas de imágenes. Las páginas generadas dinámicamente y las protegidas con claves no son extraíbles ni, por lo tanto, indexables.

3.6. Índices

La gran mayoría de los índices utilizan variantes del **archivo invertido**. Brevemente, un archivo invertido es una lista de palabras ordenadas (vocabulario), cada una de ellas asociadas a un conjunto de punteros a las páginas en donde ocurren. Algunos MB usan la eliminación de *stopwords* para reducir el tamaño del índice. Es importante recordar que lo que se indexa en definitiva es una visión lógica del texto.

Las operaciones de normalización del texto pueden incluir la remoción de signos de puntuación, de múltiples espacios a uno solo entre palabras, pasaje de mayúsculas a minúsculas, etc. Para dar al usuario una idea acerca de cada documento recuperado, el índice se complementa con una descripción corta de cada página Web (fecha de creación, tamaño, título, primeras líneas o encabezados).

Asumiendo que son necesarios 500 bytes para almacenar la URL y la descripción de cada página Web, necesitamos 350 GB para describir 700 millones de páginas. Como el usuario recibe sólo un subconjunto de la respuesta completa a cada consulta, el MB mantiene usualmente la respuesta completa en memoria, para no tener que recomputarla en el caso que el usuario requiera más documentos.

Las técnicas de indexación modernas pueden reducir el tamaño de un archivo invertido a cerca del 30% del tamaño del texto, a menos si se usan *stopwords*. Para 700 millones de páginas, esto implicaría 1050 GB de espacio en disco. Si se usan técnicas de compresión, el tamaño del índice puede reducirse al 10% del texto.

Una consulta es procesada mediante una búsqueda binaria en la lista ordenada de palabras del archivo invertido. Si buscamos por múltiples palabras, deben combinarse los resultados para generar la respuesta final. Este paso será eficiente si las palabras no son demasiado frecuentes (en la colección). Otra posibilidad es computar la respuesta completa a medida que el usuario pide más páginas Web, usando un esquema perezoso (*lazy*) de evaluación.

Los archivos invertidos pueden apuntar a las ocurrencias reales de la palabra en un documento (inversión total). Sin embargo, esto es muy costoso en espacio para la Web, porque cada puntero debe especificar una página y una posición (en cantidad de bytes o de palabras) dentro de la página.

Por otro lado, si tenemos las posiciones de las palabras en la página, podemos responder búsqueda por frases o proximidad encontrando palabras que se encuentran cerca una de otra en la página. Actualmente, algunos MB proveen búsquedas por frases, pero no se conoce la implementación real.

Encontrar palabras que comienzan con un prefijo dado necesita de dos búsquedas binarias sobre la lista ordenada de palabras. Las búsquedas más complejas, con comodines o en general cualquier expresión regular sobre una palabra, puede realizarse haciendo un recorrido secuencial sobre el vocabulario. Aunque parezca lento, se puede tardar medio minuto en recorrer el vocabulario de 630 MB correspondiente a 1 TB de texto.

La posibilidad de apuntar a páginas o a posiciones de palabras dentro de las mismas nos da una indicación de la **granularidad** del índice. Si apuntamos a bloque lógicos en lugar de a las páginas en sí, podemos bajar la densidad del índice. De esta manera, reducimos la varianza de los diferentes tamaños de los documentos, tomando bloques del mismo tamaño. Esto no sólo reduce el tamaño de los punteros, sino también la cantidad de los mismos ya que las palabras tienen localidad de referencia (i.e. todas las ocurrencias de una palabra no frecuente tenderá a

agruparse en un mismo bloque). Esta idea fue usada en el sistema Harvest anteriormente mencionado.

3.7. Algoritmos de ranking

La mayoría de los motores de búsqueda usan variantes de los modelos Booleano o vectorial para realizar el *ranking*. Como ocurría con la búsqueda, el *ranking* debe realizarse sin acceder al texto, sólo al índice. No existe demasiada información pública acerca de los algoritmos de *ranking* específicos que usan los MB actuales. Sumado a esto, se torna difícil comparar con justicia diferentes MB a raíz de sus diferencias y las modificaciones continuas de que son objeto. Más importante aún, es casi imposible medir el *recall*, ya que el número de páginas relevantes puede ser inmenso para consultas simples.

Alguno de los algoritmos de *rankings* más recientes también usan la información de los hiperenlaces. Esta es una diferencia esencial entre la Web y las bases de datos normalmente utilizadas en Recuperación de la Información.

La cantidad de hiperenlaces que apuntan a una página nos puede dar una medida de su popularidad y también, aunque no necesariamente, de su calidad. Asimismo, si dos páginas tienen muchos enlaces en común o son referenciadas por una misma página indica, con frecuencia, una relación entre esas páginas. A continuación presentamos ejemplos de técnicas de *ranking* que explotan estas cuestiones. En algunos casos dependen de la consulta y, en otros, se utiliza una medida que es función de las páginas e independiente de cada consulta específica.

3.7.1. Ordenando la Web con PageRank

Larry Page y Sergey Brin propusieron el algoritmo PageRank en [BRI/98] y [PAG/98] y lo aplicaron en el buscador Google.

Como ya fue señalado, la Web genera importantes desafíos a la RI. Por tamaño y heterogeneidad, por dinamismo y volatilidad, por su crecimiento explosivo. También impone a los MB la necesidad de lidiar con usuarios no experimentados, sus consultas cortas o ambiguas, y también con páginas diseñadas para manipular las funciones de *ranking* de los MB.

Sin embargo, a diferencia de las colecciones de documentos "planos", la Web es un medio hipertextual y nos brinda considerable información auxiliar por encima del texto de las páginas Web, como por ejemplo su estructura de enlaces y el texto que etiqueta dichos enlaces (conocido como texto de anclaje o *anchor text*).

PageRank aprovecha entonces la mencionada estructura de enlaces para construir un *ranking* global que indica la importancia de cualquier página Web. De esta manera, ayuda al MB y a los usuarios a darle un sentido a la enorme heterogeneidad de la Web.

Existen grandes diferencias entre las citas en una publicación académica y las citas en la Web. A diferencia de las citas académicas, que se realizan bajo un estricto control, las páginas Web proliferan sin control de calidad ni costos de publicación. Con un programa simple, es posible generar fácilmente enormes cantidades de páginas, inflando artificialmente la cantidad de citas. Las estrategias de atracción de atención por parte de las empresas evolucionan en respuesta a los algoritmos de los MB. Es por esta razón que cualquier algoritmo de evaluación que cuente aspectos replicables de las páginas Web es susceptible de ser manipulado.

Otra característica fundamental a tener en cuenta al analizar la utilidad de PageRank tiene que ver con que la calidad promedio de una página Web experimentada por un usuario es

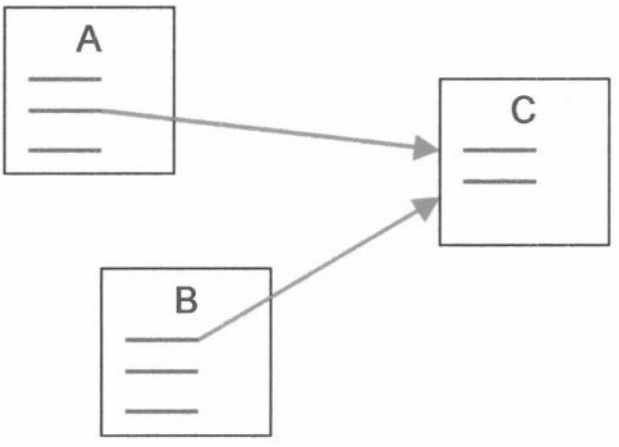
superior a la calidad promedio de cualquier página Web. Esto se da porque la facilidad para crear y publicar páginas Web redunda en una fracción enorme de páginas de baja calidad, que los usuarios no están dispuestos a leer.

Si bien existen múltiples dimensiones para diferenciar las páginas Web, PageRank está pensado para dar una aproximación a la **importancia relativa global** de las páginas Web. Para ello utiliza la información presente en el grafo de la Web.

La primera tentación es aplicar todas las técnicas de análisis de citas que fueron utilizadas para entornos académicos a la Web. Una primera aproximación sería pensar cada enlace como si fuera una cita académica. Entonces, una página como www.yahoo.com tendrá infinidad de retroenlaces (*backlinks*) o citas apuntándole.

El hecho de que una página como Yahoo! tenga tantos retroenlaces implica generalmente que es una página realmente importante. De hecho, muchos buscadores han intentado usar la cantidad de retroenlaces en su intento por sesgar sus bases de datos en favor de páginas de mayor calidad. Sin embargo, como esbozamos antes, contar los retroenlaces en la Web es más problemático que contarlos en las bases de datos de citas académicas.

Con las estimaciones actuales, el grafo de la Web extraíble e indexable consta de 4000 millones de nodos (páginas) y 40.000 millones de ejes (enlaces), a razón de 10 enlaces en promedio por página. Toda página tiene ciertos enlaces entrantes o salientes. En la figura vemos que A y B son retroenlaces de C.



Al indexar una página, no se puede saber si se encontraron todos los retroenlaces de esa página particular pero sí podemos saber todos sus enlaces salientes en ese momento.

Las páginas Web varían mucho en cuanto a la cantidad de retroenlaces que poseen. La página de un portal popular tendrá muchos más retroenlaces que la mayoría de las páginas web que poseen más bien pocos retroenlaces, si es que tienen. PageRank provee un método más sofisticado de conteo de los retroenlaces.

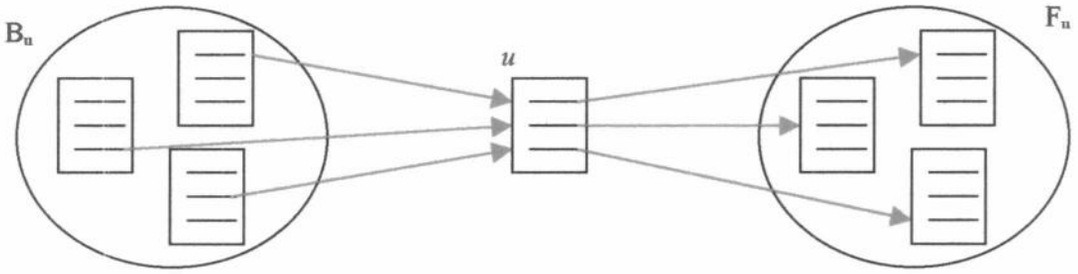
La razón por la cual se recurre a este esquema más interesante es que en muchos casos la cantidad de retroenlaces simple y llana no se corresponde con nuestra noción de importancia. Por ejemplo, si una página web tiene un enlace desde la página central de Yahoo!, puede ser un único enlace, pero muy importante. Dicha página debería ser considerada más relevante que otras que tienen muchos retroenlaces en sitios sin importancia.

PageRank es un intento por ver cuánto nos podemos aproximar a la noción de "importancia" partiendo solamente de la estructura de hiperenlaces.

Basándonos en la discusión precedente, se puede dar la siguiente descripción intuitiva de PageRank: una página tiene alto puntaje (*rank*) si la suma de los puntajes de sus retroenlaces es alta. Esto cubre los casos en que una página tiene muchos retroenlaces y en que tiene pocos retroenlaces pero con puntajes altos.

Veamos ahora una descripción matemática de PageRank, junto con una justificación intuitiva de las fórmulas intervinientes.

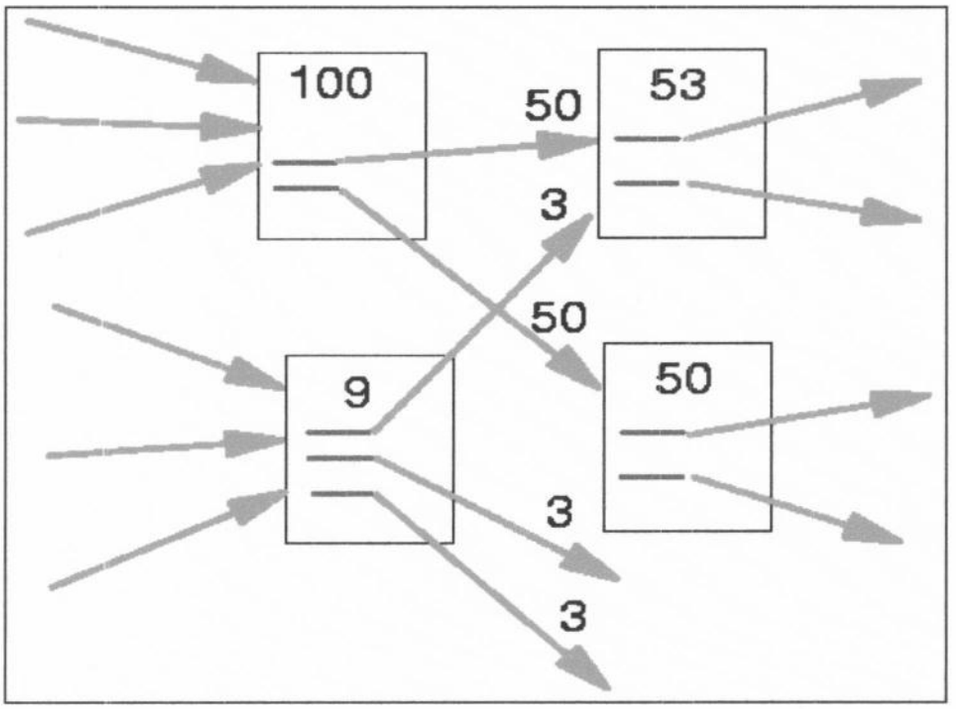
Sea u una página web. Entonces tomemos F_u como el conjunto de páginas a las que apunta u y B_u como el conjunto de páginas que apuntan a u. Sea $N_u = |F_u|$ la cantidad de enlaces desde u, y sea c un factor usado para normalizar, de manera que el puntaje total de todas las páginas sea constante.



Comencemos por definir una función de *ranking* R, que es una versión un poco simplificada de PageRank:

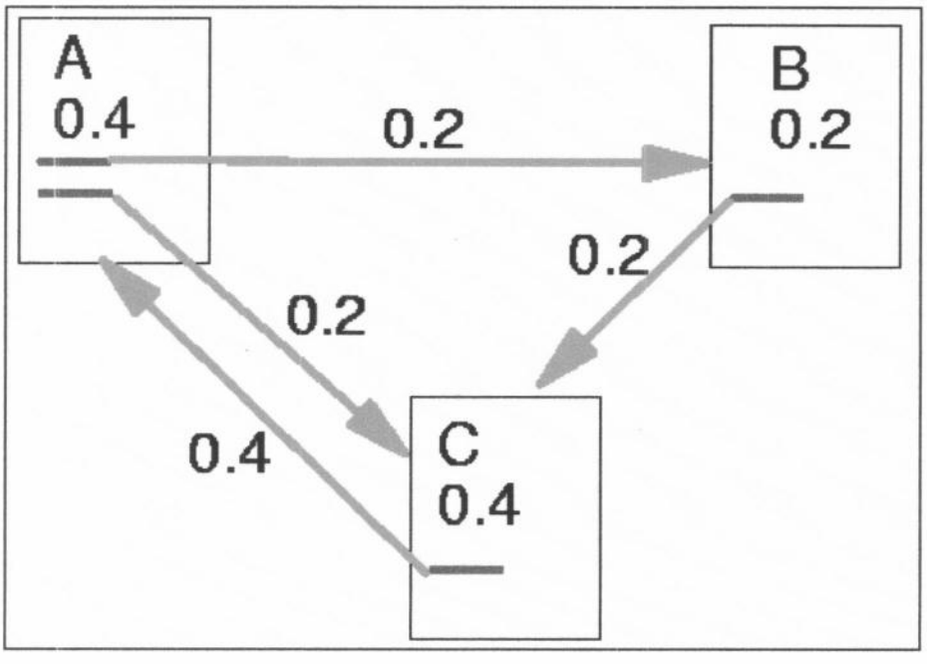
$$R(u) = c \sum_{v \in B_u} \frac{R(v)}{N_v}$$

Notemos que el puntaje de una página es dividido de manera equitativa entre sus enlaces salientes para contribuir al puntaje de las páginas a las que apunta.



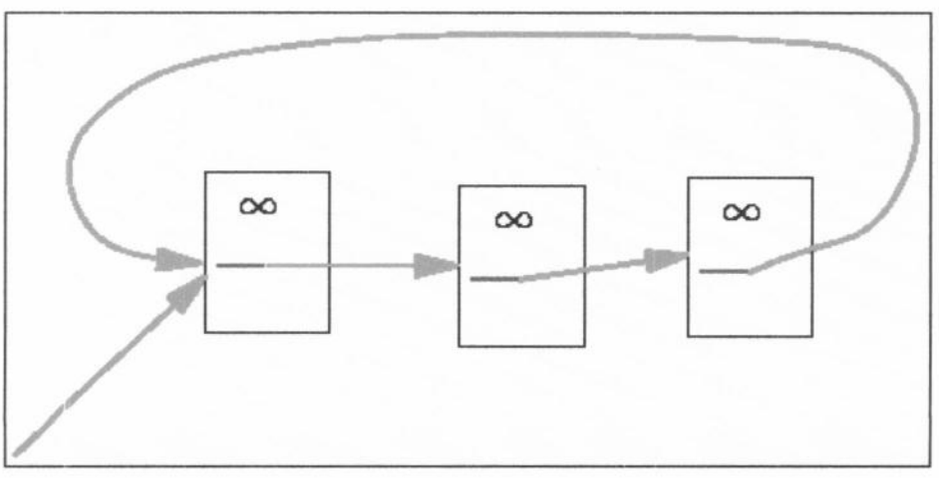
La ecuación es recursiva pero puede ser computada comenzando por cualquier conjunto de puntajes e iterando la computación hasta que converja. La figura anterior demuestra la propagación ponderada del puntaje de una página a la otra.

En la siguiente figura observamos una solución consistente y estable para un conjunto de páginas.



Visto desde otro ángulo, supongamos que A es la matriz cuadrada donde sus filas y columnas corresponden a las páginas web. Sea $A_{u,v} = 1 / N_u$ si existe un eje de u a v y $A_{u,v} = 0$ si no lo hay. Si tratamos a R como un vector sobre las páginas web, entonces tenemos que $R = cAR$. Entonces R es un autovector de A con autovalor c. De hecho, lo que se busca es el autovector dominante de A. Puede computarse mediante la aplicación repetida de A tomando cualquier vector inicial no degenerado.

Hay un pequeño problema con esta función simplificada de puntajes. Consideremos dos o más páginas que se apuntan entre sí, pero no apuntan a ninguna otra página. Y también supongamos que hay una página que apunta a una de ellas. Entonces, durante la iteración, este bucle acumulará puntaje pero no lo distribuirá nunca (pues no hay enlaces salientes). El círculo o bucle constituye una trampa que llamaremos acumulador de puntaje (*rank sink*), graficado a continuación.



Para sortear este obstáculo, se introduce un origen de puntaje. Sea E(u) algún vector sobre las páginas web que corresponde a un origen de puntaje. Entonces el PageRank de un conjunto de páginas web es una asignación R' a dichas páginas que satisface

$$R'(u) = c \sum_{v \in B_u} \frac{R'(v)}{N_v} + cE(u)$$

de forma tal que c está maximizado y $||R'||_1 = 1$ ($||R'||_1$ es la norma 1 de R').

Donde E(u) es un vector sobre las páginas web que corresponde con un origen de puntaje. Notar que si E es todo positivo, c debe ser reducido para balancear la ecuación. Por ello, esta técnica implica el uso de un factor de decaimiento.

En notación matricial, se tiene que R' = c (AR' + E). Como $||R'||_1 = 1$, podemos reescribir la ecuación como R' = c (A + E x **1**) R' donde **1** es el vector que contiene todos unos. Entonces R' es un autovector de (A + E x **1**).

La definición que acabamos de ver tiene otra justificación intuitiva en los recorridos aleatorios (*random walks*) sobre grafos.

La versión simplificada corresponde a la distribución de probabilidad de parada de un recorrido aleatorio sobre un grafo de la web. Intuitivamente, esto puede ser pensado como el modelado del comportamiento de un "navegante aleatorio". Este "navegante aleatorio" simplemente pincha sobre sucesivos hiperenlaces de una manera azarosa. Sin embargo, si un navegante real llega a atorarse en un pequeño bucle de páginas, es difícil pensar que se quede allí para siempre. Por el contrario, el navegante saltará a alguna otra página.

De hecho en [BRI/98] se da una fórmula para PageRank que da cuenta de esta interpretación probabilística.

$$R'(u) = d \sum_{v \in B_u} \frac{R'(v)}{N_v} + (1-d).E(u)$$

En esta nueva fórmula, *d* es un factor de ponderación que puede verse como la probabilidad de que un navegador aleatorio elija un enlace de la página en que se encuentra. Por el contrario, *(1 - d)* es la probabilidad de que elija continuar su caminata por cualquier página web, según la distribución de pesos asignada en E. Google usa $d = 0.85$.

Puede verse al factor adicional E como modelador de este comportamiento: el navegante se aburre periódicamente y salta a una página al azar elegida según la distribución de E.

Hasta aquí se pensó a E como un parámetro definido por el usuario. En la mayoría de los casos se toma E uniforme sobre todas las páginas con un valor fijo.

Los beneficios de PageRank se hacen evidentes con consultas subespecificadas o de baja calidad semántica. Por ejemplo, una consulta acerca de "universidad" retornará cualquier cantidad de páginas que nombren "universidad" en un motor de búsqueda convencional. Con el uso de PageRank, las páginas centrales de las universidades más importantes serán listadas primero.

Uno de los objetivos en el diseño de PageRank es manejar correctamente los **intereses comunes** para una consulta. Por dar un ejemplo, a un usuario que busca "flores" puede devolvérsele (usando PageRank) un sitio comercial comúnmente usado que contiene poca información a excepción de cómo comprar flores.

La tarea de encontrar sitios con gran cantidad de información sobre "flores" es muy diferente a la tarea que consiste en encontrar el sitio más comúnmente utilizado que hable de flores. Existe un sistema que aborda la primera tarea [MAR/97], que intenta encontrar sitios que discuten un tema en detalle y que vemos con más detenimiento en otra sección.

Claramente, ambas tareas son importantes, y un MB de propósito general debe retornar automáticamente resultados que conformen las necesidades de estas dos tareas.

El hecho de que un sitio sea de interés común en el grafo de la web no está contenido en el código HTML de la página. Aun cuando hubiera formas de definir buena meta-información de este tipo dentro de una página, sería un asunto problemático dado que no se puede confiar en el autor de una página para este tipo de evaluaciones. Muchos autores de páginas web dirían simplemente que sus páginas son las mejores y más usadas de la web.

Consideremos cómo PageRank ayuda a representar los diversos escenarios en que surgen intereses comunes. Además de las páginas de mucho uso, como el sitio de venta de flores, PageRank puede representar una noción cooperativa de autoridad o confianza. Por ejemplo, un usuario podría preferir una noticia por el simple hecho de que es apuntada directamente por la página central de un diario importante. Es claro que tal noticia recibirá un alto PageRank por ser mencionada y apuntada por una página importante. Esto parece capturar un tipo de confianza cooperativa, dado que si una página es mencionada por una fuente autorizada o de confianza, es más probable que sea, ella misma, autorizada y de confianza. La calidad y la importancia parecen responder a este tipo de definiciones circulares.

PageRank puede utilizarse para **personalizar las búsquedas** en la web. Se usa para ello al vector E sobre las páginas web, tomado como origen de puntaje para calcular PageRank para evitar, como vimos, el problema de los *rank sinks*.

Si en el vector E, en lugar de asignar un peso uniforme a todas las páginas web sólo por existir, asignamos más peso a determinadas páginas de acuerdo a algún perfil de usuario o interés temático, todas las páginas web sesgarán sus PageRanks en favor de estos intereses.

El vector E ya no modela el comportamiento de un navegante aleatorio, sino el comportamiento de la persona que realiza la búsqueda, siempre según el tipo de entrada que se considere como contexto o perfil del usuario. Con este fin, pueden usarse por ejemplo los *bookmarks* o la página web inicial de un usuario.

Se han reportado otros usos a PageRank que están más allá de este trabajo, y son detallados en [PAG/98].

3.7.2. La síntesis de Google: texto, estructura HTML e hiperenlaces

En [BRI/98], los autores de PageRank describieron Google, en lo que fue la primera vez que se publicó la implementación detallada de un motor de búsqueda de la Web de gran escala.

Veremos el algoritmo de asignación de relevancia a páginas, sin detenernos en el detalle de la arquitectura total del sistema, como por ejemplo los módulos de indexación o *crawling*.

Google hace uso de la estructura de hiperenlaces de la Web para calcular una medida de calidad de una página Web. Este *ranking* se denomina PageRank, descripto en detalle en la sección anterior. Recordemos que la medida de PageRank para una página web determinada es independiente de cualquier consulta que realice un usuario en el MB.

Google también considera el texto asociado a los enlaces para mejorar las respuestas a las consultas. A este tipo de texto se lo denomina de anclaje o *anchor*.

La mayoría de los buscadores asocia el texto de los enlaces con la página en la que aparecen; pero en el caso de Google, también se asocia el texto con la página a la que se apunta en ese enlace. Esto tiene varias ventajas. Primero, los anclajes proveen a menudo descripciones más precisas de las páginas web que las páginas en sí mismas. Segundo, puede haber texto de anclaje asociado a documentos que no fueron o no pueden ser indexados, como imágenes, programas y bases de datos. Esto permite retornar páginas web que no han sido visitadas por los *crawlers*.

Nótese que las páginas que no fueron visitadas pueden causar diversos problemas, porque no se chequea su validez antes de retornarla al usuario. En el caso extremo, el MB puede devolver páginas web que nunca existieron realmente, pero que tienen enlaces apuntándoles. A pesar de ello, Google puede ordenar los resultados para que este problema se atenúe o no ocurra.

La idea de propagar el texto de anclaje a la página apuntada fue implementada primero en el buscador World Wide Web Worm, por su utilidad para buscar en información no textual y porque permite expandir la cobertura de documentos con menor cantidad de documentos recorridos por los *robots*.

En Google, se usa el texto de los enlaces más que nada para mejorar la calidad de los resultados a las consultas.

Además del uso de PageRank y el anclaje, se usa información de posición de cada aparición de un término en un documento, para poder hacer uso intensivo de la proximidad entre los términos de la consulta. Otro aspecto que se agrega a esta batería de recursos, es el uso de los detalles de presentación como por ejemplo el tamaño de la fuente con que aparecen las palabras. Las apariciones más grandes o en negrita son más pesadas que las de texto simple.

Vemos entonces que se utiliza todo el potencial de la Web para aplicarlo en el algoritmo de asignación de puntajes, incluyendo los detalles de la estructura HTML de las páginas.

Aunque el código HTML de todas las páginas web que han sido visitadas por los recolectores se encuentra comprimido en el índice de documentos, las búsquedas no se realizan sobre el texto completo de las mismas, sino sobre las diferentes estructuras del índice que proveen la visión lógica de los documentos.

En su índice, Google convierte cada documento en un conjunto de ocurrencias de palabras llamadas *hits* (aciertos). Los *hits* registran la palabra, la posición en el documento, una aproximación al tamaño de la fuente, y la capitalización. Las apariciones en el texto de anclaje son mantenidas en otro archivo.

Una lista de *hits* consiste en una lista de ocurrencias de una palabra en un documento particular. Hay dos tipos de *hits*: de fantasía (*fancy*) y planos. Los hits de fantasía incluyen los que ocurren en una URL, título, texto de anclaje, o elemento (*tag*) HTML meta. Los *hits* planos incluyen todos los demás.

Un *hit* plano consiste en un bit de capitalización, 3 bits para el tamaño de la fuente, y 12 bits para la posición en el documento. El tamaño de la fuente se representa en relación con el resto del documento y puede tomar 7 valores diferentes.

El uso del tamaño de fuente relativo al resto del documento se justifica para no considerar un documento más importante que otro sólo porque uno está escrito con una fuente más grande.

Un *hit* de fantasía consta también de un bit de capitalización, un valor especial de tamaño que sirve para indicar que es un *hit* de este tipo, y 8 bits de posición.

En el momento de la consulta, Google busca los documentos (hasta cierta cantidad fijada en el sistema) que contienen todos los términos de la consulta.

La función de *ranking* de Google utiliza la información de la posición, la fuente y la capitalización, para ponderarla luego con la proveniente del texto de anclaje y el puntaje PageRank del documento. La combinación de todos estos factores es difícil, y la idea en Google es no asignar demasiada influencia a ninguno de estos factores por sobre los demás.

Primero, consideremos el caso más simple, la consulta de **una palabra** sola. Para ordenar un documento ante una consulta de una palabra, se examina la lista de *hits* de esa palabra en ese documento. Google considera que cada *hit* es de un determinado tipo (título, anclaje, URL, texto plano, texto plano con fuente pequeña, etc.), cada uno de los cuales tiene su propio **peso de tipo**.

Los pesos de tipo conforman un vector indexado por tipo. Google cuenta la cantidad de *hits* de cada tipo en la lista de *hits* del documento. Luego, cada cuenta es convertida en un **peso de conteo**. Los pesos de conteo crecen linealmente a la par del conteo de los hits, pero pronto son acotados para evitar una excesiva influencia por demasiadas repeticiones.

Se toma entonces el producto interno del vector de los pesos de conteo y de los pesos de tipo para arribar a un puntaje RI del documento. Finalmente, el puntaje RI se combina con PageRank para darle el puntaje final al documento.

Para una búsqueda **multipalabra**, la situación es más complicada.

Ahora deben recorrerse múltiples lista de *hits* al mismo tiempo para que los hits que ocurren cercanos en un documento sean más pesados que los que ocurren lejos entre sí. Los *hits* (relacionados con un mismo documento) encontrados en las múltiples listas de *hits* son reunidos para juntar *hits* cercanos.

Para cada conjunto de hits reunidos se calcula una medida de proximidad. La proximidad se basa en la distancia que separa las posiciones de los *hits* en el documento o en el anclaje, y es clasificada en 10 valores distintos, que van desde "frase" a "ni siquiera cerca".

Ahora se cuenta no ya la cantidad de *hits* de cada tipo (de *hit*) sino la cantidad de hits de cada **tipo y proximidad**. Cada par tipo-proximidad tiene su propio peso asociado, que conforma el vector de tipo-proximidad. Luego, los conteos se convierten nuevamente en pesos de conteo, evitando influencias por repetición. Finalmente, se toma el producto interno entre el vector de pesos de conteo y el vector de pesos de tipo-proximidad, para computar el puntaje RI y combinarlo posteriormente con PageRank.

Los autores de Google no aclaran cuáles son los pesos de tipo y los pesos de tipo-proximidad utilizados, aunque reconocen que estos parámetros de la función de *ranking* son elegidos y varían de acuerdo a nociones subjetivas, a las que denominan irónicamente de "magia negra". Sin embargo, cuando tienen que cambiar estos parámetros, evalúan su impacto mediante un mecanismo de retroalimentación del usuario que incorporaron al buscador. Esta herramienta de *user feedback* se utilizó sólo con este fin y no se incluyó en la interfaz pública de Google.

3.7.3. Identificación de páginas autorizadas y concentradoras

En esta sección estudiaremos el algoritmo HITS, presentado en [KLE/98].

Hemos visto que la estructura en red de un ambiente hiperenlazado puede ser una fuente de información valiosa acerca del contenido del mismo.

El objetivo central de HITS es la destilación de **temas amplios** de búsqueda, a través del descubrimiento de **fuentes autorizadas** en tales temas. Para ello se presenta una formulación algorítmica de la noción de autoridad, basada en la relación entre un conjunto de páginas relevantes autorizadas (*authorities*) y el conjunto de páginas **concentradoras** que las identifica en la estructura de enlaces.

La idea se origina en el problema de la búsqueda en la Web, que es el proceso de descubrir páginas que son relevantes para una consulta dada. La calidad de un método de búsqueda necesita de una evaluación humana, debido a la subjetividad inherente en conceptos como la relevancia. Se intenta entonces atenuar la carencia de funciones objetivas concretas que correspondan a las **nociones humanas de calidad**.

Para comenzar, se distinguen tres **tipos de consultas**: específicas, de temas amplios y de páginas similares. La dificultad en el manejo de las primeras es el problema de la escasez: hay muy pocas páginas que contienen la información requerida, y es muy difícil determinar la identidad de tales páginas. Las consultas de temas amplios, por el contrario, imponen una respuesta de varios miles de páginas web relevantes. Surge aquí el problema de la abundancia: la cantidad de páginas que se devuelven es demasiado grande para que la digiera un humano. Es necesario entonces filtrar un pequeño conjunto de páginas autorizadas.

En la Web resulta complicado reconocer estas autoridades. Primero, hay páginas importantes que no incluyen en su contenido textual una descripción de sí mismas. Por ejemplo, los motores de búsqueda. Segundo, páginas como la de la Universidad de Harvard que seguramente incluye "Harvard" se ve acompañada de millones de páginas que tienen ese término y no son importantes. Podemos sospechar que no existe una medida puramente endógena de una página que nos permita determinar su autoridad.

El análisis de la estructura de los hiperenlaces entre páginas web nos puede ayudar nuevamente a abordar las dificultades expuestas. Los **hiperenlaces** codifican una cantidad considerable de **juicio humano latente** y es precisamente este tipo de opinión lo que se necesita para formular la noción de autoridad.

En este sentido, un enlace desde la página p a otra página q nos puede indicar que el creador de p ha conferido autoridad a q por medio de tal enlace. Los enlaces nos permiten encontrar una autoridad potencial a través de las páginas que la apuntan, sorteando el problema de la ausencia de auto-descriptividad.

Sin embargo, algunos enlaces no confieren autoridad, sino que sirven a los fines navigacionales o publicitarios.

Otro problema es encontrar un balance entre los criterios de relevancia y popularidad, que contribuyen a nuestra noción de autoridad. Si retornamos las páginas con más enlaces entrantes, corremos el riesgo de considerar páginas universalmente populares como Yahoo! como autoridad para cualquier consulta.

El modelo considera la relación existente entre las autoridades para un tema amplio y las páginas que apuntan a muchas autoridades relacionadas, que llamamos concentradores. El algoritmo opera sobre **subgrafos focalizados** de la Web que se construyen a partir de la salida de un MB basado en texto.

Analicemos cómo se construyen tales subgrafos.

Veamos a la Web como un grafo dirigido G = (V, E): los nodos (V) son las páginas y los enlaces (E) los ejes. Un eje (p,q) indica un enlace de la página p a la q. Si $W \subseteq V$, el subgrafo inducido G[W] tiene como nodos a W y los ejes son todos los enlaces entre páginas de W.

Supongamos una consulta sobre un tema amplio, especificada por la cadena σ. La idea es concentrar el esfuerzo en un subgrafo de páginas relevantes. Como vimos, no serviría tomar para el análisis al conjunto Q_σ de las páginas que contienen a σ pues muchas autoridades no estarían allí y podría contener millones de páginas.

Nos enfocaríamos pues en una colección S_σ tal que:

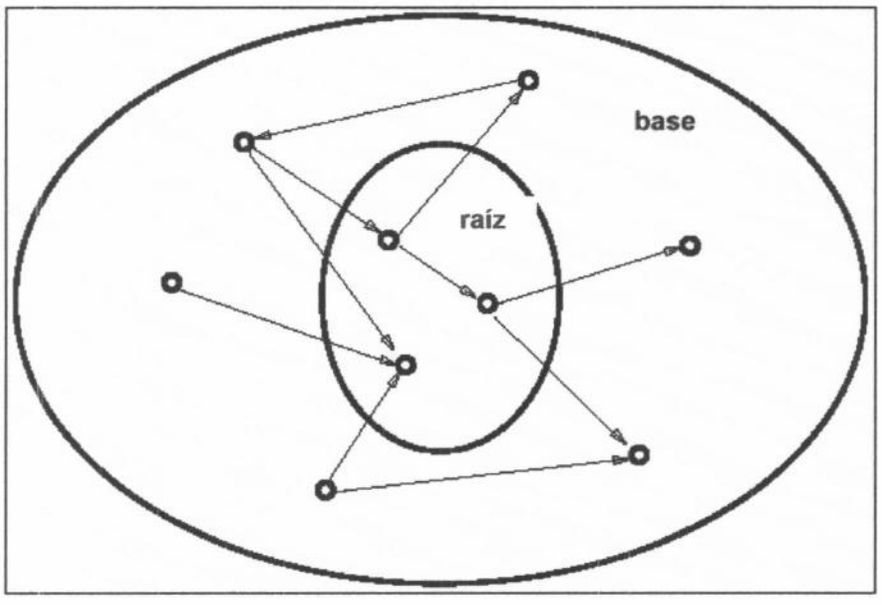
(a) Sea relativamente pequeña;

(b) Sea rica en páginas relevantes; y

(c) Contenga muchas de las autoridades más fuertes.

Se toman las primeras t páginas resultado de algún buscador basado en texto, el conjunto raíz R_σ. Satisface (a) y (b), pero está lejos de (c) pues basta observar que todas las páginas en R_σ contienen a σ y hemos argumentado que ni siquiera Q_σ satisfaría la condición (c).

Se ha observado que hay muy pocos enlaces entre páginas de R_σ. Pero es probable que las autoridades estén apuntadas por al menos una página de R_σ. Entonces obtenemos el conjunto base S_σ expandiendo R_σ para incluir los enlaces que ingresan y parten de R_σ. Limitamos a *d* la cantidad de páginas que apuntan a una página de R_σ para agregarlas a S_σ. Distinguimos dos tipos de enlaces en $G[S_\sigma]$: *intrínseco* es aquél que apunta a una URL con el mismo nombre de dominio (navigacional en general); y *transversal* el que no. Entonces se quitan de S_σ los enlaces intrínsecos por carecer de información acerca de las autoridades, dando lugar al grafo G_σ.

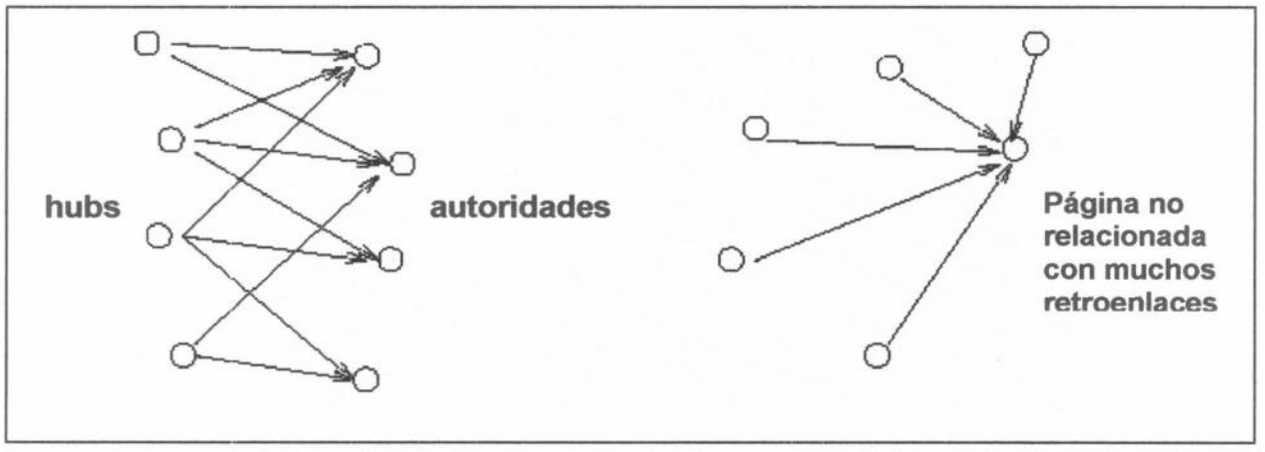


Ahora estamos listos para empezar a computar las páginas concentradores y las autorizadas.

El enfoque más simple para extraer las **autoridades** del grafo G_σ sería tomar las que tienen más enlaces entrantes. Previamente descartamos este argumento cuando se aplicaba a la colección de todas las páginas que contenían a la consulta; pero ahora hemos construido explícitamente una colección de páginas relevantes que contiene la mayoría de las autoridades que nos interesa. Por ello, estas autoridades pertenecen a G_σ y están muy referenciadas por páginas de G_σ.

Sin embargo, subsisten algunos problemas. El caso de páginas universalmente populares ante cualquier consulta todavía no ha sido resuelto. No recurriremos al análisis del contenido

textual para sortear esta complicación, pero sí a la estructura de enlaces. Las autoridades no sólo deben ser populares, también deben tener superposición en las páginas que las apuntan. Estas serán las páginas **concentradoras**, con enlaces a múltiples autoridades. Ahora podemos distinguir las que son meramente populares de las que también son autoridades.



Los concentradores y las autoridades exhiben una **relación de mutua influencia**: un buen concentrador apunta a muchas buenas autoridades; una buena autoridad es apuntada por muchos buenos concentradores. Necesitamos un método para romper este círculo.

Mediante un **algoritmo iterativo** aprovechamos la relación entre concentradores y autoridades. Este método mantiene pesos numéricos no negativos para cada página. Estos son el **peso de autoridad** $x^{(p)}$ y el **peso de concentrador** $y^{(p)}$. Se mantiene el invariante de que los pesos de cada tipo están normalizados para que su suma sea 1:

$$\sum_{p \in G\sigma} \left(x^{(p)}\right)^2 = 1 \qquad \sum_{p \in G\sigma} \left(y^{(p)}\right)^2 = 1$$

Las páginas con valores más grandes de pesos para x e y, son mejores autoridades y concentradoras.

Es natural expresar la relación mutua de la siguiente manera: si p apunta a muchas páginas con altos valores de x, entonces debe recibir un alto valor de y; y si p es apuntada por muchas páginas con altos valores de y, entonces debe recibir un alto valor de x.

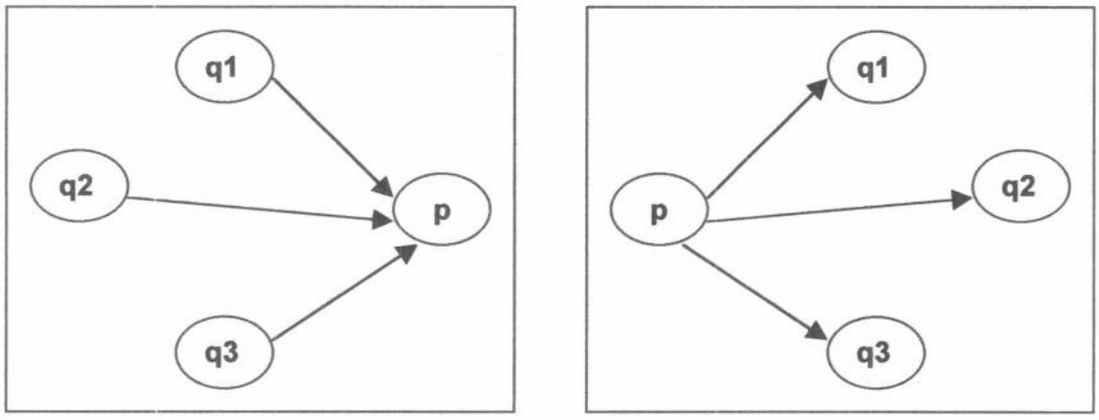
Se definen entonces dos operaciones sobre los pesos, que se denotan *I* y *O*. La operación *I* actualiza los pesos de x así:

$$x^{(p)} \leftarrow \sum_{q:(q,p) \in E} y^{(q)}$$

La operación *I* actualiza los pesos de x así:

$$y^{(p)} \leftarrow \sum_{q:(p,q) \in E} x^{(q)}$$

Las operaciones I y O son la manera en que las concentradoras y las autoridades se enfatizan unas a otras, tal como podemos verlo en los siguientes gráficos.



Ahora, para lograr el equilibrio, se deben aplicar las operaciones alternadamente hasta alcanzar un punto fijo. Una vez allí, se pueden filtrar las *c* primeras concentradoras y autoridades, tomando las *c* coordenadas más grandes del vector *y*, y del vector *x* respectivamente.

Para calcular los puntos fijos x* e y*, se demuestra que si:

- A es la matriz de adyacencia del grafo G_σ de páginas,
- *x* es el vector normalizado de pesos de autoridad,
- *y* es el vector normalizado de pesos de concentrador,

entonces y* es autovector principal de AA^T, y x* el autovector principal de A^TA.

Con esto también queda demostrado que el algoritmo iterativo converge, aunque no es necesario esperar a la convergencia. En los experimentos, no se necesitó superar las 20 iteraciones con valores de *c* entre 5 y 10.

Se observa en los experimentos que muy pocos resultados del conjunto raíz R_σ resultan buenos concentradores o autoridades. Por ejemplo, en la consulta "Gates", la página que resultó con el mayor puntaje de autoridad, aparecía en el lugar 123 de AltaVista, el buscador del que se extrajo R_σ, a pesar de tener la palabra "Gates" en el título.

Es importante destacar dos cuestiones. Primero, el único momento en que se usó el contenido textual de las páginas fue en el inicial llamado a la "caja negra" del buscador basado en texto, que produjo el conjunto raíz R_σ. Después de esto, el análisis ignoró el contenido textual de las páginas. No es la intención demostrar que es mejor ignorar el texto para buscar páginas autorizadas; sino más bien mostrar que se puede conseguir bastante con sólo realizar un análisis puro de la estructura de enlaces.

Segundo, para muchos temas amplios, el algoritmo produce páginas que pueden considerarse autoridades de la web en su totalidad, a pesar de actuar en un pequeño subgrafo de la web.

La utilidad de este algoritmo en la Web se hace evidente al observar que, para un amplio espectro de temas, las autoridades más fuertes evitan apuntarse entre ellas. Considérese, por ejemplo, los buscadores o las fábricas de automóviles. Por lo tanto, deben conectarse por medio de una capa intermedia de páginas concentradoras relativamente anónimas, que apuntan a un conjunto de autoridades relacionadas temáticamente.

Cuando el concepto especificado en la consulta σ no es lo suficientemente abarcador o amplio, puede ocurrir que no haya suficientes páginas relevantes en G_σ. El algoritmo produce, en este caso, autoridades representativas de un concepto o tema más amplio que compite con σ.

Este proceso es la **difusión**, que no necesariamente es negativa pues abstrae el tema de la consulta y lo enmarca.

Algunos resultados sugieren, sin embargo, que el uso de autovectores no principales, combinado con el uso de contenido textual puede ser una forma simple de extraer colecciones de páginas autorizadas que sean relevantes a una consulta referida a un tema muy específico o no suficientemente amplio.

3.7.4. Algoritmo de Híper Búsqueda o *HyperSearch*

En esta sección describiremos el algoritmo HyperSearch, presentado en [MAR/97].

Nuevamente, el gran problema a encarar es la clasificación de los objetos web en respuesta a una necesidad del usuario. En este algoritmo, se utiliza una medida del contenido informativo de un objeto Web, denominada **hiperinformación**.

El problema con los MB basados sólo en texto es que ven a los objetos Web sólo como piezas textuales, pero no tienen en cuenta la estructura Web de la que forman parte.

El poder de la Web reside en su capacidad para redirigir el flujo de información por medio de los hiperenlaces, por lo que tiene sentido analizar cuidadosamente la estructura Web. Se ha observado que el tomar la medida de la popularidad de una página (cantidad de retroenlaces) ha redundado en una ganancia pequeña en la precisión de la información.

Marchiori presenta una noción de información total de un objeto Web, que no se compone solamente de su información textual, sino también de la hiperinformación que es la medida del contenido informativo potencial de dicho objeto respecto a la estructura Web en que se encuentra. Informalmente, mide cuánta información es obtenible navegando desde dicho objeto Web.

Se aplica el algoritmo a una estructura Web estática, es decir, que se descarta el comportamiento dinámico de la estructura Web en el tiempo en que se la considera. Esta suposición no es tan restrictiva pues según observaciones empíricas la Web es localmente consistente en un tiempo suficientemente pequeño.

¿Cuál es la diferencia entre información e hiperinformación? La mayoría de los MB simplemente olvidan la parte híper del hipertexto (los enlaces), calculando sus puntajes en función de los componentes textuales. El pesar de manera diferente a las palabras que aparecen en el título o en los encabezados, no se contradice con lo antedicho pues esas secciones no son otra cosa que simples **atributos** del texto.

Una excepción a este comportamiento es considerar la visibilidad o popularidad de un objeto Web, como algunos de los MB hacen. El problema se evidencia al notar que la visibilidad no dice nada acerca del contenido informativo de un objeto Web. Este razonamiento sería correcto si todos los objetos Web fueran conocidos por los usuarios y los MB, algo claramente falso. En concreto, la popularidad es algo completamente diferente a la calidad.

Lo que se propone entonces es sumar la hiperinformación (HIPERINFO) a la información textual (INFOTEXTO), dando lugar a la información (total) en la Web (INFO).

Para cualquier objeto web A tenemos:

$$INFO(A) = INFOTEXTO(A) + HIPERINFO(A)$$

Aunque en general estas funciones dependen de una consulta específica, consideraremos en este desarrollo que dicha consulta está implícita.

La presencia de enlaces en un objeto Web aumenta, sin dudas, su contenido informativo con el contenido informativo de los objetos Web apuntados. Sólo resta determinar en qué magnitud.

Recursivamente, los enlaces presentes en los objetos Web apuntados, agregan su contribución, y así siguiendo. Por ello, el análisis del contenido informativo de un objeto Web A debe incluir a todos los objetos Web alcanzables desde él mediante sus hiperenlaces.

Por razones prácticas, debemos frenar el análisis a cierta profundidad, por lo que se pone una cota a la profundidad de evaluación. La definición de **profundidad** es natural: dado un objeto Web O, la profundidad relativa de otro objeto Web O' es la mínima cantidad de enlaces que se deben activar para alcanzar O' desde O. Entonces, decir que la evaluación tiene una profundidad *k* significa que consideramos a los objetos Web a una profundidad menor o igual a *k*.

Fijando una cierta profundidad, seleccionamos un **vecindario local** de objetos Web en la WWW. Ahora debemos determinar la hiperinformación de un objeto Web A con respecto a su vecindario (la hiperinformación con profundidad *k*). La notamos $HIPERINFO_{[k]}$, y la correspondiente información total $INFO_{[k]}$. En lo que sigue se sobreentiende el nivel k.

Se asume que INFO, HIPERINFO e INFOTEXTO son funciones de los objetos Web en los números reales no negativos. El sentido intuitivo es que a mayor información de un objeto Web, mayor será el correspondiente número.

Al implementarlas, sin embargo, estas funciones deben estar acotadas, es decir que para todo A, hay un M tal que $INFO(A) \leq M$. Se asume entonces que INFOTEXTO tiene una cota superior de 1, y que HIPERINFO está acotada.

Para empezar, consideremos el caso simple de tener un solo enlace en cada objeto Web.

Con profundidad 1, tenemos un enlace de un objeto A a otro B.

Un enfoque ingenuo es agregar la información textual del objeto apuntado. Esta idea es asimilable a reemplazar un objeto Web con el que se obtiene de reemplazar los enlaces con los correspondientes objetos Web apuntados.

Veamos por qué no es correcto. Si un objeto A tiene información textual casi cero, y apunta a B que posee una alta información total, A tendrá mayor contenido informativo que B, cuando lo que queda claro es que el usuario está mucho más interesado en B que en A.

El problema se acrecienta al aumentar la profundidad del análisis. Consideremos que B_k está a una profundidad k desde A, i.e. para ir de A a B se necesitan k *clicks*, con k grande (e.g. k>10).

Sean A, B_1, ..., B_{k-1} con contenido informativo casi cero, y B_k con información total muy alta. Entonces A tendrá una información total mayor a B_k, lo que es paradójico ya que A tiene una relación demasiado débil con B_k como para ser más útil que B_k mismo.

El problema es que la información textual apuntada por un enlace no puede considerarse real sino **potencial**: para el usuario hay un **costo** para alcanzar dicha información, hacer *click* y esperar.

La solución consiste en que la contribución a la hiperinformación desde un objeto Web ubicado a profundidad *k* no sea su información textual, sino su información textual ponderada con un factor de disminución que crezca con la profundidad, o la lejanía de la información al usuario.

Se toma un factor de disminución que cae exponencialmente con la profundidad, es decir que la contribución a la hiperinformación de A dada por un objeto B a profundidad *k* es F^k . INFOTEXTO(B) para un factor de disminución F (0<F<1).

Entonces, en el ejemplo vemos que:

$$\text{HIPERINFO(A)} \neq \text{INFOTEXTO}(B_1) + \text{INFOTEXTO}(B_2) + \ldots + \text{INFOTEXTO}(B_k)$$

sino

$$\text{HIPERINFO(A)} = F.\text{INFOTEXTO}(B_1) + F^2.\text{INFOTEXTO}(B_2) + \ldots + F^k \,.\, \text{INFOTEXTO}(B_k).$$

La información textual de A puede verse como un caso trivial de hiperinformación, ya que:

$$\text{INFOTEXTO(A)} = F^0.\text{INFOTEXTO(A)}$$

pues el objeto se encuentra a "distancia cero" de sí mismo.

¿Por qué tomar un factor de decaimiento exponencial y no polinomial, por ejemplo?

Consideremos la información total de un objeto Web con profundidad *k*. Si tenemos:

es razonable asumir que la información total de A (con profundidad *k*) está dada por el contenido textual de A, más la información total ponderada de b (c con profundidad *k*-1). También es razonable asumir que esta ponderación o decaimiento puede aproximarse multiplicando por un factor F (0<F<1). Tenemos entonces la relación recursiva:

$$\text{INFO}_{[k]}(A) = \mathbf{F}.\ \text{INFO}_{[k-1]}(B)$$

Como para todo objeto A tenemos que:

$$\text{INFO}_{[0]}(A) = \text{INFOTEXTO(A)}$$

eliminamos la recursión, y si

entonces:

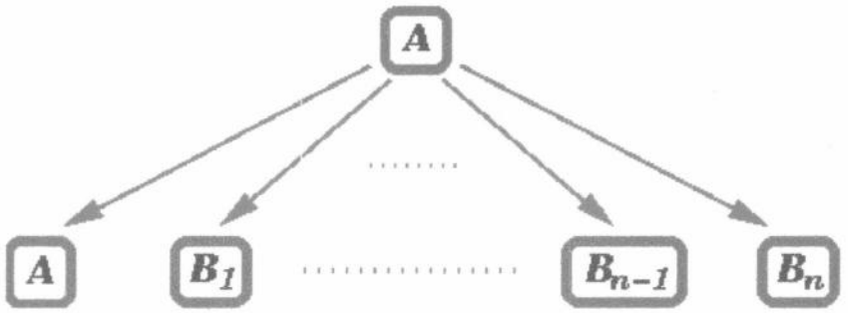
$$\text{INFO(A)} = \text{INFOTEXTO(A)} + \mathbf{F}.\ (\text{INFOTEXTO}(B_1) + (\mathbf{F}.\ \text{INFOTEXTO}(B_2) + \ldots + \mathbf{F}.\ \text{INFOTEXTO}(B_k))))$$

O sea:

$$\text{INFO(A)} = \text{INFOTEXTO(A)} + \mathbf{F}.\text{INFOTEXTO}(B_1) + \mathbf{F}^2.\text{INFOTEXTO}(B_2) + \ldots + \mathbf{F}^k \,.\, \text{INFOTEXTO}(B_k)$$

Que es la razón de decaimiento que se había adoptado.

Ahora pasemos al caso en que hay más de un enlace en el mismo objeto Web. Asumiremos por simplicidad que la profundidad es 1. Tenemos la siguiente situación:



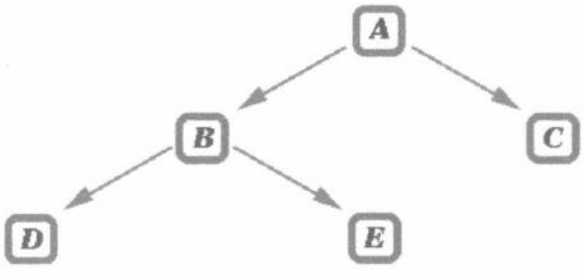
¿Cuál es la hiperinformación en este caso? La respuesta más fácil, sumar las contribuciones de cada enlace (i.e. $\mathbf{F}$.INFOTEXTO(B_1) + ... + $\mathbf{F}$.INFOTEXTO(B_{n-1}) + $\mathbf{F}$.INFOTEXTO(B_n)) no es viable pues la hiperinformación tiene que estar acotada.

Uno podría pensar que la interpretación de un enlace como información potencial llevaría a la fórmula antedicha. Sin embargo, los enlaces no pueden ser activados al mismo tiempo, sino de manera secuencial. En el mejor caso, se seleccionará el enlace más informativo, el segundo más informativo, y así. Supongamos que el orden es INFOTEXTO(B_1) ≥ INFOTEXTO(B_2) ≥ ... ≥ INFOTEXTO(B_n). Entonces:

$$\text{HIPERINFO}(A) = \mathbf{F}.\text{INFOTEXTO}(B_1) + ... + \mathbf{F}^{n-1}.\text{INFOTEXTO}(B_{n-1}) + \mathbf{F}^{n}.\text{INFOTEXTO}(B_n)$$

Esta fórmula nos da una cota, para cualquier cantidad de enlaces, de F/(F+1).

Veamos el caso más general, con enlaces múltiples en una misma página y una profundidad *k* arbitraria).

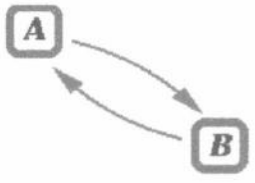


Con $\mathbf{F}$ = 0.5; INFOTEXTO(B) = 0.4; INFOTEXTO(C) = 0.3; INFOTEXTO(D) = 0.2; INFOTEXTO(E) = 0.6.

La secuencia de selecciones que maximiza la hiperinformación de A consiste en partir de A hacia B, luego a E, luego a C, y terminar con D:

$$\text{HIPERINFO}(A) = \mathbf{0,5}.\text{INFOTEXTO}(B) + \mathbf{0,5}^2.\text{INFOTEXTO}(E) + \mathbf{0,5}^3.\text{INFOTEXTO}(C)$$
$$\mathbf{0,5}^4.\text{INFOTEXTO}(D) = 0,4.$$

Finalmente, hay algunos problemas con el modelo. Veamos el caso del enlace que retorna.



Tenemos una recursión, en la que la información textual de A es sumada repetidamente (con un factor de decaimiento cada vez más bajo) a la hiperinformación de A. Esto es porque de A podemos ir a B, y luego volver a A, etc.

El más importante de los problemas proviene de las posibles **duplicaciones de los enlaces**, que consiste en tener dos enlaces en un objeto Web apuntando al mismo objeto Web. La solución pasa por no considerar todas las posibles secuencias de selecciones sino sólo las que carecen de repeticiones. Esta idea es consistente con la noción intuitiva de pensar que si uno ya observó un objeto, ya dispone de su información, y no tiene sentido obtenerla de nuevo.

Además, hay muchos tipos de enlaces, y cada uno debería recibir un tratamiento especial. Algunos de estos tipos de enlaces son:

- enlaces locales (apuntan al objeto en el que se encuentran) que deben ser descartados;
- enlaces de marcos (*frames*) que deben ser expandidos, es decir si A tiene un enlace de marco hacia B, se debe expandir B en A, ya que el usuario no ve ningún enlace.
- Otros enlaces que deben ser activados: imágenes (*tag* IMG), de fondo. Como es complicado obtener información textual de las imágenes, debe considerarse por ahora el nombre de las imágenes (atributo ALT). Esto ocurre también con videos, sonidos, etc.

3.7.5. Comparación de los algoritmos de ranking

Para comenzar con la comparación de los algoritmos expuestos, veamos qué es lo que hace especiales a las páginas Web que nos puede ayudar a mejorar la efectividad de recuperación de los motores de búsqueda.

La primera característica esencial de las páginas web es que son documentos sumamente estructurados, vía tags. En el presente, la mayoría de las páginas web están en HTML, y en el futuro se extenderá el uso del XML.

Estos tags conllevan una rica información acerca de los términos usados en los documentos. Hemos visto cómo Google utiliza dicha información como una pista acerca de la importancia con que un término indica el contenido del documento.

Los estudios indican que a mayor peso asignado a los términos debido a su ubicación o sus características de estilo puede derivar en una mayor efectividad de recuperación respecto de sistemas que no hacen uso de este tipo de información [CUT/97].

La segunda característica especial de las páginas web es que están extensamente enlazadas. Vimos que un enlace de una página A a otra página B, nos provee de un camino conveniente para navegar de una página a la otra.

Como primera observación, un enlace nos indica que el contenido de las páginas intervinientes están relacionadas de alguna manera. Segundo, mediante el enlace, el autor de la página A puede dar su valoración (positiva o negativa) acerca de la página B.

Hemos visto, con PageRank, cómo se utilizó la información de enlaces para computar la importancia global de las páginas web basada en si una página estaba apuntada por muchas páginas o por páginas importantes. Esta medida es independiente (se calcula antes o después) de cualquier consulta de usuario.

El grado de relevancia de una página puede ser una combinación de su similitud hacia la consulta y su PageRank, como vimos que se implementó en Google.

La información de enlaces también fue utilizada en HITS para computar la autoridad, o grado de importancia, de las páginas web **respecto de un tema** determinado. Aquí vemos una primera diferencia entre HITS y PageRank, pues este último no refiere a ningún tema.

Una consulta puede ser considerada un tema en el sentido del párrafo anterior.

Como vimos con Google, otra manera de utilizar la información de los enlaces, es combinarla con la estructura HTML para considerar los términos que circundan los retroenlaces de un documento. Estos términos tienen a menudo una relación semántica con los términos utilizados para indexar al documento apuntado por el enlace. En general, una página web puede estar apuntada por muchas otras páginas web, y por ello puede tener muchos términos de anclaje asociados.

El algoritmo HITS ignora el contenido textual luego de tomar el conjunto raíz a partir de un MB basado en texto.

Una modificación al algoritmo descripto, sirvió para abordar la categorización automática de recursos web de alta calidad [CHA/98]. Se agregaron en este trabajo, además del análisis de los enlaces, algunas medidas basadas en el texto, como por ejemplo ponderar la contribución de cada enlace de acuerdo al texto de anclaje de la página origen.

Una de las diferencias entre el algoritmo HITS y PageRank es que en este último la autoridad se delega entre autoridades, sin la noción de páginas concentradoras. La forma de lidiar con autoridades que no confieren autoridad (puntos muertos), es la noción de salto al azar incluida en la fórmula de PageRank.

Tanto HITS como PageRank generan puntajes de calidad para todas las potenciales páginas relevantes usando métodos iterativos de computación basados en la propagación o acumulación de las aprobaciones supuestamente implícitas por los enlaces. Sin embargo, las opiniones vertidas en esos enlaces tienen pesos diferentes: dependen del puntaje de calidad obtenida por sus autores.

Estos sistemas implementan implícitamente una función cooperativa de *ranking* de páginas web. Se observa que tales funciones aumentan significativamente la calidad de las respuestas.

Aun así, la precisión de estas funciones tienen mucho por mejorar, ya que no todos los enlaces significan la misma aprobación o alguna aprobación siquiera.

Según [HEN/98], HITS no funciona bien en todos los casos debido a tres razones:

- Las relaciones mutuamente recíprocas entre diferentes sitios. Expandiremos la explicación de este punto al hablar de la PMB o manipulación de los MB.
- Los enlaces generados automáticamente. Los documentos web generados por aplicaciones (e.g. de autoría web, de conversión de bases de datos) a menudo tienen enlaces que fueron insertados por la herramienta.
- Los nodos no relevantes. El grafo de vecindad que se construye en HITS contiene frecuentemente documentos no relevantes al tema de la consulta. Si estos nodos están bien conectados, aparece el problema del desplazamiento del tema inicial, pues las autoridades y concentradoras más altamente consideradas tienden a no versar acerca del tema original. Por ejemplo, al correr el algoritmo para "mango fruit", la computación se desplazó hacia el tema general "fruit".

Al igual que HITS, la técnica de HiperBúsqueda se puede implementar sobre cualquier función de información textual, manipulando sus puntajes "locales" para obtener un puntaje más "global" de un objeto Web. Es decir, se puede integrar con los MB existentes posprocesando sus puntajes.

A diferencia de HITS y PageRank, el protagonismo de la información textual de los objetos Web es fundamental en el cálculo de los puntajes de los objetos.

A semejanza de HITS, y a diferencia de PageRank, el cálculo de la importancia de los objetos se realiza respecto de una consulta especificada por el usuario.

Nótese que [MAR/97] hace una diferencia entre los enlaces internos y externos al dominio en que se encuentra la página, algo que se distinguía tanto en HITS como en PageRank. Esta diferenciación, sin embargo, no es inherente al modelo de hiperbúsqueda, sino que se aconseja su uso para evitar que un autor mejore el puntaje de los objetos Web que componga mediante enlaces cruzados entre objetos de distintos dominios que administre.

El problema del *rank sink* o acumulador de puntaje, que ocurría cuando dos o más páginas se apuntaban entre sí [PAG/98], circularmente, se resuelve en el modelo de HiperBúsqueda mediante el factor de decaimiento que disminuye gradualmente la recursividad en el cálculo del puntaje.

Capítulo 4. Persuasión de Motores de Búsqueda

4.1. La presión del mercado sobre las funciones de ranking

Un aspecto poco analizado, pero que sin embargo tiene un impacto fundamental en las funciones de *ranking* de los MB de la Web, es la manipulación de dichas funciones que algunos autores de páginas Web intentan ejercer, y de hecho ejercen.

Este tema ha sido abordado contadas veces en la literatura, y mucho menos con la dedicación y el detalle que merece. En general, en las publicaciones científicas que dan cuenta de los diversos algoritmos que se utilizan o utilizaron en MB sobre colecciones de documentos de la Web, no se habla de las técnicas de manipulación o si se lo hace es de manera lateral y demasiado breve.

Es importante señalar que, en la literatura científica acerca del tema (y en los sitios Web especializados), se ha bautizado a este ataque directo a los MB de diversas formas: persuasión de motores de búsqueda (PMB o SEP por *search engine persuasion*), optimización de MB, manipulación de MB, y también *web spam*.

Esta última terminología se adopta en analogía con el otro tipo de *spam*, el más conocido hasta ahora, que consiste en el envío de correo electrónico no solicitado. En ambos casos, lo que se está realizando es el sabotaje a determinado servicio, en un caso el de *email* y en el otro de búsqueda de información. Como consecuencia de la utilización indebida de estos medios o servicios con fines comerciales, el sabotaje finalmente lo sufre el usuario.

Si en el caso del correo, la batalla se da en los servidores de los ISP, de los portales y en general en cualquier servidor de email, cuando hablamos de servicios de búsqueda debemos concentrarnos solamente en los MB que realicen su recolección, indexación y *ranking* de manera automática.

Por ello, es evidente que los directorios, que suponen una inspección humana a las páginas antes de agregarlas, no son susceptibles de manipulaciones de este tipo.

Tampoco existía este problema en los sistemas de recuperación de información (SRI) tradicionales, que actuaban sobre una colección de documentos cerrada y sobre temas no tan diversos o heterogéneos como ocurre en la Web.

Un importante cambio que supuso la Web para los diseñadores de SRI, es que la **presión competitiva del mercado** provoca la necesidad de una mayor exposición mediática de las empresas. Los responsables de los sitios Web saben que los MB son una gran oportunidad de exponerse ante millones de potenciales clientes. Saben también que el costo de colocar un aviso publicitario en dichos portales de búsqueda crece con la cantidad de personas a las que puedan mostrarse.

En este contexto, los responsables de los sitios Web intentan manipular, con métodos más o menos sofisticados, al servidor de búsqueda para que muestre las páginas del sitio en las primeras posiciones de las consultas relevantes a los productos o servicios que tales empresas ofrecen.

Otra consecuencia indeseada de la presión que ejerce el mercado, se da en el **modelo de negocio** sobre el que se asientan la mayor parte de los MB de la Web. Si bien éste es un aspecto independiente de la PMB, está relacionado, y exponemos esta discusión aquí por su influencia igualmente perniciosa sobre la calidad de los resultados.

Actualmente, el modelo de negocio predominante entre los MB comerciales es la publicidad. Los objetivos del modelo de negocio basado en publicidad no siempre se condicen con brindarle búsquedas de calidad a los usuarios.

Tomemos lo expuesto en [BRI/98], donde la página más relevante de Google para "teléfono celular" fue, en determinado momento, "El efecto del uso del teléfono celular en la atención del conductor", un estudio que explica con detalle las distracciones y los riesgos asociados a conversar por un teléfono celular mientras se conduce. Este resultado apareció primero por su gran importancia juzgada por el algoritmo PageRank.

Es claro que un MB que estuviera recibiendo dinero por mostrar avisos de telefonía celular tendría "dificultades" para justificar la página devuelta por Google, ante los auspiciantes que pagan por dichos avisos. Por este tipo de razones y la experiencia histórica con otros medios, es de esperar que los MB mantenidos por la publicidad serán esencialmente sesgados a favor de los auspiciantes y en contra de los intereses de los usuarios.

Como es muy complejo evaluar los MB, aun para los expertos, el sesgo mencionado en los MB es particularmente dañino. Un buen ejemplo es el caso de OpenText que, según se reporta en [MAR/96], estaba dispuesto a vender a las compañías el derecho de aparecer al tope de los resultados ante determinadas consultas. Este tipo de sesgo es mucho más problemático que la publicidad, porque no está claro quién merecería aparecer allí, y quién está dispuesto a pagar dinero para ser incluido en los resultados. Este episodio generó un escándalo, y OpenText dejó de ser un MB viable.

A pesar de ello, el mercado parece estar dispuesto a tolerar otro tipo de influencias no tan evidentes. Por ejemplo, un MB podría agregar un pequeño factor a los resultados asociados a compañías "amigas" y restar otro factor a las competidoras. Este tipo de sesgo es muy difícil de detectar pero su potencial efecto en el mercado (y los usuarios) podría ser igualmente negativo. Al parecer, actualmente muchos buscadores están volviendo a recurrir rutinariamente a ofrecer un posicionamiento pago.

Agregado a lo expuesto, la entrada de dinero por publicidad a menudo incentiva la baja calidad de los resultados a las consultas. Por ejemplo, se ha notado que un MB importante no devolvió la página central de una línea aérea cuando se ingresó su nombre como consulta. Ello sucedió porque la línea aérea había ubicado un aviso imponente, ligado a la consulta por su nombre. Un MB mejor no habría requerido este aviso y habría perdido posiblemente el dinero proveniente del aviso de la línea aérea.

En general, se puede sostener desde el punto de vista del consumidor que cuanto mejor es el MB, necesitará de una menor cantidad de avisos para encontrar lo que desea. Esto por supuesto erosiona en sus cimientos al modelo de negocio basado en publicidad ostentado por los MB comerciales actuales. Desde luego, siempre habrá anunciantes que induzcan a los clientes a optar por productos mejores o genuinamente nuevos. Pero los avisos publicitarios causan suficiente ruido o intereses contrapuestos, por lo que se torna crucial disponer de un MB que sea transparente y ajeno a intereses comerciales, como puede darse en el ámbito académico.

Google pareció cumplir en un primer momento ese rol de referente, pero actualmente ofrece un servicio de publicidad que aunque ingenioso y no contaminante de la interfaz del buscador, de todas maneras implica la adopción del modelo de negocio que critican sus creadores en [BRI/98].

Este servicio (*AdWords*) focaliza la publicidad en frases, palabras o conjuntos de palabras que elije el anunciante. Los avisos son de texto puro, sin imágenes, pero aparecen a la derecha de los resultados y en otro color. El anunciante paga más para aparecer más cerca del

tope, aunque los avisos también son ordenados de acuerdo a la cantidad de veces que se los activa (*clickthrough*).

4.2. Seguridad de los motores de búsqueda de la WWW

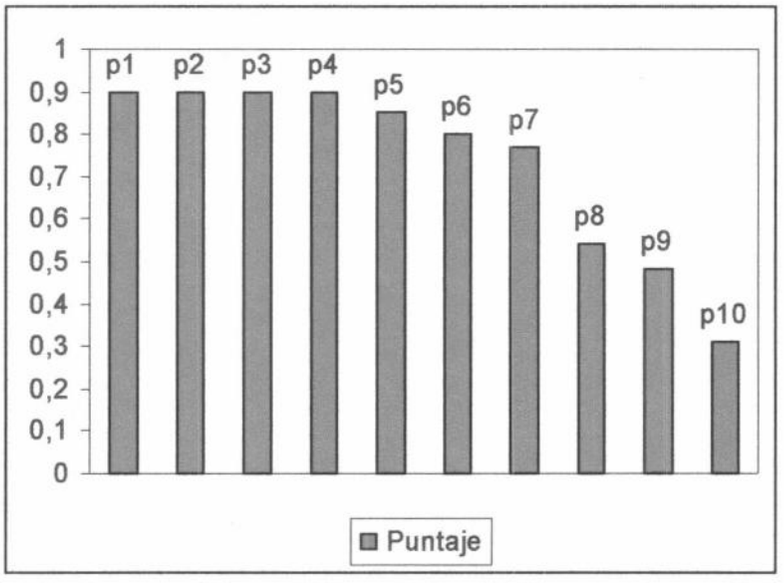
Volviendo al tema principal de este capítulo, la PMB, expondremos a continuación los conceptos más importantes vertidos en [MAR/96], uno de los pocos *papers* que trató sistemáticamente el tema de la manipulación de los MB y sus consecuencias en la adopción de escudos de seguridad que transforman las funciones de *ranking*.

Como fue mencionado varias veces, el tamaño de la Web provoca que los usuarios deban recurrir a los buscadores en general, y a los MB de recolección automática en particular, para poder acceder y manejar la gran cantidad de información disponible.

También por eso, se señaló que los MB se transformaron en el centro de interés del mercado publicitario, lo que volvió esencial la aparición al tope de los resultados. Esto revolucionó a los propios MB, y a las empresas de diseño Web, a las que no sólo se les pide el buen diseño del sitio sino también su ubicación preferencial en los resultados de los MB.

La PMB se puede definir entonces como la manera de engañar a los mecanismos de evaluación de los MB para obtener publicidad gratis.

Hoy es una práctica tan común que provoca serios problemas a los MB, pues la elevación artificial de los puntajes tiene el efecto de tornar inútiles los mecanismos de evaluación, confundiendo al usuario. Un problema común es el llamado **efecto aplanamiento** que ocurre cuando varias páginas tienen el puntaje más alto, como ejemplifica el gráfico.



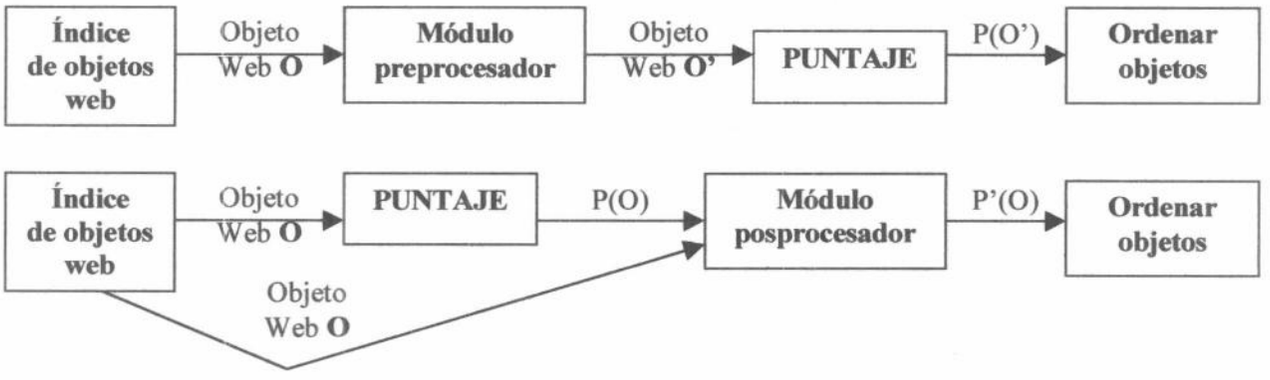
El gran problema de hacer seguros a los MB, es que una vez que se han invertido recursos en obtener una evaluación apropiada del contenido informativo de la WWW, la tarea de reconstruir desde el principio esta función se torna muy cara.

Por ello, los mecanismos de evaluación originales no son modificados, sino tratados como una caja negra, e integrados con **módulos de seguridad** en la forma de pre- y pos-procesadores. Esta modularización y separación de ocupaciones baja notablemente los costos de mantenimiento del software, por ser más comprensible y susceptible a pruebas.

Normalmente, se espera de un MB que retorne los objetos web que son relevantes a cierta consulta, un *ranking*, es decir una secuencia de objetos web ordenados por su relevancia. Para ello utiliza la **función de puntaje** que para un objeto web y la consulta devuelve un número real no negativo, su puntaje.

Si a la función de puntaje la llamamos PUNTAJE, puede mejorarse su seguridad preprocesando los objetos web y luego usando el PUNTAJE sobre estos objetos procesados; o bien posprocesando, usando otra función de puntaje que usa a su vez a PUNTAJE. Se asume que PUNTAJE tiene una cota superior de 1.

Para cada objeto, los módulos de seguridad se aplicarían de la siguiente manera:



4.2.1. Actualización rápida

La PMB es un proceso intrínsecamente **adaptativo**. Para obtener puntajes altos se comienza con intentos (en general imitando a los objetos web de alto puntaje), se ve el puntaje asignado por el MB, se modifica a los objetos web, se observan las nuevas respuestas, y así, hasta obtener un puntaje convincente.

El punto a tener en cuenta es que los MB no refrescan sus datos rápidamente debido al enorme volumen de información presente en la Web. La PMB será más efectiva cuanto más rápido actualicen sus datos los MB.

Una táctica de defensa sería agrandar el período de refresco, pero achicarlo es un factor publicitario importante para los MBs, pues deben mantener su índice al día.

Vemos entonces dos factores que hacen más efectiva a la PMB: la presión del mercado y los tiempos de actualización de índices cada vez menores.

4.2.2. *Spamdexing*: la repetición artificial de palabras clave

La más simple y común de las formas de PMB es la repetición artificial de palabras clave en un objeto web, para aumentar su relevancia. Esto degradó el desempeño de los MB, pues una cantidad creciente de objetos web está diseñado para tener un contenido textual artificialmente alto. Este fenómeno se extendió tanto que los MB han incluido topes o penalizaciones para contrarrestarlo [web spam], como hemos visto que ocurría en Google.

Algunos experimentos muestran que esta técnica es muy efectiva en los MB actuales, aunque el tiempo que tardaron para incluir en los índices las páginas enviadas fue mayor a 2 meses.

Poner un límite a la cantidad de repeticiones no resuelve el problema; lo que produce es la compresión del rango de posibles puntajes y el aumento del efecto aplanamiento. Un estudio confirmó que los MB que penalizaban (Lycos e InfoSeek) son los que más incurren en el aplanamiento.

La penalización puede funcionar así: primero se asigna el puntaje a un objeto web, luego se hacen pruebas para ver si penalizarlo o no. Si falla, se le da un puntaje bajo (puede ameritar su desindexación); si no, su puntaje permanece inalterado.

La confiabilidad de las penas es un problema cuando el fenómeno PMB no se puede detectar con total certeza. Por ejemplo, 5 apariciones consecutivas de una palabra son casi seguro PMB, pero 10 apariciones diseminadas por un objeto web ameritan un delicado análisis, por el peligro de conllevaría "culpar" a una página "inocente".

La situación es crítica:

i. los MB deben hacer pública la especificación de sus penalizaciones para evitar que objetos web que no hacen PMB sean penados; pero
ii. no pueden asumir que un usuario cualquiera leerá las especificaciones, y
iii. al hacerlas públicas, las hace inútiles desde el punto de vista de la seguridad.

Por lo tanto, lo lógico es **no usar penalizaciones**, a menos que la chance de penalizar un objeto que no hace PMB sea muy baja. Varios estudios empíricos coinciden en mostrar la carencia de precisión de los MB que penalizan.

La mejor solución sería no basar el puntaje en la frecuencia de aparición de las claves, como la mayoría de los MB hace; sino en técnicas más sofisticadas de posprocesamiento (que veremos más adelante) o más simples como el truncamiento (preproceso).

El **truncamiento** consiste en ignorar las ocurrencias de una palabra que superan cierto límite. Si éste es muy bajo, deja de lado mucha información; si es muy alto, no es efectivo contra la PMB de repetición. Un umbral de entre 5 y 7 ocurrencias resultó en beneficios inmediatos con WebCrawler.

4.2.3. Componentes fantasmales

Se llama así a las partes del código en un objeto web, que no pueden ser observadas con un navegador. Por ejemplo, los comentarios HTML y los tags descriptivos META son por lejos los más usados. Los atributos de tags que no son comprendidos por los navegadores, son ignorados y son por lo tanto componentes fantasmales. El texto invisible (camuflado con el color de fondo) o muy pequeño, la ubicación del texto en lugares lejanos de la pantalla visible, y el tag NOFRAMES también lo son.

Como el usuario final no los ve, estos componentes pueden ser manipulados y facilitan la PMB sin afectar el diseño de una página. Los componentes visibles afectan la apariencia de un objeto web, que es un factor de importancia desde lo publicitario.

Según estudios, más del 80% concentra la PMB de repetición en componentes fantasmales. Éstos pueden reconocerse sintácticamente, por lo que una solución sería descartarlos al evaluar el puntaje de un objeto web. Esto daría lugar a técnicas PMB más sofisticadas pero que pueden impactar severamente en la apariencia del objeto, lo que puede inhibir en alguna medida a la PMB.

El tag META es ignorado por varios MBs; pero por lo menos Altavista, HotBot e InfoSeek lo usaron en algún momento para inferir claves relevantes para un objeto web. Lo que nos llevaría a descartar todo lo anterior excepto el tag META. Pero dado que se usa este tag para PMB de repetición, se le deben aplicar penalizaciones. Aquí, la chance de penalizar un objeto web que no usa PMB es muy baja, pues quien usa el tag META también debería estar al tanto de las especificaciones de penalización. Hacerlas públicas tendría sentido en este caso, pues sólo se penalizaría el mal uso de este tag.

4.2.4. El enfoque probabilístico

Una forma efectiva de enfrentar a la PMB son las **técnicas aleatorias**. Como caso límite, supongamos un posprocesador que mezcla, al azar, un *ranking* de páginas producido por un MB, y lo muestra al usuario: esto hace inútil a la PMB. Si bien se pierde la información del *ranking* original, este caso da una idea del uso del azar en aras de la seguridad.

El objetivo del enfoque probabilístico es hacer incierto el éxito de la PMB.

Mostremos un posprocesador cuya **efectividad crece en proporción a la presión del mercado**. Primero, fijamos un "parámetro de pérdida" ε, tal que dos puntajes p_1 y p_2 sean indistinguibles si $|p_1-p_2| \leq \varepsilon$. Luego, dado un *ranking* $r_1, r_2, ..., r_k$ agrupamos sus elementos en *clusters*, cada uno conteniendo elementos indistinguibles. El *cluster* superior del *ranking* ρ es:

$$T(\rho) = \{r_i \in \rho: \mid \text{PUNTAJE}(r_1) - \text{PUNTAJE}(r_i) \mid \leq \varepsilon\}$$

Podemos dividir a ρ en un conjunto de *clusters* disjuntos si $C_1 = T(\rho)$, $C_2 = T(\rho/C1)$ hasta que veamos un j tal que $C_j = \varnothing$. El máximo número de *clusters* posible es k, si $\forall i: C_i = \{r_i\}$.

Dado un *cluster*, podemos mezclar sus elementos pues son de similar relevancia. Este ordenamiento local puede hacerse al azar, o de forma tal que la chance de asignar un puntaje bajo a un elemento sea inversamente proporcional al puntaje original.

Los objetos similares pueden intercambiarse. Así, bajo condiciones de **competencia severa**, un objeto web PMB no tiene la certeza de ser el primero. La elección de ε es delicada, pues la mezcla puede interferir con los puntajes originales del MB. La opción menos riesgosa es $\varepsilon=0$, dado que no se cambia la información de la función original de puntajes (sólo se mezclan los objetos de igual puntaje).

Se puede limitar este mezclado al cluster superior, si sólo nos interesa la seguridad de los elementos más relevantes. En esta variante, con $\varepsilon=0$ el azar se aplica si y sólo si se observa el efecto aplanamiento.

Es posible dificultar el proceso general de PMB adaptativo si se cambia al azar la función de puntajes luego de un número fijo de consultas o de días. De esta manera, la PMB debe afrontar la reconstrucción del comportamiento de las diversas funciones de puntaje, que se alternan al azar.

A esta alternancia se la puede pre- y posprocesar con las técnicas anti-PMB ya vistas. En esta sección ya vimos una familia completa, que se obtiene variando ε y la cantidad de clusters superiores a mezclar.

4.2.5. El enfoque del único primero

La efectividad de la PMB de repetición reside en la ingenua suposición hecha por los MBs de que la "frecuencia implica relevancia". Es decir, que la relevancia de una clave es proporcional a la cantidad de veces que aparece en un objeto web. Esta idea se complementa con la de que un "alto porcentaje implica relevancia", según la cual dicha relevancia está dada por el porcentaje de aparición de la clave. Por sí sola, esta última noción de relevancia es un enfoque insuficiente.

La función de puntaje por porcentajes descarta por completo la información de "frecuencia", entendida como cantidad de ocurrencias. La utilidad del porcentaje reside, sin embargo, en que hay a lo sumo una clave para la que se asigna el primer puesto a un objeto web determinado. No previene la PMB, pero limita su alcance ya que la publicidad en la web tiene una necesidad: llegar a diferentes audiencias con distintas claves.

Pero esto no es suficiente para una seguridad real. Se pueden preparar, en un mismo sitio, k copias de un objeto web, modificadas para k claves distintas, con el fin de llegar a la cima de los k puntajes. Para contrarrestarlo, se debe asegurar que haya **una sola clave** para la cual (los objetos de) **un sitio web completo** obtengan el primer puesto.

Sin embargo, en lo concerniente a la manipulación de las funciones de *ranking*, esta precaución tampoco asegura por sí sola que no se incurra en "trampas". No es tan difícil para un mismo autor tener a su disposición varios **nombres de dominio diferentes**, en los cuales ingresar las páginas por él diseñadas. De esta manera, puede hacer que múltiples páginas de diferentes dominios se apunten entre sí, logrando escapar al cerrojo de los algoritmos que no consideraban los enlaces entre objetos de un mismo dominio. Ahora el autor lo único que tiene que hacer es ubicar la página origen o destino de un enlace en cualquier otro dominio donde tenga acceso de escritura.

No existe ninguna manera efectiva de diferenciar a las páginas que incurren en este tipo de maniobras de las que simplemente forman parte de una comunidad de páginas o dominios que se apuntan entre sí, para asignarse importancia o con fines navigacionales. Ni siquiera el análisis de la ubicación de los servidores de nombres (DNS) de los dominios asegura algún éxito.

Es por ello que cualquier medida preventiva que pretenda detectar y castigar a los sitios que deliberadamente practiquen estos métodos "sofisticados" de utilización de enlaces, correrá el peligro concreto de penalizar a los autores o sitios "inocentes".

Por ejemplo, debe tenerse en cuenta que es posible tener varios dominios con nombres distintos bajo una misma dirección en el nivel de red (dirección IP por *Internet Protocol*), o también bajo una misma dirección física (la misma máquina). Estas posibilidades están dadas por lo que se denominan direcciones o **servidores Web virtuales**, que son manejadas por el mismo o distintos procesos servidores Web.

Consideremos el caso de un ISP o una empresa dedicada al servicio de hospedaje de páginas web (*web hosting*). Es claro que no puede disponer de una máquina dedicada para cada nombre de dominio, ni siquiera de una dirección IP por dominio, por ser también un recurso cada vez más valioso por su escasez. Además, la mayoría de las arquitecturas de red que se conectan a Internet intentan minimizar las direcciones IP que son visibles desde afuera (el resto de Internet) por razones de seguridad. Por lo general, son pocas las direcciones IP externas y se usan con fines estrictamente necesarios, como *proxies* (intermediarios) de servicios (*email*, web, *ftp*, etc), *firewalls* (software de filtrado de mensajes de diversos protocolos) servidores de *email* o, como vemos, servidores Web.

Lo que se quiere demostrar con esta argumentación es que, una estrategia que intente reconocer dominios "cómplices" entre sí para evaluar una posible cooperación conjunta en la manipulación, mediante la identificación de direcciones subyacentes en el nivel de red o físico, está destinada a fracasar porque castigará necesariamente a dominios que no tienen participación en la supuesta "conspiración".

Para terminar de completar este panorama no muy auspicioso, debemos agregar a lo antedicho que existe aún otra posibilidad para el potencial manipulador. No necesita utilizar el mismo ISP o servicio para todos sus dominios, puede usar diferentes servicios sin perder por ello ningún tipo de acceso a publicar páginas en los dominios. Al contratar o disponer él mismo de diversas direcciones IP (quizás de distintas subredes para camuflarse mejor), el autor de objetos Web perniciosos, tiene absoluta libertad de publicar páginas en la cantidad de dominios que desee. El anonimato de estas prácticas está absolutamente garantizado por las características de la Web e Internet.

Como conclusión de esta discusión, es posible diseñar k páginas orientadas a k claves distintas, esta vez en **k sitios diferentes,** como evolución a la opción anterior consistente en ubicar k páginas en el mismo sitio.

Marchiori parece no tener en cuenta esta posibilidad, ya que el esquema del único puntero que propone, hace posible la PMB para una sola clave por sitio.

Observemos igualmente el esquema que describe.

Se intenta conseguir la seguridad del enfoque del único puntero usando un posprocesador. Primero, se debe asegurar que por cada objeto web, la PMB sea posible con una clave nada más. Una solución es integrar la función de puntaje principal con la de porcentajes en una combinación lineal

$$\alpha \,.\, \text{PUNTAJE} + \beta \,.\, \text{PORCENTAJE} \; (0 \leq \alpha \leq 1, 0 \leq \beta \leq 1, \alpha + \beta = 1)$$

Ahora hay que asegurar que la PMB sea posible para una clave sola pero para todos los objetos de un mismo sitio. Se logra con otro posprocesador que penaliza a todos menos uno de los objetos del mismo sitio con puntaje máximo. Empíricamente, la chance de penalizar un objeto no PMB es menor al 0.1%, y puede bajar si elegimos al azar qué objetos penalizar.

El componente de porcentaje en la función de puntaje mejora la seguridad del objeto web respecto de la PMB de repetición, al no garantizar un puntaje alto por "alta frecuencia". Si se aumenta el puntaje de una clave repitiéndola, necesariamente el puntaje de las otras claves **debe descender**.

4.2.6. El enfoque Híper

Este último enfoque revierte las estrategias vistas hasta ahora en contra de la PMB. Funciona mejor si sus detalles son **públicos**, dado que promueve la PMB en lugar de limitarla.

La idea es que un objeto web, para obtener un puntaje alto, debe promocionar a los objetos competidores. Esto ubica a la PMB en un dilema: si no publicitan a la competencia tienen puntaje bajo; pero si alcanzan un alto puntaje, sus competidores también lo harán indirectamente.

Hay dos polos opuestos de atracción: nadie publicita a nadie, o todos a todos. Como es obvio, este último es el más fuerte. Entonces, todos harán una fuerte PMB, lo que ofrecerá al usuario un espectro completo del mercado, visto desde todos los competidores.

La Web entonces se redimensionaría, aumentando su conectividad gracias a la PMB. (Recordemos que cerca del 80% no tiene enlaces a otros sitios web.)

Un posprocesador posible agregaría a la función principal de puntaje, el componente de la "hiperinformación" (HI), que describe cuánta promoción hace un objeto web a su competencia. Los MBs pueden ocultar su PUNTAJE; pero deberían hacer público que usan la HI, sin dar detalles de su implementación.

Originalmente pensada para mejorar las funciones de puntaje (como ya se vio en otra sección sobre [MAR/97]), la HI se aplica también en seguridad.

Recordemos brevemente cómo funciona la HI. La HI tiene en cuenta a los enlaces de un objeto web A, es decir los PUNTAJEs de los objetos apuntados por A. Como se estudió en su momento, la idea es incluir en el PUNTAJE de A la información potencial que se obtenga desde él a través de sus enlaces.

La influencia de la información de un objeto B en otro A disminuye exponencialmente con la distancia k, en enlaces, de A a B. La contribución de B a la HI de A será de $\mathbf{F}^k$. PUNTAJE(B), con $0 < \mathbf{F} < 1$.

Si en A vemos enlaces B_1, B_2,..., B_n en orden decreciente de importancia informativa ($PUNTAJE(B_1) \geq PUNTAJE(B_2) \geq ... \geq PUNTAJE(B_n)$),

Entonces aportarán:

$$HI(A) = \mathbf{F} \cdot PUNTAJE(B_1) + \mathbf{F}^2 \cdot PUNTAJE(B_2) + ... + \mathbf{F}^n \cdot PUNTAJE(B_n) \leq \mathbf{F} / (\mathbf{F}-1)$$

Se deben diferenciar los enlaces internos, a objetos del mismo sitio, de los externos, que apuntan a un sitio diferente. Los internos son manipulables para PMB; los externos tienen menos probabilidad de ser manipulados (aquí se repite la argumentación de la sección anterior, la probabilidad de manipular enlaces externos es igualmente altísima, porque disponer de varios dominios es muy simple y fácil).

Con las salvedades apuntadas, el ignorar los enlaces internos mejora la medida de HI y la hace más rápida.

Se deben ignorar también los enlaces duplicados dentro de un mismo objeto, de los cuales los autorreferenciales (#) son un caso especial. También hay que descartar enlaces a imágenes, sonido, etc. Los enlaces de trama (FRAME) deben ser expandidos automáticamente.

Algunas pruebas hechas con Excite, HotBot, Lycos, WebCrawler y OpenText muestran un éxito frente a la PMB de 80-85%, sin por ello empeorar sino más bien mejorar la función de PUNTAJE. Con este último fin fue pensada la HI en su concepción.

4.3. Susceptibilidad de los algoritmos de *ranking* a la PMB

Hagamos una recopilación de las prácticas PMB:

- Repetición de claves en un sector o en toda la página (***spamdexing***).
- **Texto invisible** (camuflado con el fondo, fuera del área visible en un navegador, en tags o atributos HTML no visibles, muy pequeño).
- Contenido no relacionado con las consultas: los autores usan las palabras de las **consultas más candentes**, en páginas que no tienen contenido relacionado; y contienen enlaces a páginas con contenido irrelevante.

- Páginas **repetidas** desde sitios diferentes.
- Aprovechamiento del uso de **Direct Hit** en algunos MB. El servicio de DirectHit asigna mayor relevancia a las páginas que los usuarios eligen entre los resultados a sus consultas, teniendo en cuenta cuestiones como el tiempo en que permanecen en ellas. Los autores de las páginas simplemente eligen sus propias páginas una y otra vez para sesgar los futuros resultados a su favor.
- El envío a los MB de páginas diseñadas para la PMB, mientras el autor tiene en su web una versión "normal" de las páginas.
- Mantenimiento simultáneo de **diferentes sitios** que se brindan relevancia mutuamente. El objetivo es el dar una idea de **popularidad falsa** de las páginas.
- Páginas de entrada (***door pages***) sin contenido pero apuntando a otra página objetivo.
- Envío a los MB de muchas páginas del mismo o distinto dominios, todas con las mismas palabras importantes, y todas con un mismo enlace a la verdadera página que se quiere exponer. Este método evitaría las técnicas antiPMB de envío de muchas páginas del mismo dominio (Altavista expulsó a sitios por esto).
- El uso de **páginas relevantes** de otros sitios que aparecen en el tope de los resultados. La PMB consiste en este caso en copiarse la página en el sitio propio y enviarla al MB.

Google parece ser inmune al spamdexing por la manera en que cuenta las apariciones de los términos, acotando su influencia excesiva. Además utiliza PageRank, que disminuye aun más la influencia excesiva de páginas carentes de importancia.

Google tiene un algoritmo que elimina las páginas duplicadas o muy similares de los resultados a sus consultas, por lo que evita la multiplicación de la exposición de contenido duplicado.

PageRank, además es inmune a la popularidad falsa y al envío de páginas tope como propias. La manera en que utiliza la información de los enlaces impide que los sitios se autoasignen importancia excesiva.

La HiperBúsqueda es inmune también a la popularidad falsa ya que no tiene en cuenta los retroenlaces, pues toma su información de los enlaces salientes. Será resistente al *spamdexing* si la función INFOTEXTO es a su vez inmune. Sin embargo, no es inmune a las copias de una página exitosa en el sitio que ejerce la PMB.

Como se adelantó en el capítulo anterior, explicaremos lo expuesto en [HEN/98], donde se señala que HITS no resiste bien en todos los casos a la PMB. La razón fundamental reside en las **relaciones mutuamente recíprocas entre diferentes sitios**. A veces un conjunto de documentos de un sitio apunta a un solo documento en un segundo sitio. Esto eleva el puntaje de concentrador (*hub*) para los documentos en el primer sitio, y el puntaje de autoridad para el documento en el segundo sitio. El caso inverso, en que existe un solo documento en un primer sitio apuntando a múltiples documentos en un segundo sitio, genera el mismo problema. Como la suposición es que el conjunto de documentos de ambos sitios fue creado por el mismo autor u organización, estas situaciones le brindan un peso indebido a la opinión de ese único "autor".

Ya se enfatizó en este capítulo, la imposibilidad de distinguir este tipo de prácticas entre dominios o sitios distintos. La conclusión natural es que al no poder detectar el problema, es imposible tomar medidas específicas para evitar su influencia en las funciones de *ranking*, a menos que la **función de ranking en sí sea resistente** a esta manipulación que llamaremos **multisitio**.

Las diversas variantes de PageRank tienen una propiedad extremadamente importante: son virtualmente inmunes a la manipulación por intereses comerciales. Para obtener un alto PageRank, una página web debe convencer a una página suficientemente importante, o a muchas páginas no importantes para que la apunten. En el peor de los casos, se puede sufrir una manipulación en la forma de avisos comerciales (enlaces) en sitios importantes. Pero esto parece estar bajo un control razonable porque cuesta dinero.

Concentrémonos pues en la fórmula de PageRank. Un compromiso entre los dos extremos de un vector E con pesos uniformes y un E con peso en una sola página, consiste en tomar E con pesos en todas las páginas del directorio raíz de todos los servidores web. Esto podría provocar alguna manipulación de los PageRank. Alguien que deseara manipular este sistema podría simplemente crear un gran número de servidores con páginas en sus directorios raíz apuntando a un sitio particular.

Vemos entonces que inclusive estas funciones extremadamente resistentes a la manipulación tienen variantes que las hacen atacables, y por lo tanto manipulables.

Por último, señalemos una medida que tomó AltaVista para combatir a la PMB en el momento del envío (*submission*) de las páginas. Este MB pide un código de acceso para poder enviar páginas, para su posterior indexado en la base de datos. Este código se presenta en una imagen que contiene caracteres deformados y en posiciones irregulares, por lo que es virtualmente imposible para una aplicación de reconocimiento de caracteres (*OCR* por *optical character recognition*) descubrir el mismo.

Además, una vez ingresado el código, se pueden enviar hasta 5 páginas consecutivas. Para enviar más se debe ingresar un nuevo código, lo que hace dificultoso el envío de páginas por medio de programas automáticos. Este tipo de programas es muy utilizado por las empresas que se dedican al posicionamiento en los MB.

Capítulo 5. Experimentos

En la Introducción, se expusieron las razones más importantes por las cuales tiene sentido realizar el análisis de las funciones de *ranking* en un SRI de la web.

Existen estudios realizados en este sentido como son los casos de [PRI/98] y [LIU/00].

En el primero, se infiere el comportamiento de Altavista, Lycos, Excite e Infoseek tomando como modelo a los árboles de decisión. A partir de allí, se ve la importancia relativa que cada buscador le asigna a los atributos de las páginas (title, meta tags, header, longitud del file, frecuencia de aparición de keywords, estructura HTML) a la hora de hacer el *ranking*. También se utiliza la regresión lineal para inferir el comportamiento de Infoseek y se obtienen resultados similares.

En [LIU/00], mediante consultas específicas se aproximan las fórmulas de pesaje de términos en consultas y documentos, y las constantes intervinientes en tales fórmulas.

En el presente capítulo, se aborda esta cuestión desde dos técnicas o métodos diferentes, cuyo diseño e instanciación en experimentos se presenta en la sección respectiva.

En el primer método, que llamaremos de **hipótesis**, se trata de entender cómo influyen diversos aspectos de una página web en la relevancia que obtiene, ante una consulta de usuario en un dado MB. Así es que se diseñan *ad hoc* diversas páginas para observar cómo se ubican relativamente en los resultados de una misma consulta. Podemos decir que se intentan validar diversas hipótesis acerca de qué aspectos de una página son relativamente más importantes en términos de su influencia en el puntaje que reciben.

El segundo método es, en realidad, un sistema desarrollado íntegramente para esta tesis, que llamaremos EVALSE, por *EValuation of Search Engines* (Evaluación de Motores de Búsqueda). Su objetivo es comprender las funciones de *ranking* de los MBs, a través de la construcción de *rankings* propios y su posterior comparación con los *rankings* de los MBs ante consultas dadas. Sin entrar, por ahora, en el detalle de su funcionamiento, diremos que este sistema es el aporte más importante del presente trabajo, por su aplicabilidad, extensibilidad y originalidad.

5.1. Análisis de aspectos aislados

Primero presentamos en qué consiste esta técnica o método, y luego describimos el experimento que se realizó.

Este método apunta a comprender la influencia de diferentes aspectos de una página, en el cálculo de su relevancia respecto de consultas en un buscador.

Por dar un ejemplo, para la consulta 'heladeria' se quiere ver si AltaVista asigna mayor importancia a una página por tener 'heladeria' en el tag TITLE, respecto de otra página que la tiene sólo en el tag BODY. O también, si dadas dos páginas con 'heladeria' en el BODY, la de menor tamaño es más relevante. O quizás, si dos páginas son similares en contenido pero corresponden a diferentes URLs, qué es lo que determina que una sea más relevante: su popularidad, su cercanía a la raíz del árbol de directorios del dominio (*home directory*), la aparición de la palabra en la URL, etc.

Este tipo de hipótesis o preguntas pueden responderse si se confeccionan cuidadosamente **lotes de páginas**, cada uno asociado a una consulta.

Los lotes deben ser similares, poseer la misma cantidad de páginas y donde cada una de ellas tiene un rol definido, en el contexto del lote, para la resolución de una o más hipótesis.

En dos lotes, las páginas con el mismo rol se diferencian solamente en la palabra clave. Por ejemplo, en el lote de 'heladeria', la página exptitle.html contiene:

<TITLE>heladeria</TITLE>

y en el lote de 'retoce':

<TITLE>retoce</TITLE>

Se debe intentar que las páginas tengan una similitud estructural (cantidad de palabras, tags) cambiando solamente el atributo a medir, para aislar lo que varía de página en página.

Luego de armar los lotes, se los debe enviar al buscador objeto de análisis para su agregado al índice del mismo. El objetivo es que el MB indexe la mayor cantidad de páginas posible de cada lote, para observar cómo quedan relativamente posicionadas en el *ranking* resultado de la consulta correspondiente a tal lote.

Lo que se busca es sacar conclusiones globales para un buscador, a partir de la observación de las consultas específicas que se realicen. Para ello, se hace la suposición razonable de que, en un instante dado, la función de similitud entre una página HTML y una consulta de una palabra (no frase), es la misma para toda página y toda consulta de una palabra. Nótese que esta suposición permite que varíe el valor devuelto por la función, **pero no** la función en sí misma.

Los algoritmos internos y por ende los criterios de relevancia pueden variar en el tiempo, por lo que los resultados de las consultas deben ser analizados al unísono si se desea arribar a conclusiones válidas.

Yendo al experimento en sí, se eligieron 5 palabras-consultas: 'heladeria', 'helycopter', 'acogotado', 'bravuconada' y 'retoce'; a realizar en el buscador AltaVista. Cada lote consta de 28 páginas.

Estos lotes se intentaron agregar al índice de AltaVista, mediante el servicio de *submission* de páginas que se ofrece en el sitio del buscador [AV-submit]. El indexado está sujeto a las restricciones y los plazos del buscador, por lo que no se asegura su agregado, ni su permanencia en el índice. Además, el hecho de que una página se encuentre en el índice, no asegura su aparición en el resultado de la consulta asociada. Puede pasar, por ejemplo, que de

28 páginas de un lote, se indexen 18, y al hacer la consulta aparezcan 10 (o incluso ninguna como se discute abajo). Más adelante, el buscador puede decidir desindexar algunas páginas.

Todas estas cuestiones dificultan el análisis; por lo que se intentó, en lo posible, sortear estos obstáculos. Por ejemplo, se enviaron lotes desde dos dominios distintos para esquivar un potencial tope de páginas por dominio.

Otro recurso es el de utilizar la sintaxis de consultas de AltaVista para obtener más páginas de un lote. La consulta '*<palabra>* +url:*<dominio>*' significa que de las páginas devueltas para la consulta '*<palabra>*' se toman sólo aquéllas cuya URL contiene el string *<dominio>*, pero **manteniendo el orden** de relevancia. Se comprobó que usando esta técnica, se obtuvieron páginas que no aparecían ni siquiera entre los primeros 1000 resultados de las consultas. Por ejemplo, en un momento dado, 'heladeria' devolvió sólo 1 página del lote entre las primeras 1000; mientras que 'heladeria +url:soteica' devolvió 3 páginas del lote.

A continuación enumeramos las hipótesis que se intentan verificar, junto con las páginas de un lote tipo que intervienen en cada caso. Luego se muestra una tabla con las páginas de un lote tipo. Más adelante, se detallan los resultados obtenidos y las conclusiones a las que se puede arribar.

5.1.1. Hipótesis a verificar

En lo sucesivo, la notación p1 < p2 es una abreviatura de posición(p1) < posición(p2). La posición de una página en el ranking de una consulta q en un MB. Significa que la página p1 tiene una menor posición en el ranking que p2. Es decir, la página p1 está más cerca del tope de los resultados que p2.

Se podría pensar que esto significa que p1 es más relevante que p2 para la consulta q, pero dado que no conocemos los puntajes asignados a p1 y p2 por el MB para dicha consulta, no sabemos si el puntaje de p1 es mayor al de p2, o si en realidad sus puntajes son iguales.

Es decir,

$$pos(p1) < pos(p2) \rightarrow relevancia(p1) >= relevancia(p2).$$

Y su contrarrecíproco (notar que dos posiciones no pueden ser iguales):

$$relevancia(p1) < relevancia(p2) \rightarrow pos(p1) > pos(p2).$$

Lo deseable con este método es observar la relación entre relevancias o puntajes, pero en general disponemos sólo de las posiciones.

Hipótesis	**Descripción**	**Páginas involucradas y relación buscada**
Tags	La aparición en un tag tiene mayor relevancia que en otro tag.	Todas las que tienen exactamente 1 (una) aparición en un tag.
Descripción de un hipervínculo	A más apariciones en el texto del tag A, mayor relevancia.	explink2 < explink1
Popularidad	A mayor popularidad, mayor relevancia. (explink1 y explink2 apuntan a exppopu1, pero nadie apunta a	exppopu1 < expsino1

	expsino1).	
Popularidad de los hipervínculos	Si apunto a páginas más populares (más enlaces entrantes en el índice), tendré mayor relevancia. (exppopu2 apunta a AltaVista, y explink1 a exppopu1).	exppopu2 < explink1
Profundidad en el website	A menor profundidad en el sitio, mayor relevancia.	expsino1 < expprofu
Cantidad o frecuencia de aparición	A más apariciones en el texto del tag BODY, mayor relevancia.	expfrec4 < expfrec3 < expfrec2 < expbody1
Sinónimos o cercanía conceptual	La aparición de términos semánticamente relacionados con la consulta mejora la relevancia.	expsino2 < expsino1
Porcentaje *(frecuencia/tamaño)*	A igual cantidad de apariciones, la página de menor tamaño tiene mayor relevancia.	expbody1 < expsino1
Frames	Las páginas con frames son indexadas.	expframe
Descripción de imagen	A más apariciones en tag IMG atributo ALT, mayor relevancia. (si se tiene en cuenta la aparición en IMG-ALT al indexar).	expalt2 < expalt1
Nombre de imagen	A más apariciones en tag IMG atributo SRC, mayor relevancia. (si se tiene en cuenta la aparición en IMG-SRC al indexar).	expimag2 < expimag1
Ubicación en BODY	Cuanto más cerca del tope del tag BODY está la aparición, mejor.	expbody1 < expbody2 < expbody3
Tamaño del font	A mayor tamaño del fuente (tag FONT atributo SIZE), mayor relevancia.	expfont2 < expfont1

5.1.2. Lote típico de páginas y páginas antagonistas

En este directorio se puede ver un lote ejemplo.

Página	**Ubicación de keyword (Tag–Atributo)**	**Versus**	**Hipótesis**
<keyword>.html	Nombre de la página HTML	Otros tags	*Tags*
exptitle.html	Title	Otros tags	*Tags*
expkeyw.html	meta keywords	Otros tags	*Tags*
expclas.html	meta classification	Otros tags	*Tags*
expdesc.html	meta description	Otros tags	*Tags*
exph1.html	h1	Otros tags	*Tags*
exph2.html	h2	Otros tags	*Tags*
expframe.html	frame name target src noframes (4 veces en total)	-	*Frames*
expcom.html	comentario <!-- -->	Otros tags	*Tags*
expalt1.html	img alt	Otros tags y expalt2.html	*Tags* y *Descripción de imagen*
expalt2.html	img alt (2 tags)	expalt1.html	*Descripción de imagen*
expbody1.html	body (primera línea)	Otros tags y expbody?.html	*Tags, Frecuencia de aparición y Ubicación en BODY*
expbody2.html	body (segunda línea)	expbody?.html	*Ubicación en BODY*
expbody3.html	body (tercera línea)	expbody?.html	*Ubicación en BODY*
expfont1.html	font size=+2	Otros tags y expfont2.html	*Tags* *Tamaño del font*
expfont2.html	font size=+4	expfont1.html	*Tamaño del font*
expfrec2.html	body (2 veces)	Otros tags y expfrec3.html	*Tags y Frecuencia de aparición*
expfrec3.html	body (3 veces)	expfrec?.html	*Frecuencia de aparición*
expfrec4.html	body (4 veces)	expfrec?.html	*Frecuencia de aparición*
expimag1.html	img src	Otros tags y expimag2.html	*Tags* *Nombre de imagen*
expimag2.html	img src (2 tags)	expimag1.html	*Nombre de imagen*
explink1.html	a	Otros tags y explink2.html Link a exppopu1.html	*Tags* *Descr. Hipervínculo* *Popularidad hipervínculos*

explink2.html	a (2 tags)	explink1.html Link a exppopu1.html	*Descr. Hipervínculo*
exppopu1.html	body	expsino1.html	*Popularidad*
exppopu2.html	a	explink1.html Link a AltaVista	*Popularidad hipervínculos*
expsino1.html	body	expsino2.html expprofu.html exppopu1.html expbody1.html	*Sinónimos* *Profundidad en website* *Popularidad* *Porcentaje*
expsino2.html	body (con sinónimos)	expsino1.html	*Sinónimos*
expprofu.html	body	expsino1.html	*Profundidad en website*

5.1.3. Posiciones resultantes en varias consultas

Como ya fue destacado, todas las páginas (incluso las que no aparecen) en los rankings, fueron enviadas a AltaVista para agregarlas al índice. Su ausencia puede deberse a diversos motivos: rechazo, posicionamiento demasiado pobre (AltaVista puede limitar sus resultados a 200 páginas), descarte momentáneo para acelerar la respuesta en horas pico [AV-trunc], etc.

Se analizan dos puntos en el tiempo (2/2000 y 11/2000), que coinciden, por un lado, con picos en la cantidad de páginas por ranking; y por otro, con dos versiones distintas de relevancia de AltaVista. Entre dichos momentos evidentemente se produjo un cambio en las funciones de cálculo de relevancias, pues cambió el orden de las páginas para una misma consulta.

La siguiente tabla muestra la cantidad de páginas indexadas por cada fecha y consulta.

Palabra	**12/1999**	**2/2000**	**11/2000**	**4/2001**
heladeria	8	11	6	2
helycopter	-	25	25	18
acogotado	-	-	-	3
bravuconada	-	-	-	3
retoce	-	-	-	5

A continuación se reportan los rankings en cada consulta y fecha.

Consulta		heladeria +url:soteica	
MB		AltaVista	
Páginas enviadas		28 páginas.	
Comentarios Este ranking ha variado en el tiempo, quizás por cambios internos en las funciones de clasificación, que por ejemplo descartaron las páginas con la palabra 'heladeria' en el tag meta.			
12/1999	**2/2000**	**11/2000**	**4/2001**
1. heladeria.html	1. exptitle.html	1. exptitle	1. exppopu2

2. exptitle.html 3. expkeyw.html 4. expdesc.html 5. exph1.html 6. exph2.html 7. expalt1.html 8. expalt2.html	2. expfrec3.html 3. exppopu2.html 4. expalt2.html 5. heladeria.html 6. exph1.html 7. expkeyw.html 8. expdesc.html 9. exph2.html 10. expbody3.html 11. expalt1.html	2. *(fuera de lote)* 3. expfrec3 4. *(fuera de lote)* 5. exph1 6. exppopu2 7. expbody3 8. exph2	2. *(fuera lote)* 3. exph2

Consulta	helycopter +url:chao		
MB	AltaVista		
Páginas enviadas	28 páginas.		
12/1999	**2/2000**	**11/2000**	**4/2001**
-	1. exptitle.html 2. expframe.html 3. expsino2.html 4. expfrec2.html 5. expimag2.html 6. expfrec3.html 7. explink2.html 8. expfrec4.html 9. expalt2.html 10. exph2.html 11. expfont2.html 12. exppopu2.html 13. expfont1.html 14. expsino1.html 15. exph1.html 16. expimag1.html 17. dir/expprofu.html 18. helycopter.html 19. expbody2.html 20. expdesc.html 21. expkeyw.html 22. expbody1.html 23. explink1.html 24. expbody3.html 25. expalt1.html	1. exptitle 2. expframe 3. expfrec2 4. expimag2 5. expfrec3 6. explink2 7. expfrec4 8. expalt2 9. dir/expprofu 10. expsino2 11. exph2 12. expfont2 13. exppopu2 14. expfont1 15. expsino1 16. exph1 17. expimag1 18. helycopter 19. expbody2 20. expdesc 21. expkeyw 22. expbody1 23. explink1 24. expbody3 25. expalt1	1. exptitle 2. exppopu2 3. expalt2 4. expprofu 5. expfrec4 6. expfrec3 7. expfrec2 8. expsino1 9. exph1 10. expimag1 11. helycopter 12. expfont1 13. expbody2 14. expbody3 15. expsino2 16. expfont2 17. expkeyw 18. explink1

Por completitud, se reportan los restantes rankings, aunque no participen del análisis por la escasez de páginas indexadas.

Consulta:	acogotado +url:soteica
MB:	AltaVista.
Páginas enviadas:	28 páginas.
Fecha:	4/2001.
1. expdesc 2. expbody1 3. expkeyw	

Consulta:	bravuconada +url:soteica
MB:	AltaVista.
Páginas enviadas:	28 páginas.
Fecha:	4/2001.
1. expclas	
2. expfrec4	
3. expdesc	

Consulta:	retoce +url:soteica
MB:	AltaVista.
Páginas enviadas:	28 páginas.
Fecha:	4/2001.
1. exptitle	
2. expkeyw	
3. expbody3	
4. expdesc	
5. expsino2	

5.1.4. Comparación de rankings

Las siguientes 2 tablas comparan los rankings de 'heladeria' y 'helycopter' en las dos fechas analizadas. Intentan dar una medida de la cercanía entre estos dos rankings, teniendo en cuenta sólo las páginas que aparecen en 'heladeria'.

5.1.4.1. Fecha: 2/2000

Para cada página se informa la posición que tendría en cada ranking. Están ordenadas en forma ascendente por la columna heladeria(11pp), ya que aparecen en ese orden las 11 páginas para esa consulta en esta fecha.

En helycopter(25pp) aparece la posición de cada página en el ranking en esta fecha. En helycopter(11pp) se calcula la posición que tendría cada página en un supuesto ranking de estas 11 páginas para 'helycopter', con el fin de poder compararlo con heladeria(11pp).

En la última columna, se calcula la diferencia entre las posiciones de las dos columnas previas.

Página	**helycopter (25 pp)**	**helycopter (11 pp)**	**heladeria (11 pp)**	**Diferencia (hely11 - hela11)**
exptitle.html	1	1	1	0
expfrec3.html	6	2	2	0
exppopu2.html	12	5	3	2
expalt2.html	9	3	4	-1
<keyword>.html	18	7	5	2
exph1.html	15	6	6	0
expkeyw.html	21	9	7	2
expdesc.html	20	8	8	0
exph2.html	10	4	9	-5
expbodyres.html	24	10	10	0
expalt1.html	25	11	11	0

La máxima diferencia posible entre las posiciones de dos rankings de 11 páginas es 60, que se da cuando uno es el inverso del otro. La mínima diferencia es 0, cuando los rankings son iguales. Como

en este caso, la suma de los valores absolutos de las diferencias es 12, representa un 12/60 = 1/5 del máximo.

5.1.4.2. Fecha: 11/2000

Página	helycopter (25 pp)	helycopter (6 pp)	heladeria (6 pp)	Diferencia (hely6 – hela6)
exptitle.html	1	1	1	0
expfrec3.html	5	2	2	0
exph1.html	16	5	3	2
exppopu2.html	13	4	4	0
expbodyres.html	24	6	5	1
exph2.html	11	3	6	-3

La máxima diferencia en 6 páginas es 18. Es decir que aquí la diferencia representa 6/18 = 1/3 del máximo.

5.1.5. Verificación de hipótesis

Las entradas con – (guión medio) implican que no se indexó alguna de las páginas intervinientes en la hipótesis en cuestión.

Cuando no se verifica alguna hipótesis (entrada 'NO') se aclara el orden en que aparecen en el *ranking* las páginas de interés para la misma.

Cuando se verifica la relación buscada, la entrada es 'OK'.

5.1.5.1. Fecha: 2/2000

Hipótesis	heladeria (11 pp)	helycopter (25 pp)
Tags	title < popu2 < heladeria < h1 < keyw < desc < h2 < body3 < alt1	title < sino2 < h2 < font2 < popu2 < font1 < sino1 < h1 < imag1 < profu < helycopter < body2 < desc < keyw < body1 < link1 < body3 < alt1
Popularidad Hipervínculos	-	OK
Frecuencia	-	NO frec2 < frec3 < frec4 < body1
Ubicación BODY	-	NO body2 < body1 < body3
Frames	NO	OK
Descripción hipervínculo	-	OK
Popularidad	-	-
Profundidad	-	OK
Sinónimos	-	OK
Porcentaje	-	NO

		sino1 < body1
Descripción imagen	OK	OK
Nombre imagen	-	OK
Font	-	OK

5.1.5.2. Fecha: 11/2000

Hipótesis	**heladeria (6 pp)**	**helycopter (25 pp)**
Tags	title < h1 < popu2 < body3 < h2	title < profu < sino2 < h2 < font2 < popu2 < font1 < sino1 < h1 < imag1 < helycopter < body2 < desc < keyw < body1 < link1 < body3 < alt1
Popularidad Hipervínculos	-	OK
Frecuencia	-	NO frec2 < frec3 < frec4 < body1
Ubicación BODY	-	NO body2 < body1 < body3
Frames	NO	OK
Descripción hipervínculo	-	OK
Popularidad	-	-
Profundidad	-	NO profu < sino1
Sinónimos	-	OK
Porcentaje	-	NO sino1 < body1
Descripción imagen	-	OK
Nombre imagen	-	OK
Font	-	OK

5.1.6. Conclusiones del experimento

El proceso de *submission* de páginas a AltaVista es aparentemente simple, ya que sólo implica el ingreso de URLs en un formulario web. Sin embargo, hemos visto que las políticas de aceptación de páginas por parte de este buscador son complejas, y parecen variar según las palabras que contengan los documentos enviados. Para una palabra o consulta como 'heladeria', siempre se indexaron menos páginas que para 'helycopter'. Este hecho puede ser atribuido (sin total certeza) a factores como la cantidad de páginas competidoras, que son muchas más en la

primera consulta (~2000) que en la segunda (~80). Esto puede producir la "caída" o borrado de páginas del índice, cuando pasan un umbral mínimo de relevancia que impiden que aparezcan en los resultados para consultas que contengan 'heladeria', el ejemplo de este caso.

Otro factor diferenciante entre estas dos consultas es la cantidad de documentos en los que aparece la palabra-consulta. Según el propio AltaVista, con 'heladeria' parece haber ~2000 páginas y con 'helycopter' ~100. Se puede ver una relación lineal entre estos números (frecuencia en documentos o *document frequency*) y los del párrafo anterior. Sin embargo, la diferencia entre 100 y 80, por ejemplo, nos muestra que no siempre una página que contiene 'helycopter' aparece en el resultado de la consulta consistente en ese término.

Hay una curva ascendente de agregado de páginas, un pico, y luego un descarte paulatino de las mismas, quizás por el proceso natural de actualización y renovación del índice, que va filtrando aquellas páginas que no se renuevan y/o que llevan mucho tiempo indexadas, a favor de las más recientemente recolectadas con los *crawlers* o recibidas mediante el mencionado proceso de *submission*.

Otra interpretación posible es la existencia de un límite a la cantidad de páginas por dominio, que implica quitar algunas para hacer lugar a otras más recientes para un mismo dominio.

Si se observan los rankings, de manera consistente:

- La página **exptitle** aparece en la primera o segunda posición cuando es indexada. Supera a páginas que llegan a tener hasta 4 apariciones de la palabra clave, como expframe, lo que habla de la extrema importancia relativa del tag TITLE, por sobre la frecuencia del término.
- El sublote con 2 ó más apariciones de la palabra normalmente aparece por sobre el sublote con 1 aparición, lo que nos muestra el peso de la **frecuencia** en la noción de relevancia. Quizás una excepción sea el puesto 3 logrado por expsino2, al cobrar interés los términos semánticamente cercanos a la consulta o coocurrentes en la colección.
- Indudablemente, como se mencionó, el algoritmo de asignación de puntajes de AltaVista cambió de manera ostensible en el tiempo que se analiza.

La comparación de rankings nos muestra que no se puede suponer con certeza que se use la misma función de relevancia para toda consulta de una palabra. Las páginas que fueron indexadas en ambos lotes aparecen en distinto orden en ambos rankings. Si, por el contrario, suponemos que el algoritmo utilizado para asignar relevancia es el mismo, una explicación posible es que a las páginas con relevancia demasiado cercana se las ordenó arbitrariamente.

Nuevamente, factores como la frecuencia en documentos, o la popularidad de la consulta (cantidad de páginas intervinientes en el ranking) también pueden haber influido en estas diferencias observadas.

El proceso de verificación de las hipótesis, en este experimento, no puede abstraerse de lo expuesto acerca de los problemas en la indexación de páginas, y a las diferencias entre los rankings. Es evidente que no se puede validar una hipótesis cuando alguna de las páginas no se indexó, pues sería incorrecto suponer que una página que aparece en el resultado es más relevante que otra que no aparece. Tampoco se pueden extender conclusiones a más consultas cuando ya existen diferencias entre los rankings que fueron observados.

Enfrentamos en este experimento imposibilidades prácticas dependientes mayormente del MB en estudio, tales como los mencionados inconvenientes en la indexación de páginas en

tiempo y forma; y también en las diferencias llamativas entre *rankings* de consultas distintas, tomados en un mismo momento.

Sin embargo, el método de validación de hipótesis debe interpretarse como un **procedimiento útil** para observar tendencias en los *rankings* de cualquier MB de la web.

5.2. EVALSE: evaluación de motores de búsqueda

En esta sección se describe un sistema desarrollado en el marco de la tesis, cuyo objetivo es evaluar motores de búsqueda, o en general cualquier SRI.

Cuando hablamos de evaluar, lo hacemos en línea con la terminología utilizada en la literatura concerniente a las TREC, y en particular, del sistema de recuperación de la información llamado SMART [SMART], que incluye un módulo de evaluación de otros SRIs.

Normalmente, en el marco de las TREC, se han publicado hasta ahora evaluaciones de funciones de relevancia respecto de una colección conocida, acotada, con respuestas conocidas a consultas conocidas. La analogía es que mientras en las TREC las respuestas las brindan expertos en las consultas y la colección; con EVALSE las respuestas las brinda el MB, al que consideramos el "experto" en su colección.

Una evaluación es el resultado de comparar dos *rankings* de documentos, obtenidos mediante dos funciones de asignación de relevancia diferentes. La distancia entre dos *rankings* se puede medir de diversas maneras, entre ellas precisión-recuperación (SMART) o la función Gamma de Goodman-Kruskal, que es adoptada, con modificaciones, por EVALSE.

Sabemos que un *ranking* de páginas es el resultado de asignar un puntaje a cada una de ellas mediante una función de asignación de relevancia. Por lo tanto, al evaluar un ranking de un MB contra otro ranking local confeccionado por el sistema evaluador, lo que se está realizando indirectamente es un análisis de cuán cercanas son las funciones de relevancia de ambos sistemas.

La evaluación, entonces, es un medio para analizar o **estudiar los criterios de relevancia** adoptados por un SRI, a través del estudio de los *rankings* que tal SRI arroja ante determinadas consultas.

En el caso de los MB de la web, podemos obtener un *ranking* mediante la interfaz pública, tras realizar una consulta. A partir de la secuencia de URLs obtenida, podemos recolectar las páginas correspondientes a dichas URLs y guardarlas en almacenamiento secundario para su procesamiento. Luego, se pueden producir índices de esas páginas, según un algoritmo de indexación definido. Tomando la consulta y los índices, se puede asignar un puntaje a cada página obtenida, según un algoritmo de asignación de relevancia. Así, obtenemos un *ranking* local para esa consulta, que ahora podemos usar para evaluar el ranking original proveniente del MB.

Eso es lo que hace básicamente el sistema presentado en esta sección.

EVALSE es un sistema de recolección de páginas, indexación, asignación de relevancia a las mismas respecto de una consulta, y posterior comparación del *ranking* obtenido con el *ranking* del MB para esa misma consulta.

Se utilizan las N primeras páginas resultado de una consulta en un MB, para generar R rankings locales según R esquemas diferentes de relevancia basados, por ejemplo, en un modelo de RI como el vectorial. Luego se evalúan esos R rankings respecto del ranking del MB en cuestión.

Así, es posible analizar diferentes variantes de indexación o de pesaje de términos en documentos y consulta. Para arribar a conclusiones generales acerca de un MB, es menester realizar evaluaciones para Q consultas diferentes.

Veamos un ejemplo de lo que puede hacer el sistema, y que iremos ampliando progresivamente. Tomemos las primeras 200 URLs de una consulta como "+backdoor" en el buscador AltaVista. Sabemos cómo ordenó AltaVista a dichas páginas, pero desconocemos el

puntaje exacto que les asignó. Tras recolectar las 200 páginas desde sus respectivos servidores web indicados en las URLs, EVALSE construye, digamos, 5 índices según sendas variantes de indexación, que dependen, supongamos, de si aplicamos o no *stemming* o de si eliminamos o no las *stopwords* o palabras vacías.

Ahora, construye con esas 200 páginas indexadas de 5 maneras distintas, sus propios *rankings* locales, según 256 esquemas de relevancia por ejemplo. Entonces tenemos por cada variante de indexación, 256 rankings locales. En total, 256 x 5 *rankings* para la consulta "+backdoor" solamente.

Se calculan entonces 256 x 5 evaluaciones que resultan de comparar cada *ranking* local con el obtenido con "+backdoor" en AltaVista. Esta comparación se basa solamente en las posiciones de las páginas y no en los puntajes, ya que como vimos, no disponemos de tal información en AltaVista.

Ahora podemos analizar las evaluaciones para descubrir qué esquema de relevancia se acercó más a la función de *ranking* de AltaVista; o qué variante de indexación resultó mejor evaluada; o cómo influye en las evaluaciones el usar o no *stemming*; o quizás analizar en detalle la influencia de cada componente del esquema de relevancia utilizado.

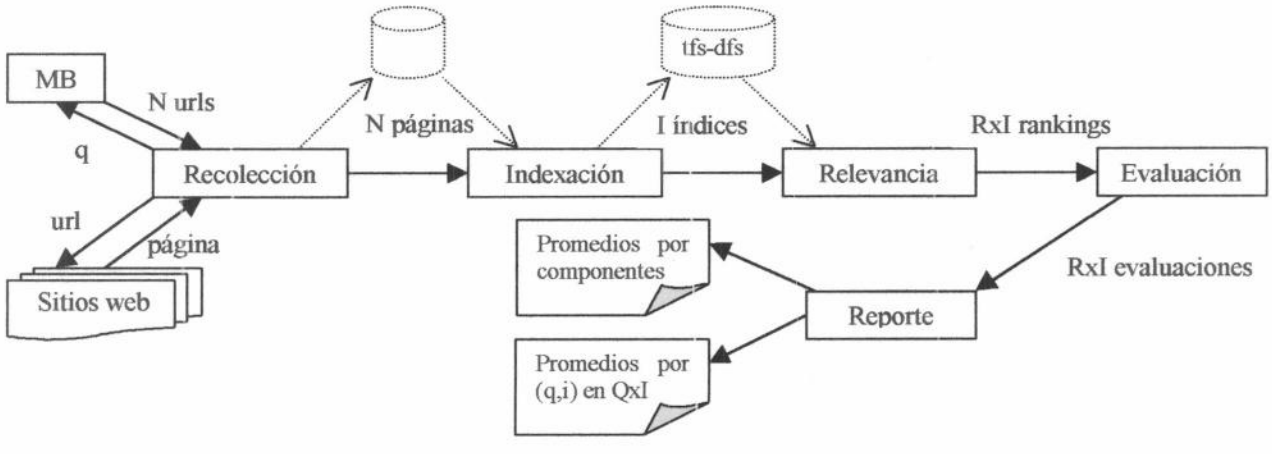
Si utilizamos por ejemplo, 30 consultas en un MB, podremos observar tendencias acerca de la función de relevancia del mismo, teniendo como apoyo las 256 x 5 x 30 evaluaciones realizadas por EVALSE.

Resumiendo, en general, se calculan R x I x Q evaluaciones para Q consultas, I variantes de indexación y R esquemas de relevancia. Toma las N primeras páginas de cada una de las Q consultas para un mismo MB.

Si tomamos las mismas Q consultas en dos o más MBs, podremos ampliar o extender nuestro análisis a la comparación de evaluaciones en todos ellos.

5.2.1. Diseño del sistema

Habiendo presentado el mecanismo con el que EVALSE aproxima a las funciones de ranking de cualquier SRI, proseguimos con el esquema modular-funcional del sistema, para una consulta q dada.



- Recolección: obtiene de la web las N primeras páginas resultado de Q consultas objeto de análisis por cada MB.

- Indexación: se procesan las páginas obtenidas, para producir los términos representativos de cada una de ellas. Se calculan la frecuencia de cada término en cada documento (tf) y la frecuencia de cada término entre las N páginas de la consulta (df). Se produce esta información para las I variantes de indexación.
- Relevancia: por cada variante de indexación, y por cada página resultado de cada consulta, se calcula un puntaje **sim(q(i), p(i), e)** en R esquemas de relevancia e. Estos puntajes nos determinan un *ranking* de N páginas por cada variante de indexación i, cada esquema de relevancia e y cada consulta q.
- Evaluación: por cada *ranking* obtenido en el módulo anterior, se calcula una evaluación respecto del *ranking* del motor de búsqueda analizado. Para ello se implementa una función de distancia entre *rankings*.
- Reporte: toma las evaluaciones del módulo anterior, y efectúa reportes de promedios por cada consulta q y variante i de indexación, por ejemplo.

5.2.2. Implementación del sistema

A continuación, se hará la descripción de la implementación concreta que se hizo de cada módulo y las decisiones adoptadas para el experimento realizado con este sistema.

Recolección:

Se analizaron 30 consultas de una palabra en AltaVista (AV) y las mismas 30 consultas en Alltheweb (AW). Se filtraron las consultas para que devolvieran sólo páginas en inglés. Se tomaron los 200 primeros resultados de cada consulta.

Consulta	#Resultados AV	#Resultados AW
Unanticipated	185489	113203
startling	180627	307828
quaint	163836	266167
imaginable	154839	222036
discerning	144833	231588
outperform	136313	117951
summertime	135563	234337
hypocrisy	133353	213335
invoicing	97883	129790
stroud	97408	119456
buxton	91329	108292
mcclain	84238	87129
impasse	81440	117591
whomever	77553	144639
optimizer	76293	92946
backdoor	76057	82811
battleground	67694	119648
dutchess	65627	64642
heros	62198	84376
frustrate	61994	97129
coreldraw	58178	67608
beeper	53169	78963
iplanet	51974	99011
colocation	48283	60304
sparring	45584	81795
proliferate	41845	66783

relaunch	41257	45543
excavator	30008	38980
seagram	29791	41588
komputer	24463	22802
Promedios	**86637,3**	**118609,0333**

Más adelante se justificará teórica y experimentalmente cómo se llega a la elección de estas consultas.

Indexación:

Para indexar las páginas se siguieron los siguientes pasos iniciales (análisis léxico):

1) Extraer el texto de interés del código HTML de cada página (veremos que en una de las variantes implementadas se toma el texto interno de algunos tags).
2) Procesar las vocales con todo tipo de acentos (caracteres internacionales) para normalizarlos en a-e-i-o-u.
3) Pasar el texto a minúsculas.
4) Transformar todo lo que no es [a-z0-9ñÑ] en espacios.
5) Extraer las palabras que quedaron separadas por espacios.
6) Descartar las palabras de una sola letra.

Una vez obtenidos los *tokens* o términos iniciales, se puede:

7) Descartar *stopwords*. Se utiliza una lista de palabras en inglés del sistema SMART.
8) Aplicar *stemming*. Se utiliza una implementación del algoritmo de Porter [POR/80] para el idioma inglés genérico.

A la consulta se la indexa exactamente de la misma manera, por lo que puede consistir ella misma en un documento.

Se implementaron 5 variantes diferentes de indexación de las páginas de cada consulta.

Se pueden distinguir estas variantes por los términos con que representan a las páginas.

La siguiente tabla presenta las 5 formas, y sus características.

Id	**Se aplica *stemming***	**Se quitan *stopwords***	**Se usa texto de tags**
---	No	No	No
--t	No	No	Sí
-st	No	Sí	Sí
s-t	Sí	No	Sí
sst	Sí	Sí	Sí

La tercera columna se refiere al texto de los siguientes pares tag-atributo HTML: *img-alt*, *acronym-title;* y *meta-content* sólo cuando el valor del atributo *meta-name* está en el conjunto {keywords, description, classification, author, copyright, rating}.

Relevancia:

Se implementaron 256 esquemas de asignación de relevancia a términos en documentos y consultas. Son los mismos que implementa el sistema SMART [SAL/83]. Se basan en el modelo RI de espacio vectorial.

Cada esquema o función de relevancia consta de 2 ternas de componentes. La primera terna concierne al peso de los términos en la consulta. La segunda terna rige la función de pesaje de términos en documentos, o páginas en nuestro caso.

Ambas ternas constan de 3 componentes: frecuencia de términos (tf), frecuencia en colección de documentos (df) y normalización.

- El componente **tf** asigna mayor peso a los términos que ocurren con mayor frecuencia en el texto de los documentos (respectivamente consulta).
- El componente **df** asigna mayor peso a los términos que ocurren en pocos documentos de la colección (en nuestro caso, tomamos las 200 páginas de cada consulta como la colección para esa consulta).
- El componente de **normalización** iguala las longitudes de los vectores de documentos de la colección, para no asignar mayor relevancia a los documentos más largos.

La siguiente tabla muestra para cada componente, las variantes de pesos de términos (*term-weighting schemes*) utilizados.

Componente de frecuencia del término (tf)		
b	1.0	Peso binario igual a 1 para los términos presentes en un vector (la frecuencia del término es ignorada).
n	tf	Frecuencia absoluta del término (cantidad de veces que el término aparece en el texto de un documento o consulta).
a	$0.5 + 0.5\frac{tf}{\max tf}$	Frecuencia del término aumentada y normalizada (usa normalización máxima, donde cada tf es dividida por el máximo tf, y luego se normaliza el valor para que quede entre 0.5 y 1.0).
l	$\ln tf + 1.0$	Frecuencia logarítmica del término, lo que reduce la importancia de la frecuencia absoluta en colecciones con gran variedad del largo de los documentos.
Componente de frecuencia en la colección de documentos (df)		
n	1.0	No se cambia el peso; se usa el componente tf original (que puede ser *b, n, a* ó *l*).
t	$\ln\frac{N}{n}$	Multiplicar el componente tf original por un factor inverso a la frecuencia en documentos (N es la cantidad total de documentos en la colección del MB, y n es la cantidad de documentos a los que un término es asignado).
Componente de normalización		
n	1.0	Sin cambios; se usan los factores provenientes de la frecuencia del término y de la frecuencia en la colección solamente (no hay normalización).
c	$\frac{1}{\sqrt{\sum_{vector} w_{ij}^2}}$	Usar normalización del coseno donde cada peso de término w_{ij} es dividido por un factor que representa la norma euclídea del vector de pesos.

Por ejemplo, el esquema ***ltc-lnc***, que brinda una alta efectividad de recuperación para las colecciones de la TREC, usa: normalización de coseno del logaritmo de la frecuencia para los términos de un documento; y normalización de coseno del logaritmo de la frecuencia de los términos, por la frecuencia inversa en documentos, para los términos de la consulta.

Es decir, el peso del término t en el documento d (w_{td}) y en la consulta q (w_{tq}) se calcularían de la siguiente manera:

$$w_{td} = (\ln tf_{td} + 1.0) \cdot \left(\frac{1}{\sqrt{\sum_{vector} w_{id}^2}}\right)$$

$$w_{tq} = (\ln tf_{tq} + 1.0) \cdot \left(\ln \frac{N}{n_t}\right) \cdot \left(\frac{1}{\sqrt{\sum_{vector} w_{iq}^2}}\right)$$

De lo expuesto se desprende que hay 16 (4x2x2) variantes de fórmulas de pesos para términos en documentos. Hay otras 16 para los términos en consultas. Tenemos en total 256 (16x16) esquemas de pesos, sólo con estas fórmulas y estos componentes.

Evaluación:

Este módulo toma los rankings calculados por el módulo Relevancia, y los compara contra el ranking del MB correspondiente a la consulta respectiva.

Para comparar los *rankings* se optó por adaptar la función gamma de Goodman-Kruskal [GOO/54] aunque se podría utilizar la función de distancia entre *rankings* que se prefiera.

La función gamma original compara *rankings* que no necesariamente contienen las mismas páginas. Por el contrario, nuestra implementación (GK') supone que el conjunto de páginas presentes en ambos *rankings* es el mismo. Además, como veremos, GK' trata como un caso especial a las páginas con igual relevancia asignada, algo no contemplado originalmente. Por lo demás, el cálculo de la distancia entre *rankings* respeta el original.

Nuestra implementación calcula la evaluación de la siguiente manera:

- GK': compara un *ranking* local rl con el *ranking* rm del MB para las mismas N páginas.
- Recorre los `pares` $= \binom{N}{2}$ posibles pares (p1, p2) de páginas.
- Observa en qué orden aparece cada par en rl y en rm.
  - Si p1 y p2 tienen el mismo puntaje en rl: `pares_ok += 0.5`.
  - Si aparecen en el mismo orden en ambos *rankings*: `pares_ok += 1;`
  - Si aparecen en orden inverso: `pares_ok += 0`.
- Distancia = 2 * `pares_ok` / `pares` – 1.

Algunos comentarios: la razón por la que se suma 0.5 si p1 y p2 tienen el mismo puntaje en rl obedece a que p1 y p2 podrían aparecer en cualquier orden en rl si nos basamos solamente en su puntaje. Por lo tanto, existe la misma probabilidad de que p1 aparezca posicionada antes que p2 o viceversa, por ello se elige sumar el valor intermedio entre estas dos posibilidades.

Vemos que la distancia resultante será:

- 1 si los dos *rankings* son iguales (y además los puntajes en rl son todos distintos, porque, si no, habría por lo menos un par que sume 0.5 en lugar de 1)
- –1 cuando los *rankings* sean uno el inverso del otro (mismo comentario previo).
- 0 cuando la cantidad de pares en orden coincidente en ambos *rankings*, sea igual a la cantidad de pares en orden inverso. También dará este valor cuando todas las páginas en rl tengan el mismo puntaje. Es evidente también que un rl que ordene sus páginas de manera aleatoria tenderá a devolver este valor 0.

Reporte:

Este módulo dispone de todas las evaluaciones calculadas por Evaluación.

Sabe, para cada consulta en estudio, qué evaluación obtuvo cada uno de los 256 esquemas de relevancia aplicados, y para cada una de las 5 variantes de indexación.

Fundamentalmente, reporta resultados de promedios de evaluación por consulta.

También puede analizar cada componente del esquema de relevancia, y reportar las evaluaciones de cada variante de cada componente, para poder comparar su influencia.

Por ejemplo, se puede ver, para cada forma de indexación, cuál de las variantes {b, n, a, l} del componente tf de términos en consultas, describe mejor el ranking de un MB (i.e. tiene una evaluación superior).

Cuando analicemos los resultados del experimento realizado, veremos la utilidad de la información provista por este módulo.

5.2.3. La elección de las consultas para la evaluación

La elección de las consultas no es trivial, ya que a partir de ellas se pretenden generalizar las observaciones para un dado MB. Lo que se desea es obtener *rankings* de 200 páginas que puedan ser útiles para analizar desde el punto de vista de las evaluaciones realizadas por EVALSE.

Es importante enfatizar que, en esta primera aplicación de la implementación realizada, nos concentraremos en consultas de una sola palabra. Esto no significa que considerar consultas más complejas carezca de sentido o sea irrelevante. Por el contrario, el abordar la evaluación de consultas multipalabra o con frases, o que utilicen operadores o comodines, es una tarea que complementaría lo realizado en este trabajo y enriquecería los resultados obtenidos.

Siempre que la restricción de una palabra por consulta implique una limitación a nuestro análisis, será debidamente aclarado. Estas y otras limitaciones, y las respectivas extensiones a EVALSE serán expuestas en la sección correspondiente, lo que permitirá entender también con mayor claridad el contexto y las posibilidades de aplicación del sistema.

Recordemos que tomamos solamente las primeras 200 páginas de cada consulta, y que no disponemos de los puntajes asignados a las mismas, sino solamente de sus posiciones en el *ranking*.

Tomemos por caso una consulta q1 que devuelve en total más de un millón de páginas, y otra q2 con muchos menos resultados, supongamos cerca de cien mil. Evidentemente, por la cantidad de páginas consideradas, los puntajes asignados por el MB en las primeras 200 páginas de q1 tenderán a ser mejores que los puntajes asignados en q2. En el caso extremo, los puntajes en los resultados de q1 serán cercanos al máximo posible para tal consulta.

Es de esperar que el rango de puntajes en las primeras 200 páginas se achique a medida que aumenta la cantidad de páginas resultado, pues la probabilidad de que haya más páginas con puntajes cercanos al máximo aumenta.

Es razonable suponer entonces que si tomamos consultas sin tantos resultados, quizás unas decenas de miles, las primeras 200 páginas tomen sus puntajes de un rango más amplio. También es esperable que no tengan, en promedio, tan buen puntaje como los de una consulta mucho más numerosa.

En el otro extremo, en consultas con muy pocos resultados (no muchos más que 200), las 200 páginas mejor consideradas quizás poseen puntajes demasiado bajos, y similares entre sí, lo que produce un efecto aplanamiento, pero esta vez hacia puntajes bajos.

Vemos entonces que para aplicar el sistema evaluador, no conviene tomar consultas con demasiados resultados, pero tampoco convienen consultas con demasiado pocos resultados. En estos extremos, la cercanía entre los puntajes de las páginas conspira contra la posibilidad de entender los aspectos específicos de las páginas que producen el ordenamiento resultante.

Para ilustrar este razonamiento, se analizaron las evaluaciones de dos lotes de 15 consultas en el buscador AltaVista. Un lote, que llamaremos de alta frecuencia (*high*), contiene consultas con 1,6 millón de resultados reportados, en promedio; el otro lote, de baja frecuencia (*low1*), arroja en promedio alrededor de 70 mil resultados por consulta.

Lote de alta frecuencia:

Consulta	# Resultados AV
wing	1178505
attorney	2109732
quake	1186074
homestead	1790028
revenue	2284777
qualify	1210508
spotlight	1065327
enterprise	3523972
diagnostic	1453597
secret	1542620
incorporate	1242408
crown	1486815
silicon	1474797
produce	1653075
pride	1000122
Promedio	**1613490,467**

Lote de baja frecuencia:

Consulta	# Resultados AV
startling	180627
proliferate	41845
seagram	29791
frustrate	61994
coreldraw	58178
backdoor	76057
sparring	45584
discerning	144833
summertime	135563
colocation	48283
battleground	67694

excavator	30008
whomever	77553
relaunch	41257
beeper	53169
Promedio	**72829,0667**

A continuación presentamos las evaluaciones de estos dos lotes para las 5 variantes de indexación implementadas (cada valor es el promedio de las 256 evaluaciones para la consulta-index id):

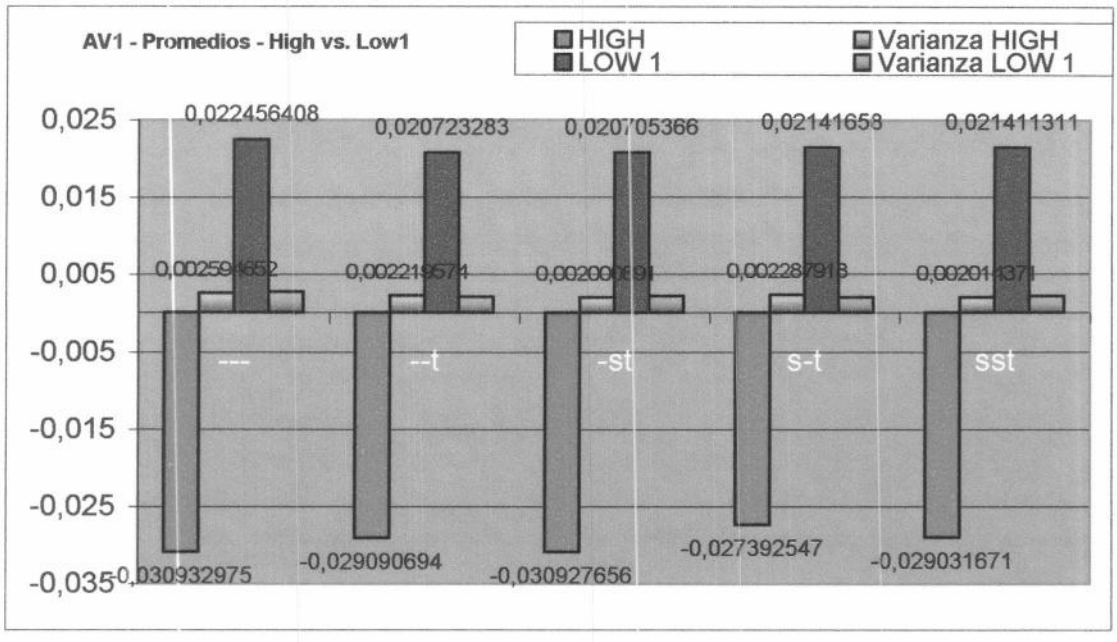
Alta frecuencia:

Consulta	**---**	**--t**	**-st**	**s-t**	**sst**
wing	-0,008269935	-0,006668914	-0,011098806	-0,001135561	-0,003723677
attorney	0,046922959	0,03843232	0,02533763	0,037634285	0,024186618
quake	-0,010435629	-0,023782248	-0,013652627	-0,03272315	-0,025390748
homestead	-0,09537838	-0,067932249	-0,077178079	-0,063128043	-0,070233995
revenue	-0,070637031	-0,043492486	-0,047886412	-0,042788895	-0,047126534
qualify	-0,022590708	-0,052548732	-0,056275516	-0,053555	-0,056958341
spotlight	0,024098661	0,039996897	0,026884985	0,047448456	0,038796011
enterprise	-0,009609281	-0,015973148	-0,018238589	-0,015786676	-0,019785914
diagnostic	0,007409717	-0,00474801	-0,00565752	-0,003478409	-0,007142433
secret	-0,025228187	-0,036652555	-0,035729298	-0,05501261	-0,054156909
incorporate	-0,013203327	-0,016629887	-0,014862187	0,000649282	0,001129676
crown	-0,087985671	-0,076105676	-0,067515156	-0,073859879	-0,067143152
silicon	0,017271351	0,025782613	0,030299002	0,025728076	0,030445572
produce	-0,08923702	-0,057748418	-0,065547284	-0,045174111	-0,050604003
pride	-0,127122147	-0,13828992	-0,13279498	-0,135705969	-0,127767236
PROMEDIO	**-0,030932975**	**-0,029090694**	**-0,030927656**	**-0,027392547**	**-0,029031671**
VARIANZA	**0,002594652**	**0,002219574**	**0,002000891**	**0,002287918**	**0,002014371**

Baja frecuencia:

Consulta	**---**	**--t**	**-st**	**s-t**	**sst**
startling	0,032548126	0,035453957	0,025871046	0,036875211	0,028829647
proliferate	0,028974428	0,019129025	0,028003755	0,015484555	0,02470062
seagram	0,09673357	0,076254629	0,072123533	0,073626106	0,070181564
frustrate	-0,0264029	-0,023937128	-0,020656939	-0,0076225	-0,003640485
coreldraw	-0,004054839	0,005842315	0,004785679	0,011033037	0,010922971
backdoor	-0,070251781	-0,085495601	-0,086595288	-0,093443032	-0,094782222
sparring	0,043245384	0,042150041	0,049231882	0,041922706	0,04812965
discerning	-0,039021597	-0,036696935	-0,04233816	-0,035347131	-0,042194999
summertime	0,064780332	0,078894106	0,076461976	0,081147453	0,080030899
colocation	7,51E-06	-0,00445115	-0,000495406	-0,001508737	0,002199303
battleground	-0,00933584	0,009495962	0,003028404	0,011549708	0,006679894
excavator	-0,018664717	0,045850738	0,030795739	0,04239418	0,024686717
whomever	0,042066263	0,034252793	0,032541899	0,034679547	0,032243172
relaunch	0,082657698	0,075429882	0,083660014	0,070663311	0,077471638
beeper	0,113564487	0,038676614	0,054162354	0,03979429	0,055711287
PROMEDIO	**0,022456408**	**0,020723283**	**0,020705366**	**0,02141658**	**0,021411311**
VARIANZA	**0,00277603**	**0,002059366**	**0,002156076**	**0,002025434**	**0,002136588**

Gráficamente:



Vemos claramente que las evaluaciones para el lote *high* son consistentemente menores que las del lote *low1*. Indudablemente estos dos lotes, representan dos muestras totalmente opuestas de evaluaciones calculadas por el sistema.

Tal como se esperaba, al tomar consultas demasiado numerosas (*high*), las evaluaciones caen respecto de otras consultas con menos resultados (*low1*).

Estas consideraciones nos indican que debemos concentrarnos en consultas de baja frecuencia para observar mejor cómo varían las evaluaciones. Por ello, se agregarán 15 consultas más de baja frecuencia a las 15 recién comparadas, para analizar las 30 consultas en su conjunto.

5.2.4. Los resultados de la aplicación de EVALSE

Mostraremos y analizaremos, en esta sección,, algunos resultados de aplicar el sistema descripto, a las 30 consultas que ya fueron oportunamente enumeradas. Recordemos que el promedio de resultados por consulta en AV es cercano a 85000, y en AW a 120000.

Veremos las evaluaciones de dichas consultas en AltaVista y AlltheWeb, por separado, para luego compararlas entre sí, y sacar algunas conclusiones.

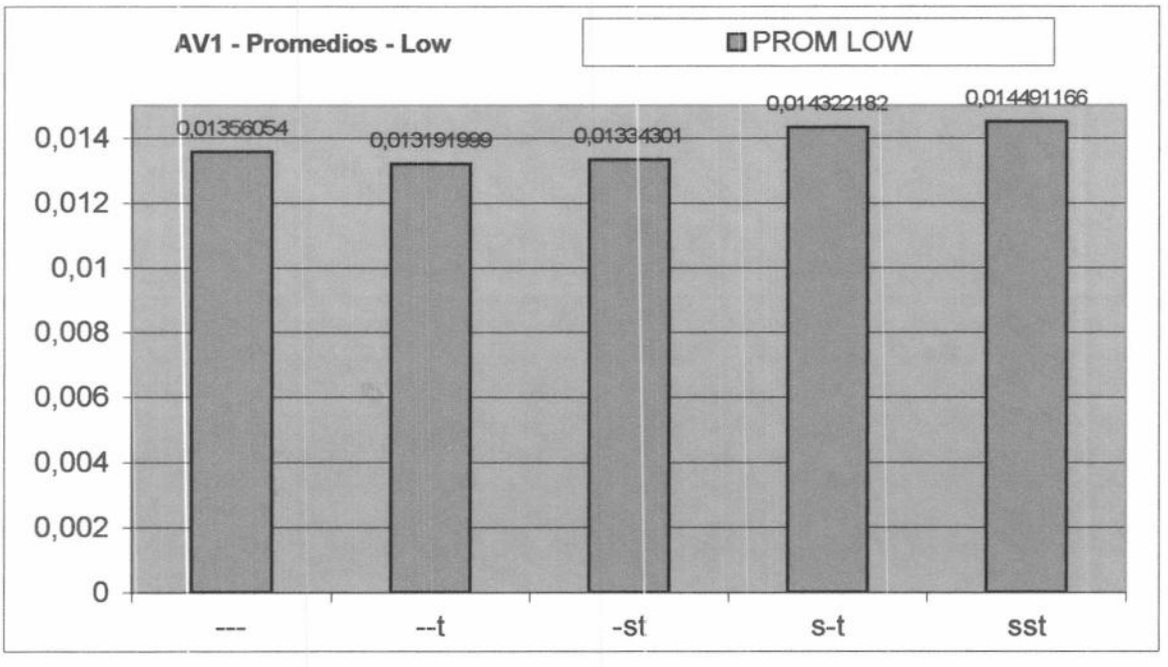
5.2.4.1. AltaVista: variantes de indexación

Comencemos con las evaluaciones de las 30 consultas en AltaVista. Cada valor de la tabla es el promedio de las 256 evaluaciones de los 256 esquemas de relevancia, por cada consulta y variante de indexación. Para confeccionar esta tabla, se realizaron entonces 256 x 5 x 30 = 38.400 evaluaciones.

Consulta	---	--t	-st	s-t	sst

unanticipated	0,028052771	0,032504873	0,034725703	0,032508354	0,033925091
startling	0,032548126	0,035453957	0,025871046	0,036875211	0,028829647
quaint	0,004999746	-0,010821151	-0,015665448	-0,008981143	-0,013089437
imaginable	-0,007447793	-0,028694158	-0,036680545	-0,017003701	-0,020598731
discerning	-0,039021597	-0,036696935	-0,04233816	-0,035347131	-0,042194999
outperform	0,022009648	0,017644726	0,015606001	0,021219931	0,019974227
summertime	0,064780332	0,078894106	0,076461976	0,081147453	0,080030899
hypocrisy	0,020673129	0,028707188	0,037221859	0,03088528	0,039958961
invoicing	-0,034007302	-0,05469482	-0,052945026	-0,048691099	-0,049018325
stroud	0,048357004	0,016116681	0,017780563	0,017561264	0,020084935
buxton	-0,087869238	-0,085258533	-0,084600849	-0,082591074	-0,083812964
mcclain	0,03360703	0,040590172	0,036991304	0,038169819	0,036062563
impasse	-0,016127032	-0,017969826	-0,016385368	-0,017501378	-0,014518462
whomever	0,042066263	0,034252793	0,032541899	0,034679547	0,032243172
optimizer	0,018923946	0,054560485	0,052411133	0,086018876	0,085850096
backdoor	-0,070251781	-0,085495601	-0,086595288	-0,093443032	-0,094782222
battleground	-0,00933584	0,009495962	0,003028404	0,011549708	0,006679894
dutchess	0,007721422	0,002448996	0,003093291	0,003261955	0,001747355
heros	-0,010159046	0,040542706	0,044645719	0,007060921	0,006740425
frustrate	-0,0264029	-0,023937128	-0,020656939	-0,0076225	-0,003640485
coreldraw	-0,004054839	0,005842315	0,004785679	0,011033037	0,010922971
beeper	0,113564487	0,038676614	0,054162354	0,03979429	0,055711287
iplanet	0,032556724	0,039806813	0,04157284	0,040378504	0,042770071
colocation	7,51E-06	-0,00445115	-0,000495406	-0,001508737	0,002199303
sparring	0,043245384	0,042150041	0,049231882	0,041922706	0,04812965
proliferate	0,028974428	0,019129025	0,028003755	0,015484555	0,02470062
relaunch	0,082657698	0,075429882	0,083660014	0,070663311	0,077471638
excavator	-0,018664717	0,045850738	0,030795739	0,04239418	0,024686717
seagram	0,09673357	0,076254629	0,072123533	0,073626106	0,070181564
komputer	0,008679058	0,009426571	0,011938646	0,006120263	0,007489506
PROM LOW	**0,01356054**	**0,013191999**	**0,01334301**	**0,014322182**	**0,014491166**

Gráficamente:



Cada barra representa la evaluación promedio de todas las consultas en cada una de las 5 variantes de indexación. Vemos que los valores de las evaluaciones son menores al 1.5 % del ranking perfecto, es decir el ranking de AV contra el que se compara cada consulta. Lo que nos interesa en este experimento, más que acercarnos absolutamente al ranking de AV, es observar cómo varían las evaluaciones al aplicar diferentes variantes, tanto de indexación como de relevancia.

En el gráfico previo, nos concentramos en las variantes de indexación. Más adelante veremos cómo influyen las variantes de relevancia.

Vemos que el uso de *stemming*, es decir, la aplicación del algoritmo de Porter a los términos representativos de las páginas, redunda en una mejora de las evaluaciones. Para verlo, debemos comparar los esquemas "--t" contra "s-t", y "-st" contra "sst".

Las diferencias son:

```
dif1 = eval(s-t) - eval(--t) = 0,00113018341682771.
dif2 = eval(sst) - eval(-st) = 0,00114815528615883.
```

Para comprender mejor estos valores, recordemos que las evaluaciones se calculan como

```
2 * pares_ok / pares - 1.
```

Donde `pares_ok` es la cantidad de pares de páginas ordenados de manera similar en ambos rankings comparados; mientras que `pares` es la cantidad total de pares de páginas considerados.

La cantidad de pares que mejoran, en promedio, como resultado de la aplicación de *stemming* se sigue de:

```
eval(s-t) = 2 * pares_ok(s-t) / pares - 1
eval(--t) = 2 * pares_ok(--t) / pares - 1
➔ dif1 = 2 * (( pares_ok(s-t) - pares_ok(--t)) / pares)
➔ dif1_pares = pares_ok(s-t) - pares_ok(--t) = dif1 * pares / 2
➔ dif1_pares = 0,00113018341682771 * 19900 / 2 = 11,2453249974357.
```

Como la cantidad de páginas por consulta es de 200, la cantidad total de `pares` es de 19900 = 200 * 199 / 2. Análogamente:

```
dif2_pares = 0,00114815528615883 * 19900 / 2 = 11,4241450972804.
```

Entonces vemos que mejoraron más de 11 pares de páginas al aplicar *stemming*.

Analicemos ahora la influencia de eliminar *stopwords*.

```
dif3 = eval(-st) - eval(--t) = 0,000151011255308545.
dif4 = eval(sst) - eval(s-t) = 0,000168983124639664.
```

La cantidad de pares mejorados es:

```
dif3_pares = 1,50256199032002.
dif4_pares = 1,68138209016466.
```

Vemos que la mejora es casi imperceptible, pues no llega a los dos pares corregidos.

Es claro que la aplicación de *stemming* es más influyente que la eliminación de *stopwords*.

Aquí podemos observar una limitación del experimento por tomar consultas con una sola palabra. Concierne a las pequeñas diferencias observadas al descartar *stopwords*. Si

tuviéramos como consulta a una *stopword* (no ocurre con ninguna de las consultas consideradas), su remoción de la consulta nos dejaría con una consulta vacía o nula. Esto provocaría que todas las páginas tuvieran relevancia nula (en cualquier esquema de relevancia), y las evaluaciones de los *rankings* locales resultantes contra el del MB carecerían de sentido.

En nuestro conjunto de consultas no hay *stopwords*, y por eso su descarte de los documentos afectaría solamente a los esquemas de relevancia que utilicen la normalización de documentos. Ello, sin embargo, no debería afectar casi en nada el *ranking* final de los documentos (nuevamente en cualquier esquema de relevancia), a menos que se suponga que algunos documentos tengan más *stopwords* que otros.

No es sorprendente entonces que la remoción de *stopwords* en nuestro conjunto de consultas analizado sea imperceptible en términos de los pares corregidos.

Veamos el cambio producido por la aplicación sucesiva de descarte de *stopwords* y luego *stemming* a los términos restantes:

```
dif5 = eval(sst) - eval(--t) = 0,00129916654146737.
dif5_pares = 12,9267070876004.
```

La mejora de casi 13 pares es equivalente a sumar las mejoras por separado.

Por último, el uso de texto de atributos de tags elegidos no parece mejorar la evaluación.

Resumamos lo desarrollado hasta aquí en una tabla:

Variante	Id post	Id pre	Diferencia	Pares mejorados
Stemming	s-t	--t	0,00113018341682771	11,2453249974357
	sst	-st	0,00114815528615883	11,4241450972804
Stopwords	-st	--t	0,000151011255308545	1,50256199032002
	sst	s-t	0,000168983124639664	1,68138209016466
Stop-stem	sst	--t	0,00129916654146737	12,9267070876004
Tags	--t	---	-0,000368540505426493	-3,66697802899361

5.2.4.2. AltaVista: variantes de relevancia

Observemos ahora la influencia de las variantes de relevancia.

Recordemos que tenemos 3 componentes principales: tf, df, normalización. Cada componente tiene sus variantes. Son afectados tanto términos en consultas como en páginas.

5.2.4.2.1. Componente tf de consultas

Comencemos con el componente tf de relevancia en consultas. Se mostarán 5 tablas, una para cada variante de indexación, y luego el gráfico. Las 4 columnas de valores se refieren a cada una de las variantes del componente tf : {b, n, a, l}. Cada valor es el promedio de las evaluaciones realizadas con la variante y la consulta específica. Por ejemplo, en la celda (backdoor, qtfb) se encuentra la evaluación promedio de las 64 evaluaciones con esquema de

relevancia b??-??? (donde '?' es cualquier variante); mientras que en (backdoor, qtfn) tenemos la evaluación promedio de los 64 esquemas n??-???.

De esta manera, podremos comparar la influencia de las variantes de cada componente.

—

Consulta	qtfb	qtfn	qtfa	qtfl
startling	0,032548126	0,032548126	0,032548126	0,032548126
proliferate	0,028974428	0,028974428	0,028974428	0,028974428
seagram	0,09673357	0,09673357	0,09673357	0,09673357
frustrate	-0,0264029	-0,0264029	-0,0264029	-0,0264029
coreldraw	-0,004054839	-0,004054839	-0,004054839	-0,004054839
backdoor	-0,070251781	-0,070251781	-0,070251781	-0,070251781
sparring	0,043245384	0,043245384	0,043245384	0,043245384
discerning	-0,039021597	-0,039021597	-0,039021597	-0,039021597
summertime	0,064780332	0,064780332	0,064780332	0,064780332
colocation	7,51E-06	7,51E-06	7,51E-06	7,51E-06
battleground	-0,00933584	-0,00933584	-0,00933584	-0,00933584
excavator	-0,018664717	-0,018664717	-0,018664717	-0,018664717
whomever	0,042066263	0,042066263	0,042066263	0,042066263
relaunch	0,082657698	0,082657698	0,082657698	0,082657698
beeper	0,113564487	0,113564487	0,113564487	0,113564487
invoicing	-0,034007302	-0,034007302	-0,034007302	-0,034007302
komputer	0,008679058	0,008679058	0,008679058	0,008679058
mcclain	0,03360703	0,03360703	0,03360703	0,03360703
imaginable	-0,007447793	-0,007447793	-0,007447793	-0,007447793
unanticipated	0,028052771	0,028052771	0,028052771	0,028052771
hypocrisy	0,020673129	0,020673129	0,020673129	0,020673129
iplanet	0,032556724	0,032556724	0,032556724	0,032556724
buxton	-0,087869238	-0,087869238	-0,087869238	-0,087869238
heros	-0,010159046	-0,010159046	-0,010159046	-0,010159046
stroud	0,048357004	0,048357004	0,048357004	0,048357004
impasse	-0,016127032	-0,016127032	-0,016127032	-0,016127032
quaint	0,004999746	0,004999746	0,004999746	0,004999746
dutchess	0,007721422	0,007721422	0,007721422	0,007721422
optimizer	0,018923946	0,018923946	0,018923946	0,018923946
outperform	0,022009648	0,022009648	0,022009648	0,022009648
PROMEDIO	**0,01356054**	**0,01356054**	**0,01356054**	**0,01356054**

–t

Consulta	qtfb	qtfn	qtfa	qtfl
startling	0,035453957	0,035453957	0,035453957	0,035453957
proliferate	0,019129025	0,019129025	0,019129025	0,019129025
seagram	0,076254629	0,076254629	0,076254629	0,076254629
frustrate	-0,023937128	-0,023937128	-0,023937128	-0,023937128
coreldraw	0,005842315	0,005842315	0,005842315	0,005842315
backdoor	-0,085495601	-0,085495601	-0,085495601	-0,085495601
sparring	0,042150041	0,042150041	0,042150041	0,042150041
discerning	-0,036696935	-0,036696935	-0,036696935	-0,036696935
summertime	0,078894106	0,078894106	0,078894106	0,078894106
colocation	-0,00445115	-0,00445115	-0,00445115	-0,00445115
battleground	0,009495962	0,009495962	0,009495962	0,009495962
excavator	0,045850738	0,045850738	0,045850738	0,045850738
whomever	0,034252793	0,034252793	0,034252793	0,034252793
relaunch	0,075429882	0,075429882	0,075429882	0,075429882

beeper	0,038676614	0,038676614	0,038676614	0,038676614
invoicing	-0,05469482	-0,05469482	-0,05469482	-0,05469482
komputer	0,009426571	0,009426571	0,009426571	0,009426571
mcclain	0,040590172	0,040590172	0,040590172	0,040590172
imaginable	-0,028694158	-0,028694158	-0,028694158	-0,028694158
unanticipated	0,032504873	0,032504873	0,032504873	0,032504873
hypocrisy	0,028707188	0,028707188	0,028707188	0,028707188
iplanet	0,039806813	0,039806813	0,039806813	0,039806813
buxton	-0,085258533	-0,085258533	-0,085258533	-0,085258533
heros	0,040542706	0,040542706	0,040542706	0,040542706
stroud	0,016116681	0,016116681	0,016116681	0,016116681
impasse	-0,017969826	-0,017969826	-0,017969826	-0,017969826
quaint	-0,010821151	-0,010821151	-0,010821151	-0,010821151
dutchess	0,002448996	0,002448996	0,002448996	0,002448996
optimizer	0,054560485	0,054560485	0,054560485	0,054560485
outperform	0,017644726	0,017644726	0,017644726	0,017644726
PROMEDIO	**0,013191999**	**0,013191999**	**0,013191999**	**0,013191999**

-st

Consulta	**qtfb**	**qtfn**	**qtfa**	**qtfl**
startling	0,025871046	0,025871046	0,025871046	0,025871046
proliferate	0,028003755	0,028003755	0,028003755	0,028003755
seagram	0,072123533	0,072123533	0,072123533	0,072123533
frustrate	-0,020656939	-0,020656939	-0,020656939	-0,020656939
coreldraw	0,004785679	0,004785679	0,004785679	0,004785679
backdoor	-0,086595288	-0,086595288	-0,086595288	-0,086595288
sparring	0,049231882	0,049231882	0,049231882	0,049231882
discerning	-0,04233816	-0,04233816	-0,04233816	-0,04233816
summertime	0,076461976	0,076461976	0,076461976	0,076461976
colocation	-0,000495406	-0,000495406	-0,000495406	-0,000495406
battleground	0,003028404	0,003028404	0,003028404	0,003028404
excavator	0,030795739	0,030795739	0,030795739	0,030795739
whomever	0,032541899	0,032541899	0,032541899	0,032541899
relaunch	0,083660014	0,083660014	0,083660014	0,083660014
beeper	0,054162354	0,054162354	0,054162354	0,054162354
invoicing	-0,052945026	-0,052945026	-0,052945026	-0,052945026
komputer	0,011938646	0,011938646	0,011938646	0,011938646
mcclain	0,036991304	0,036991304	0,036991304	0,036991304
imaginable	-0,036680545	-0,036680545	-0,036680545	-0,036680545
unanticipated	0,034725703	0,034725703	0,034725703	0,034725703
hypocrisy	0,037221859	0,037221859	0,037221859	0,037221859
iplanet	0,04157284	0,04157284	0,04157284	0,04157284
buxton	-0,084600849	-0,084600849	-0,084600849	-0,084600849
heros	0,044645719	0,044645719	0,044645719	0,044645719
stroud	0,017780563	0,017780563	0,017780563	0,017780563
impasse	-0,016385368	-0,016385368	-0,016385368	-0,016385368
quaint	-0,015665448	-0,015665448	-0,015665448	-0,015665448
dutchess	0,003093291	0,003093291	0,003093291	0,003093291
optimizer	0,052411133	0,052411133	0,052411133	0,052411133
outperform	0,015606001	0,015606001	0,015606001	0,015606001
PROMEDIO	**0,01334301**	**0,01334301**	**0,01334301**	**0,01334301**

s-t

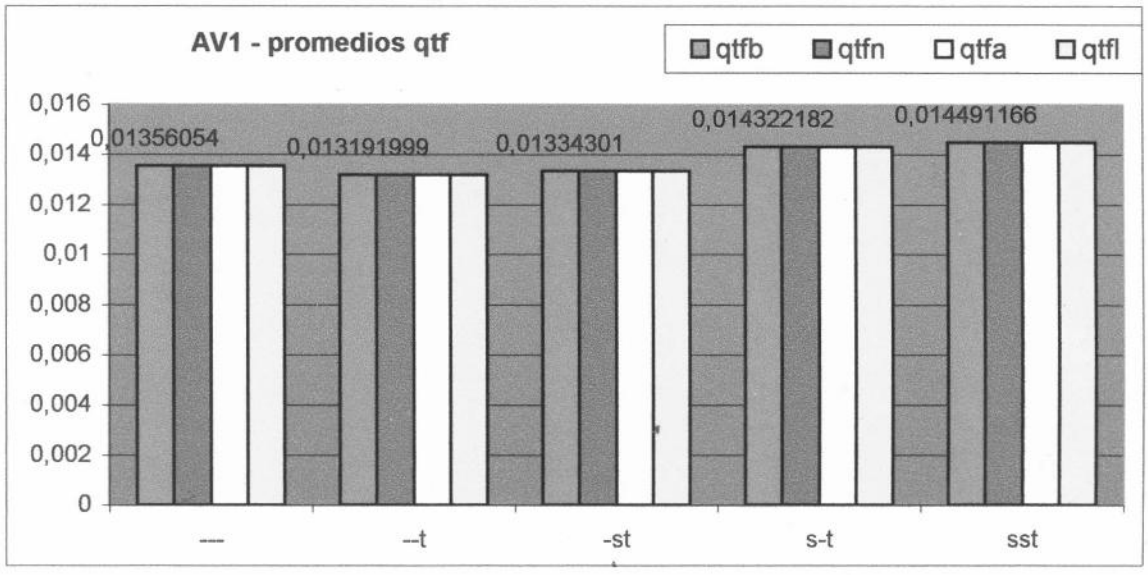
Consulta	qtfb	qtfn	qtfa	qtfl
startling	0,036875211	0,036875211	0,036875211	0,036875211
proliferate	0,015484555	0,015484555	0,015484555	0,015484555
seagram	0,073626106	0,073626106	0,073626106	0,073626106
frustrate	-0,0076225	-0,0076225	-0,0076225	-0,0076225
coreldraw	0,011033037	0,011033037	0,011033037	0,011033037
backdoor	-0,093443032	-0,093443032	-0,093443032	-0,093443032
sparring	0,041922706	0,041922706	0,041922706	0,041922706
discerning	-0,035347131	-0,035347131	-0,035347131	-0,035347131
summertime	0,081147453	0,081147453	0,081147453	0,081147453
colocation	-0,001508737	-0,001508737	-0,001508737	-0,001508737
battleground	0,011549708	0,011549708	0,011549708	0,011549708
excavator	0,04239418	0,04239418	0,04239418	0,04239418
whomever	0,034679547	0,034679547	0,034679547	0,034679547
relaunch	0,070663311	0,070663311	0,070663311	0,070663311
beeper	0,03979429	0,03979429	0,03979429	0,03979429
invoicing	-0,048691099	-0,048691099	-0,048691099	-0,048691099
komputer	0,006120263	0,006120263	0,006120263	0,006120263
mcclain	0,038169819	0,038169819	0,038169819	0,038169819
imaginable	-0,017003701	-0,017003701	-0,017003701	-0,017003701
unanticipated	0,032508354	0,032508354	0,032508354	0,032508354
hypocrisy	0,03088528	0,03088528	0,03088528	0,03088528
iplanet	0,040378504	0,040378504	0,040378504	0,040378504
buxton	-0,082591074	-0,082591074	-0,082591074	-0,082591074
heros	0,007060921	0,007060921	0,007060921	0,007060921
stroud	0,017561264	0,017561264	0,017561264	0,017561264
impasse	-0,017501378	-0,017501378	-0,017501378	-0,017501378
quaint	-0,008981143	-0,008981143	-0,008981143	-0,008981143
dutchess	0,003261955	0,003261955	0,003261955	0,003261955
optimizer	0,086018876	0,086018876	0,086018876	0,086018876
outperform	0,021219931	0,021219931	0,021219931	0,021219931
PROMEDIO	**0,014322182**	**0,014322182**	**0,014322182**	**0,014322182**

sst

Consulta	qtfb	qtfn	qtfa	qtfl
startling	0,028829647	0,028829647	0,028829647	0,028829647
proliferate	0,02470062	0,02470062	0,02470062	0,02470062
seagram	0,070181564	0,070181564	0,070181564	0,070181564
frustrate	-0,003640485	-0,003640485	-0,003640485	-0,003640485
coreldraw	0,010922971	0,010922971	0,010922971	0,010922971
backdoor	-0,094782222	-0,094782222	-0,094782222	-0,094782222
sparring	0,04812965	0,04812965	0,04812965	0,04812965
discerning	-0,042194999	-0,042194999	-0,042194999	-0,042194999
summertime	0,080030899	0,080030899	0,080030899	0,080030899
colocation	0,002199303	0,002199303	0,002199303	0,002199303
battleground	0,006679894	0,006679894	0,006679894	0,006679894
excavator	0,024686717	0,024686717	0,024686717	0,024686717
whomever	0,032243172	0,032243172	0,032243172	0,032243172
relaunch	0,077471638	0,077471638	0,077471638	0,077471638
beeper	0,055711287	0,055711287	0,055711287	0,055711287
invoicing	-0,049018325	-0,049018325	-0,049018325	-0,049018325
komputer	0,007489506	0,007489506	0,007489506	0,007489506
mcclain	0,036062563	0,036062563	0,036062563	0,036062563
imaginable	-0,020598731	-0,020598731	-0,020598731	-0,020598731

unanticipated	0,033925091	0,033925091	0,033925091	0,033925091
hypocrisy	0,039958961	0,039958961	0,039958961	0,039958961
iplanet	0,042770071	0,042770071	0,042770071	0,042770071
buxton	-0,083812964	-0,083812964	-0,083812964	-0,083812964
heros	0,006740425	0,006740425	0,006740425	0,006740425
stroud	0,020084935	0,020084935	0,020084935	0,020084935
impasse	-0,014518462	-0,014518462	-0,014518462	-0,014518462
quaint	-0,013089437	-0,013089437	-0,013089437	-0,013089437
dutchess	0,001747355	0,001747355	0,001747355	0,001747355
optimizer	0,085850096	0,085850096	0,085850096	0,085850096
outperform	0,019974227	0,019974227	0,019974227	0,019974227
PROMEDIO	**0,014491166**	**0,014491166**	**0,014491166**	**0,014491166**

Gráficamente:



Algunas cosas obvias que se ven son que los promedios en cada variante de indexación son los mismos que vimos al promediar en general.

También vemos claramente que si nos concentramos en una variante de indexación dada, las 4 evaluaciones promedio son iguales aunque cambiemos las variantes del componente tf de la consulta. Esto se explica porque cualquiera sea la variante en {b, n, a, l}, las páginas de cada *ranking* local se ordenarán de la misma manera, y por lo tanto la evaluación será la misma.

Pongamos estas palabras en fórmulas. La idea es ver que el peso asignado a un término de una consulta, aunque varíe si cambiamos de variante tf, no provocará que los puntajes asignados a las páginas cambien el orden de las mismas en el *ranking* resultante.

El puntaje de una página d se calcula haciendo el producto interno entre los vectores de pesos de dicha página y de la consulta q. Cuando la consulta consta de un solo término t, la similitud se calcula:

$$sim(d, q) = w_{td} \cdot w_{tq}$$

El peso w_{tq} del término t en la consulta q, consta de tres componentes: tf, df y normalización.

$$w_{tq} = qtf.\ qdf.\ qnorm$$

La relación entre los puntajes de 2 páginas será:

$$sim(d1, q) / sim(d2, q) = w_{td1} / w_{td2}$$

Por lo que esta relación no cambia si se varía qtf, y el orden relativo de d1 y d2 en un *ranking* tampoco cambia. En realidad, se puede adelantar que no importa qué valor tenga w_{tq}, el orden de las páginas en un *ranking* no cambiará, y por lo tanto tampoco la evaluación del *ranking* respecto al del sistema evaluado.

5.2.4.2.2. *Componentes df y normalización en consultas*

Debemos esperar entonces que las evaluaciones no se modifiquen tampoco al variar los componentes df o normalización de consultas. Es lo que mostramos a continuación.

—

Consulta	qdfn	qdft	qnn	qnc
startling	0,03254813	0,03254813	0,03254813	0,03254813
proliferate	0,02897443	0,02897443	0,02897443	0,02897443
seagram	0,09673357	0,09673357	0,09673357	0,09673357
frustrate	-0,0264029	-0,0264029	-0,0264029	-0,0264029
coreldraw	-0,00405484	-0,00405484	-0,00405484	-0,00405484
backdoor	-0,07025178	-0,07025178	-0,07025178	-0,07025178
sparring	0,04324538	0,04324538	0,04324538	0,04324538
discerning	-0,0390216	-0,0390216	-0,0390216	-0,0390216
summertime	0,06478033	0,06478033	0,06478033	0,06478033
colocation	7,51E-06	7,51E-06	7,51E-06	7,51E-06
battleground	-0,00933584	-0,00933584	-0,00933584	-0,00933584
excavator	-0,01866472	-0,01866472	-0,01866472	-0,01866472
whomever	0,04206626	0,04206626	0,04206626	0,04206626
relaunch	0,0826577	0,0826577	0,0826577	0,0826577
beeper	0,11356449	0,11356449	0,11356449	0,11356449
invoicing	-0,0340073	-0,0340073	-0,0340073	-0,0340073
komputer	0,00867906	0,00867906	0,00867906	0,00867906
mcclain	0,03360703	0,03360703	0,03360703	0,03360703
imaginable	-0,00744779	-0,00744779	-0,00744779	-0,00744779
unanticipated	0,02805277	0,02805277	0,02805277	0,02805277
hypocrisy	0,02067313	0,02067313	0,02067313	0,02067313
iplanet	0,03255672	0,03255672	0,03255672	0,03255672
buxton	-0,08786924	-0,08786924	-0,08786924	-0,08786924
heros	-0,01015905	-0,01015905	-0,01015905	-0,01015905
stroud	0,048357	0,048357	0,048357	0,048357
impasse	-0,01612703	-0,01612703	-0,01612703	-0,01612703
quaint	0,00499975	0,00499975	0,00499975	0,00499975
dutchess	0,00772142	0,00772142	0,00772142	0,00772142
optimizer	0,01892395	0,01892395	0,01892395	0,01892395
outperform	0,02200965	0,02200965	0,02200965	0,02200965
PROMEDIO	**0,01356054**	**0,01356054**	**0,01356054**	**0,01356054**

–t

Consulta	qdfn	qdft	qnn	qnc
startling	0,03545396	0,03545396	0,03545396	0,03545396
proliferate	0,01912902	0,01912902	0,01912902	0,01912902
seagram	0,07625463	0,07625463	0,07625463	0,07625463

frustrate	-0,02393713	-0,02393713	-0,02393713	-0,02393713
coreldraw	0,00584231	0,00584231	0,00584231	0,00584231
backdoor	-0,0854956	-0,0854956	-0,0854956	-0,0854956
sparring	0,04215004	0,04215004	0,04215004	0,04215004
discerning	-0,03669693	-0,03669693	-0,03669693	-0,03669693
summertime	0,07889411	0,07889411	0,07889411	0,07889411
colocation	-0,00445115	-0,00445115	-0,00445115	-0,00445115
battleground	0,00949596	0,00949596	0,00949596	0,00949596
excavator	0,04585074	0,04585074	0,04585074	0,04585074
whomever	0,03425279	0,03425279	0,03425279	0,03425279
relaunch	0,07542988	0,07542988	0,07542988	0,07542988
beeper	0,03867661	0,03867661	0,03867661	0,03867661
invoicing	-0,05469482	-0,05469482	-0,05469482	-0,05469482
komputer	0,00942657	0,00942657	0,00942657	0,00942657
mcclain	0,04059017	0,04059017	0,04059017	0,04059017
imaginable	-0,02869416	-0,02869416	-0,02869416	-0,02869416
unanticipated	0,03250487	0,03250487	0,03250487	0,03250487
hypocrisy	0,02870719	0,02870719	0,02870719	0,02870719
iplanet	0,03980681	0,03980681	0,03980681	0,03980681
buxton	-0,08525853	-0,08525853	-0,08525853	-0,08525853
heros	0,04054271	0,04054271	0,04054271	0,04054271
stroud	0,01611668	0,01611668	0,01611668	0,01611668
impasse	-0,01796983	-0,01796983	-0,01796983	-0,01796983
quaint	-0,01082115	-0,01082115	-0,01082115	-0,01082115
dutchess	0,002449	0,002449	0,002449	0,002449
optimizer	0,05456048	0,05456048	0,05456048	0,05456048
outperform	0,01764473	0,01764473	0,01764473	0,01764473
PROMEDIO	**0,013192**	**0,013192**	**0,013192**	**0,013192**

-st

Consulta	**qdfn**	**qdft**	**qnn**	**qnc**
startling	0,02587105	0,02587105	0,02587105	0,02587105
proliferate	0,02800375	0,02800375	0,02800375	0,02800375
seagram	0,07212353	0,07212353	0,07212353	0,07212353
frustrate	-0,02065694	-0,02065694	-0,02065694	-0,02065694
coreldraw	0,00478568	0,00478568	0,00478568	0,00478568
backdoor	-0,08659529	-0,08659529	-0,08659529	-0,08659529
sparring	0,04923188	0,04923188	0,04923188	0,04923188
discerning	-0,04233816	-0,04233816	-0,04233816	-0,04233816
summertime	0,07646198	0,07646198	0,07646198	0,07646198
colocation	-0,00049541	-0,00049541	-0,00049541	-0,00049541
battleground	0,0030284	0,0030284	0,0030284	0,0030284
excavator	0,03079574	0,03079574	0,03079574	0,03079574
whomever	0,0325419	0,0325419	0,0325419	0,0325419
relaunch	0,08366001	0,08366001	0,08366001	0,08366001
beeper	0,05416235	0,05416235	0,05416235	0,05416235
invoicing	-0,05294503	-0,05294503	-0,05294503	-0,05294503
komputer	0,01193865	0,01193865	0,01193865	0,01193865
mcclain	0,0369913	0,0369913	0,0369913	0,0369913
imaginable	-0,03668054	-0,03668054	-0,03668054	-0,03668054
unanticipated	0,0347257	0,0347257	0,0347257	0,0347257
hypocrisy	0,03722186	0,03722186	0,03722186	0,03722186
iplanet	0,04157284	0,04157284	0,04157284	0,04157284
buxton	-0,08460085	-0,08460085	-0,08460085	-0,08460085

heros	0,04464572	0,04464572	0,04464572	0,04464572
stroud	0,01778056	0,01778056	0,01778056	0,01778056
impasse	-0,01638537	-0,01638537	-0,01638537	-0,01638537
quaint	-0,01566545	-0,01566545	-0,01566545	-0,01566545
dutchess	0,00309329	0,00309329	0,00309329	0,00309329
optimizer	0,05241113	0,05241113	0,05241113	0,05241113
outperform	0,015606	0,015606	0,015606	0,015606
PROMEDIO	**0,01334301**	**0,01334301**	**0,01334301**	**0,01334301**

s-t

Consulta	qdfn	qdft	qnn	qnc
startling	0,03687521	0,03687521	0,03687521	0,03687521
proliferate	0,01548455	0,01548455	0,01548455	0,01548455
seagram	0,07362611	0,07362611	0,07362611	0,07362611
frustrate	-0,0076225	-0,0076225	-0,0076225	-0,0076225
coreldraw	0,01103304	0,01103304	0,01103304	0,01103304
backdoor	-0,09344303	-0,09344303	-0,09344303	-0,09344303
sparring	0,04192271	0,04192271	0,04192271	0,04192271
discerning	-0,03534713	-0,03534713	-0,03534713	-0,03534713
summertime	0,08114745	0,08114745	0,08114745	0,08114745
colocation	-0,00150874	-0,00150874	-0,00150874	-0,00150874
battleground	0,01154971	0,01154971	0,01154971	0,01154971
excavator	0,04239418	0,04239418	0,04239418	0,04239418
whomever	0,03467955	0,03467955	0,03467955	0,03467955
relaunch	0,07066331	0,07066331	0,07066331	0,07066331
beeper	0,03979429	0,03979429	0,03979429	0,03979429
invoicing	-0,0486911	-0,0486911	-0,0486911	-0,0486911
komputer	0,00612026	0,00612026	0,00612026	0,00612026
mcclain	0,03816982	0,03816982	0,03816982	0,03816982
imaginable	-0,0170037	-0,0170037	-0,0170037	-0,0170037
unanticipated	0,03250835	0,03250835	0,03250835	0,03250835
hypocrisy	0,03088528	0,03088528	0,03088528	0,03088528
iplanet	0,0403785	0,0403785	0,0403785	0,0403785
buxton	-0,08259107	-0,08259107	-0,08259107	-0,08259107
heros	0,00706092	0,00706092	0,00706092	0,00706092
stroud	0,01756126	0,01756126	0,01756126	0,01756126
impasse	-0,01750138	-0,01750138	-0,01750138	-0,01750138
quaint	-0,00898114	-0,00898114	-0,00898114	-0,00898114
dutchess	0,00326195	0,00326195	0,00326195	0,00326195
optimizer	0,08601888	0,08601888	0,08601888	0,08601888
outperform	0,02121993	0,02121993	0,02121993	0,02121993
PROMEDIO	**0,01432218**	**0,01432218**	**0,01432218**	**0,01432218**

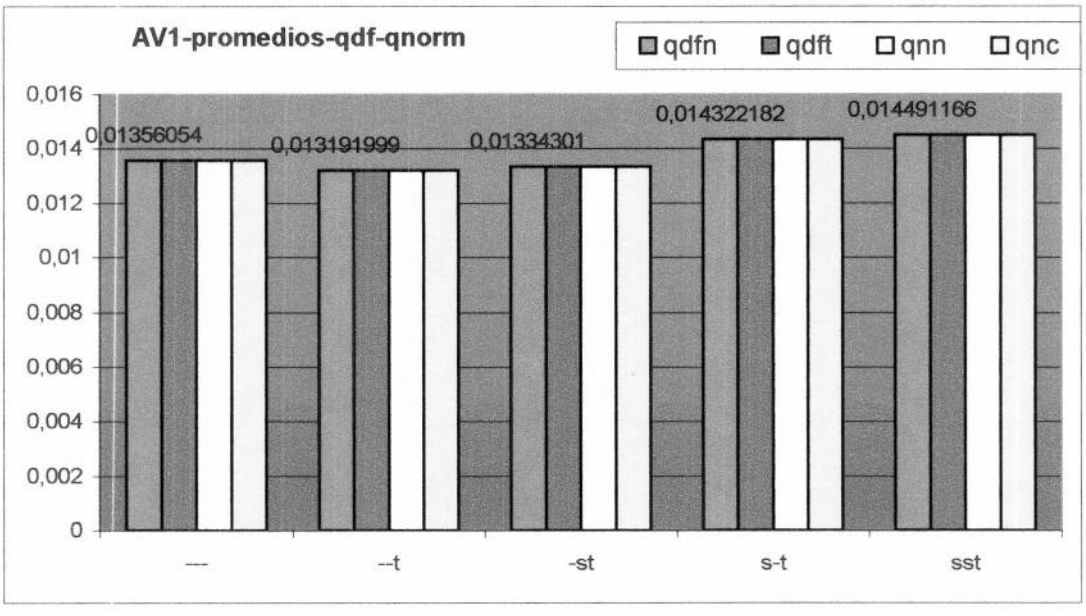
sst

Consulta	qdfn	qdft	qnn	qnc
startling	0,02882965	0,02882965	0,02882965	0,02882965
proliferate	0,02470062	0,02470062	0,02470062	0,02470062
seagram	0,07018156	0,07018156	0,07018156	0,07018156
frustrate	-0,00364049	-0,00364049	-0,00364049	-0,00364049
coreldraw	0,01092297	0,01092297	0,01092297	0,01092297
backdoor	-0,09478222	-0,09478222	-0,09478222	-0,09478222
sparring	0,04812965	0,04812965	0,04812965	0,04812965
discerning	-0,042195	-0,042195	-0,042195	-0,042195

summertime	0,0800309	0,0800309	0,0800309	0,0800309
colocation	0,0021993	0,0021993	0,0021993	0,0021993
battleground	0,00667989	0,00667989	0,00667989	0,00667989
excavator	0,02468672	0,02468672	0,02468672	0,02468672
whomever	0,03224317	0,03224317	0,03224317	0,03224317
relaunch	0,07747164	0,07747164	0,07747164	0,07747164
beeper	0,05571129	0,05571129	0,05571129	0,05571129
invoicing	-0,04901832	-0,04901832	-0,04901832	-0,04901832
komputer	0,00748951	0,00748951	0,00748951	0,00748951
mcclain	0,03606256	0,03606256	0,03606256	0,03606256
imaginable	-0,02059873	-0,02059873	-0,02059873	-0,02059873
unanticipated	0,03392509	0,03392509	0,03392509	0,03392509
hypocrisy	0,03995896	0,03995896	0,03995896	0,03995896
iplanet	0,04277007	0,04277007	0,04277007	0,04277007
buxton	-0,08381296	-0,08381296	-0,08381296	-0,08381296
heros	0,00674043	0,00674043	0,00674043	0,00674043
stroud	0,02008493	0,02008493	0,02008493	0,02008493
impasse	-0,01451846	-0,01451846	-0,01451846	-0,01451846
quaint	-0,01308944	-0,01308944	-0,01308944	-0,01308944
dutchess	0,00174736	0,00174736	0,00174736	0,00174736
optimizer	0,0858501	0,0858501	0,0858501	0,0858501
outperform	0,01997423	0,01997423	0,01997423	0,01997423
PROMEDIO	**0,01449117**	**0,01449117**	**0,01449117**	**0,01449117**

En estas 5 tablas (una por cada variante de indexación) tenemos las evaluaciones promedio de las 30 consultas en las variantes {n, t} del componente df de consultas, y las variantes {n, c} del componente de normalización de consultas. Cada valor es la evaluación promedio de las 128 evaluaciones realizadas con cada variante. Por ejemplo, en (outperform, qdfn) tenemos el promedio de las 128 evaluaciones de esquemas ?n?-??? (? es cualquier variante); y en (outperform, qnc) se promedian 128 evaluaciones con ??c-??? como esquema.

Gráficamente:



Nuevamente los promedios en cada variante de indexación son coincidentes con los vistos hasta ahora. Las variantes de los componentes df y normalización de consultas, tal como se mostró en el análisis de fórmulas, no cambian las evaluaciones.

5.2.4.2.3. Componente tf de documentos

Pasemos ahora a analizar el efecto de cambiar las variantes de relevancia, pero ahora en la terna correspondiente a los documentos (o páginas).

Comencemos con las variantes del componente tf.

Consulta	dtfb	dtfn	dtfa	dtfl
startling	0,02465383	0,03385681	0,04497354	0,02670832
proliferate	0,02424906	0,03473091	0,01672545	0,04019229
seagram	0,04740757	0,1229678	0,10052727	0,11603164
frustrate	-0,01328589	-0,03206629	-0,02822314	-0,03203627
coreldraw	-0,00751972	0,00091575	-0,0041561	-0,00545928
backdoor	-0,0388736	-0,07782815	-0,08725982	-0,07704555
sparring	0,03324607	0,04767153	0,04213282	0,04993111
discerning	-0,03419503	-0,0390216	-0,03933519	-0,04353458
summertime	0,06931401	0,04691548	0,09370952	0,04918232
colocation	-0,01188975	0,01675374	-0,01747433	0,01264037
battleground	-0,00348092	-0,01517683	-0,0005291	-0,0181565
excavator	-0,04805068	-0,01262879	0,0047619	-0,0187413
whomever	0,05439168	0,02554314	0,06604593	0,0222843
relaunch	0,04496317	0,11154669	0,05452602	0,11959492
beeper	0,09231787	0,13026853	0,10331493	0,12835662
invoicing	-0,02550289	-0,04216037	-0,02606779	-0,04229815
komputer	-0,03439998	0,04102697	-0,00971767	0,03780691
mcclain	0,03396938	0,01926433	0,06658786	0,01460655
imaginable	-0,01272799	-0,0041898	-0,00889506	-0,00397832
unanticipated	0,02238931	0,0269702	0,03692565	0,02592593
hypocrisy	0,00426756	0,04592059	-0,01788831	0,05039267

iplanet	0,04837847	0,01006771	0,04974459	0,02203611
buxton	-0,03673682	-0,1143769	-0,09648256	-0,10388067
heros	-0,01932322	0,00090807	-0,02537258	0,00315154
stroud	0,04023949	0,05207463	0,05233918	0,04877471
impasse	0,00250758	-0,02989804	-0,01025076	-0,02686691
quaint	0,02904675	-0,00699203	0,00411147	-0,0061672
dutchess	-0,01372247	0,01910622	0,01002537	0,01547658
optimizer	-0,01634059	0,03904657	0,01694682	0,03604299
outperform	0,02890563	0,01160455	0,03383558	0,01369284
PROMEDIO	**0,00947326**	**0,01542738**	**0,01418605**	**0,01515547**

--t

Consulta	**dtfb**	**dtfn**	**dtfa**	**dtfl**
startling	0,02209276	0,04100529	0,04258134	0,03613644
proliferate	0,01718057	0,02386506	0,00870406	0,02676641
seagram	0,03344109	0,09974264	0,08025234	0,09158245
frustrate	-0,01537261	-0,02555095	-0,02840329	-0,02642167
coreldraw	-0,00125035	0,01024937	0,00151451	0,01285573
backdoor	-0,04678055	-0,09655656	-0,10109024	-0,09755505
sparring	0,02562689	0,05250758	0,03769634	0,05276936
discerning	-0,01164376	-0,05112893	-0,02936846	-0,0546466
summertime	0,07592563	0,06658841	0,10338407	0,06967832
colocation	-0,04410617	0,02735243	-0,01441182	0,01336096
battleground	0,02291841	-0,00292398	0,02066277	-0,00267335
excavator	0,02354497	0,04493177	0,0673907	0,04753551
whomever	0,04743948	0,01843575	0,05574178	0,01539417
relaunch	0,02882217	0,11009147	0,04766572	0,11514018
beeper	-0,01824219	0,07620321	0,02025473	0,07649071
invoicing	-0,03269496	-0,07610912	-0,02893359	-0,08104161
komputer	-0,02205164	0,03445748	-0,00635386	0,0316543
mcclain	0,03574243	0,02839694	0,0730609	0,02516042
imaginable	-0,03505155	-0,02083003	-0,04199048	-0,01690457
unanticipated	0,02428293	0,03233083	0,04227235	0,03113339
hypocrisy	0,0151887	0,05006545	-0,00586278	0,05543739
iplanet	0,05757009	0,01107745	0,06402946	0,02655025
buxton	-0,02656108	-0,11656696	-0,09283692	-0,10506917
heros	0,02581326	0,05190695	0,0264409	0,05800972
stroud	0,01399332	0,01385408	0,02949039	0,00712893
impasse	0,00502893	-0,03320474	-0,01077432	-0,03292918
quaint	0,00241105	-0,01399675	-0,02273996	-0,00895894
dutchess	-0,00416937	0,00670607	0,00277958	0,00447971
optimizer	0,02488289	0,06751171	0,06332323	0,06252411
outperform	0,02685699	0,00578906	0,02923606	0,0086968
PROMEDIO	**0,00902794**	**0,01454003**	**0,01445718**	**0,01474284**

-st

Consulta	**dtfb**	**dtfn**	**dtfa**	**dtfl**
startling	0,01865924	0,0290161	0,02544186	0,03036699
proliferate	0,01870236	0,03588292	0,02884287	0,02858687
seagram	0,03151089	0,09415605	0,07058565	0,09224154
frustrate	-0,01472708	-0,02205308	-0,02059689	-0,02525071
coreldraw	-0,00181389	0,01056636	0,0004931	0,00989715
backdoor	-0,04796794	-0,0964756	-0,10343804	-0,09849957

sparring	0,02834114	0,05833563	0,0553596	0,05489115
discerning	-0,01404341	-0,0599231	-0,03749455	-0,05789158
summertime	0,07738288	0,06282383	0,09733916	0,06830203
colocation	-0,04385096	0,03148081	-0,00216177	0,01255029
battleground	0,02074631	-0,01077694	0,00690615	-0,0047619
excavator	0,01981342	0,03327764	0,02834865	0,04174325
whomever	0,04605835	0,01834264	0,0483861	0,01738051
relaunch	0,0291637	0,1181694	0,07087491	0,11643205
beeper	-0,01382899	0,09261975	0,05836352	0,07949514
invoicing	-0,02779002	-0,07688068	-0,02857537	-0,07853403
komputer	-0,02057098	0,03813754	-0,00212754	0,03231557
mcclain	0,03602387	0,0233733	0,06499775	0,0235703
imaginable	-0,03337298	-0,03172086	-0,0637457	-0,01788263
unanticipated	0,02453356	0,03558897	0,04675578	0,03202451
hypocrisy	0,01821553	0,0598822	0,01360711	0,05718259
iplanet	0,05826278	0,01297064	0,06639945	0,0286585
buxton	-0,02979275	-0,11436355	-0,08865712	-0,10558998
heros	0,02769617	0,05516532	0,03638962	0,05933177
stroud	0,01329713	0,01265664	0,03767753	0,00749095
impasse	0,00549738	-0,02987049	-0,00980986	-0,0313585
quaint	-0,00312167	-0,01969443	-0,02228313	-0,01756256
dutchess	-0,00578854	0,00960708	0,00914832	-0,0005937
optimizer	0,02499311	0,06398457	0,06095343	0,05971342
outperform	0,02784827	0,00219403	0,02275971	0,00962199
PROMEDIO	**0,00900256**	**0,01454909**	**0,01569134**	**0,01412905**

s-t

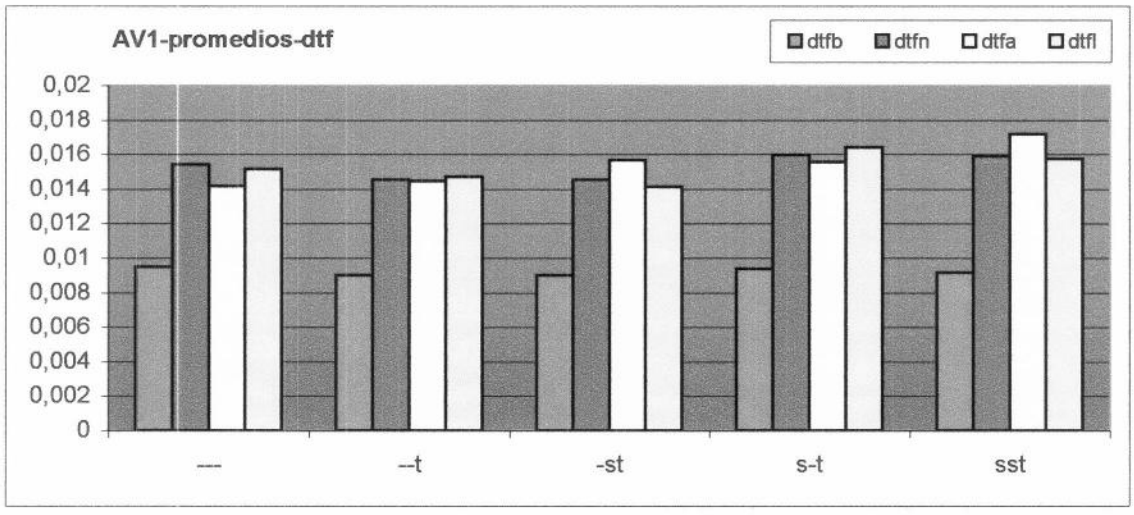
Consulta	**dtfb**	**dtfn**	**dtfa**	**dtfl**
startling	0,02254306	0,04269391	0,04345379	0,03881009
proliferate	0,015289	0,01934236	0,00497781	0,02232905
seagram	0,03151089	0,09514469	0,07996987	0,08787898
frustrate	-0,01367621	9,01E-05	-0,01771453	0,00081066
coreldraw	0,00154973	0,01454635	0,01035503	0,01768104
backdoor	-0,04753616	-0,10805268	-0,11061636	-0,10756693
sparring	0,02799669	0,04917333	0,03992835	0,05059245
discerning	-0,01014398	-0,0495337	-0,02793685	-0,053774
summertime	0,07434693	0,07121654	0,10454447	0,07448187
colocation	-0,03985768	0,02805801	-0,00945776	0,01522248
battleground	0,02199944	-0,00050125	0,02684489	-0,00214425
excavator	0,02628794	0,03357004	0,06512114	0,04459761
whomever	0,04725326	0,01905649	0,05738672	0,01502173
relaunch	0,0291043	0,10208779	0,04561654	0,10584462
beeper	-0,01750906	0,07766948	0,02315853	0,0758582
invoicing	-0,03736567	-0,06525214	-0,02488289	-0,06726371
komputer	-0,02001035	0,02885113	-0,00911391	0,02475418
mcclain	0,03180232	0,02763706	0,06914894	0,02409096
imaginable	-0,03170764	-0,00260375	-0,03991541	0,006212
unanticipated	0,02449179	0,03191312	0,04250905	0,03111947
hypocrisy	0,01582952	0,05352858	-0,00447208	0,0586551
iplanet	0,05779282	0,01251782	0,06426705	0,02693633
buxton	-0,01902943	-0,11601944	-0,09059345	-0,10472197
heros	0,00512793	0,01179157	-0,00408632	0,0154105
stroud	0,01578947	0,01549708	0,02921192	0,00974659
impasse	0,00524938	-0,0316065	-0,0111601	-0,03248829

quaint	0,00633217	-0,01390792	-0,02073499	-0,00761383
dutchess	-0,00080959	0,00711086	0,00315738	0,00358916
optimizer	0,03164784	0,112345	0,09840176	0,1016809
outperform	0,02661909	0,01258261	0,02928892	0,01638911
PROMEDIO	**0,00936393**	**0,01596488**	**0,01555525**	**0,01640467**

sst

Consulta	**dtfb**	**dtfn**	**dtfa**	**dtfl**
startling	0,01912361	0,03267477	0,0294664	0,03405381
proliferate	0,01757879	0,03144556	0,02412106	0,02565707
seagram	0,02890591	0,09075074	0,07501098	0,08605863
frustrate	-0,01243019	0,00507416	-0,00993815	0,00273224
coreldraw	-0,00047549	0,01394759	0,01532122	0,01489856
backdoor	-0,04802191	-0,10874082	-0,1119117	-0,11045445
sparring	0,03119317	0,05275558	0,05745384	0,05111601
discerning	-0,01292539	-0,05913231	-0,03970332	-0,05701898
summertime	0,07592563	0,0689497	0,10211572	0,07313256
colocation	-0,0397526	0,03095538	0,0015913	0,01600312
battleground	0,02025898	-0,0053467	0,0160401	-0,0042328
excavator	0,02071846	0,01882484	0,02340574	0,03579783
whomever	0,04455307	0,01952204	0,04931719	0,01558038
relaunch	0,02837669	0,11012117	0,06502435	0,10636434
beeper	-0,0088264	0,09240412	0,05890978	0,08035766
invoicing	-0,03408653	-0,06620281	-0,03138606	-0,06439791
komputer	-0,02042723	0,03066241	-0,00531884	0,02504169
mcclain	0,03170382	0,02293707	0,06755882	0,02205055
imaginable	-0,02936823	-0,01025641	-0,04789849	0,00512821
unanticipated	0,02328042	0,03461431	0,04515455	0,03265107
hypocrisy	0,01806555	0,06326353	0,01769743	0,06080934
iplanet	0,05994415	0,014637	0,06672972	0,02976941
buxton	-0,02395705	-0,11389616	-0,09100742	-0,10639122
heros	0,00619625	0,00957481	-0,00471396	0,0159046
stroud	0,01714007	0,01555277	0,03737121	0,01027569
impasse	0,00544227	-0,02752824	-0,00531827	-0,03066961
quaint	-0,00048221	-0,01801939	-0,01804477	-0,01581138
dutchess	-0,00447971	0,0091753	0,00304944	-0,00075561
optimizer	0,0325434	0,11005787	0,100248	0,10055112
outperform	0,02876024	0,00818134	0,02474227	0,01821306
PROMEDIO	**0,00914925**	**0,01589864**	**0,0171696**	**0,01574717**

Gráficamente:



Promediemos a su vez los promedios de las 5 tablas anteriores:

Index id	dtfb	dtfn	dtfa	dtfl	Prom
---	0,009473262	0,01542738	0,01418605	0,01515547	**0,01356054**
--t	0,009027944	0,01454003	0,01445718	0,01474284	**0,013192**
-st	0,009002562	0,01454909	0,01569134	0,01412905	**0,01334301**
s-t	0,009363927	0,01596488	0,01555525	0,01640467	**0,01432218**
sst	0,009149252	0,01589864	0,0171696	0,01574717	**0,01449117**
Promedio	**0,009203389**	**0,015276**	**0,01541189**	**0,01523584**	

Es evidente que en cualquier variante de indexación (*index id*), la peor performance es de la variante b del componente tf. Esta variante, recordemos, implica la ausencia del componente tf, pues se multiplica por el valor 1 al calcular el peso de un término t en un documento d, i.e. w_{td}.

Luego tenemos las otras tres variantes {n, a, l} que van turnando su supremacía; pero la mejor evaluación, promediando para todos los index ids, es para la variante a. Luego la sigue la variante n; más abajo la l.

Es importante recalcar que la variante b (factor 1) queda muy rezagada frente a las otras 3, algo bastante razonable pues la frecuencia de un término en un documento tiene que ser tenida en cuenta de alguna manera por el MB, en este caso AltaVista.

5.2.4.2.4. Componentes df y normalización en documentos

Analicemos ahora los componentes df y normalización de términos en documentos.

Consulta	ddfn	ddft	dnn	dnc
startling	0,03284363	0,03225262	0,02617359	0,03892266
proliferate	0,02860109	0,02934776	0,03645181	0,02149704
seagram	0,09666688	0,09680026	0,10233193	0,09113521
frustrate	-0,02648922	-0,02631658	-0,02410977	-0,02869603
coreldraw	-0,00416491	-0,00394477	0,00567061	-0,01378029
backdoor	-0,07019781	-0,07030575	-0,05244765	-0,08805592
sparring	0,04294572	0,04354505	0,04162304	0,04486773
discerning	-0,03870801	-0,03933519	-0,0325998	-0,04544339
summertime	0,06490852	0,06465215	0,03239691	0,09716375
colocation	0,00013511	-0,0001201	0,0178046	-0,01778959

battleground	-0,00873712	-0,00993456	-0,01393762	-0,00473406
excavator	-0,01818435	-0,01914508	-0,03079922	-0,00653021
whomever	0,0426676	0,04146493	0,02212911	0,06200341
relaunch	0,08258345	0,08273194	0,10262236	0,06269304
beeper	0,11337042	0,11375855	0,12237652	0,10475246
invoicing	-0,03430697	-0,03370763	-0,03969413	-0,02832047
komputer	0,0090636	0,00829452	0,03138117	-0,01402306
mcclain	0,03392716	0,0332869	0,00688112	0,06033294
imaginable	-0,00755353	-0,00734206	0,00117632	-0,0160719
unanticipated	0,02809106	0,02801448	0,01946533	0,03664021
hypocrisy	0,02046521	0,02088105	0,05423757	-0,01289131
iplanet	0,032579	0,03253445	0,00605845	0,059055
buxton	-0,08749533	-0,08824315	-0,08426366	-0,09147481
heros	-0,01040943	-0,00990866	-0,00058758	-0,01973052
stroud	0,04825258	0,04846143	0,04793929	0,04877471
impasse	-0,01567925	-0,01657481	-0,02240287	-0,0098512
quaint	0,00482844	0,00517106	-0,0022207	0,01222019
dutchess	0,0082038	0,00723904	0,01067304	0,00476981
optimizer	0,01941995	0,01842794	0,02960871	0,00823918
outperform	0,02205921	0,02196008	0,00728258	0,03673672
PROMEDIO	**0,01365622**	**0,01346486**	**0,0140407**	**0,01308038**

–t

Consulta	**ddfn**	**ddft**	**dnn**	**dnc**
startling	0,03539767	0,03551024	0,03190082	0,03900709
proliferate	0,01868102	0,01957703	0,02696553	0,01129252
seagram	0,07658025	0,07592901	0,08168037	0,07082889
frustrate	-0,02394463	-0,02392962	-0,01909566	-0,0287786
coreldraw	0,00599641	0,00568822	0,01310228	-0,00141765
backdoor	-0,08548548	-0,08550572	-0,07054188	-0,10044932
sparring	0,04186415	0,04243593	0,04618352	0,03811656
discerning	-0,03641743	-0,03697644	-0,03619928	-0,03719459
summertime	0,07906952	0,0787187	0,04722582	0,11056239
colocation	-0,00452621	-0,00437609	0,02178286	-0,03068516
battleground	0,00996241	0,00902952	0,00087719	0,01811473
excavator	0,04635895	0,04534252	0,02580061	0,06590086
whomever	0,03463687	0,03386872	0,01575109	0,0527545
relaunch	0,07509949	0,07576028	0,09449988	0,05635988
beeper	0,03857599	0,03877724	0,05400782	0,02334541
invoicing	-0,05456048	-0,05482915	-0,06796638	-0,04142326
komputer	0,00994767	0,00890547	0,02915301	-0,01029987
mcclain	0,04077311	0,04040724	0,01450805	0,0666723
imaginable	-0,02866772	-0,02872059	-0,01369284	-0,04369548
unanticipated	0,03258145	0,03242829	0,02325258	0,04175717
hypocrisy	0,02858448	0,0288299	0,05748255	-6,82E+09
iplanet	0,03970658	0,03990704	0,0060436	0,07357003
buxton	-0,0851517	-0,08536537	-0,08577266	-0,0847444
heros	0,04054271	0,04054271	0,04489611	0,03618931
stroud	0,01575466	0,0164787	0,01162629	0,02060707
impasse	-0,01753238	-0,01840727	-0,0262469	-0,00969275
quaint	-0,01084336	-0,01079894	-0,01083701	-0,01080529
dutchess	0,0032316	0,0016664	0,00541073	-0,00051274
optimizer	0,05472582	0,05439515	0,0584183	0,05070267
outperform	0,0176249	0,01766455	0,00466561	0,03062384

PROMEDIO	0,01328554	0,01309845	0,0128294	-227239674

-st

Consulta	ddfn	ddft	dnn	dnc
startling	0,02536446	0,02637763	0,0260188	0,02572329
proliferate	0,02812464	0,02788286	0,03569803	0,02030948
seagram	0,07202938	0,07221769	0,07676856	0,0674785
frustrate	-0,02104726	-0,02026662	-0,01558278	-0,0257311
coreldraw	0,00450831	0,00506305	0,01459918	-0,00502782
backdoor	-0,08644349	-0,08674709	-0,06948942	-0,10370116
sparring	0,04880821	0,04965555	0,05202535	0,04643841
discerning	-0,0427915	-0,04188482	-0,03876254	-0,04591378
summertime	0,07663401	0,07628994	0,04477008	0,10815388
colocation	-0,00098331	-7,51E-06	0,02574611	-0,02673692
battleground	0,00382902	0,00222779	-0,00512392	0,01118073
excavator	0,03044417	0,03114731	0,01214146	0,04945001
whomever	0,03316263	0,03192117	0,01328367	0,05180012
relaunch	0,08348182	0,0838382	0,10507246	0,06224756
beeper	0,05444627	0,05387844	0,06747743	0,04084728
invoicing	-0,05287269	-0,05301736	-0,06817305	-0,037717
komputer	0,01276522	0,01111207	0,03057616	-0,00669887
mcclain	0,03726219	0,03672042	0,01124339	0,06273922
imaginable	-0,03670367	-0,03665741	-0,02628866	-0,04707243
unanticipated	0,03478836	0,03466305	0,02499304	0,04445837
hypocrisy	0,03720141	0,03724231	0,06367256	0,01077116
iplanet	0,04226716	0,04087852	0,00603495	0,07711073
buxton	-0,08425031	-0,08495139	-0,07906896	-0,09013274
heros	0,04474921	0,04454223	0,04728647	0,04200497
stroud	0,0176831	0,01787803	0,01464773	0,02091339
impasse	-0,01586525	-0,01690548	-0,02503444	-0,00773629
quaint	-0,01585579	-0,0154751	-0,00696665	-0,02436425
dutchess	0,00393324	0,00225335	0,00928325	-0,00309666
optimizer	0,05238358	0,05243869	0,05712317	0,04769909
outperform	0,01570182	0,01551018	0,0005419	0,0306701
PROMEDIO	**0,01342516**	**0,01326086**	**0,01348378**	**0,01320224**

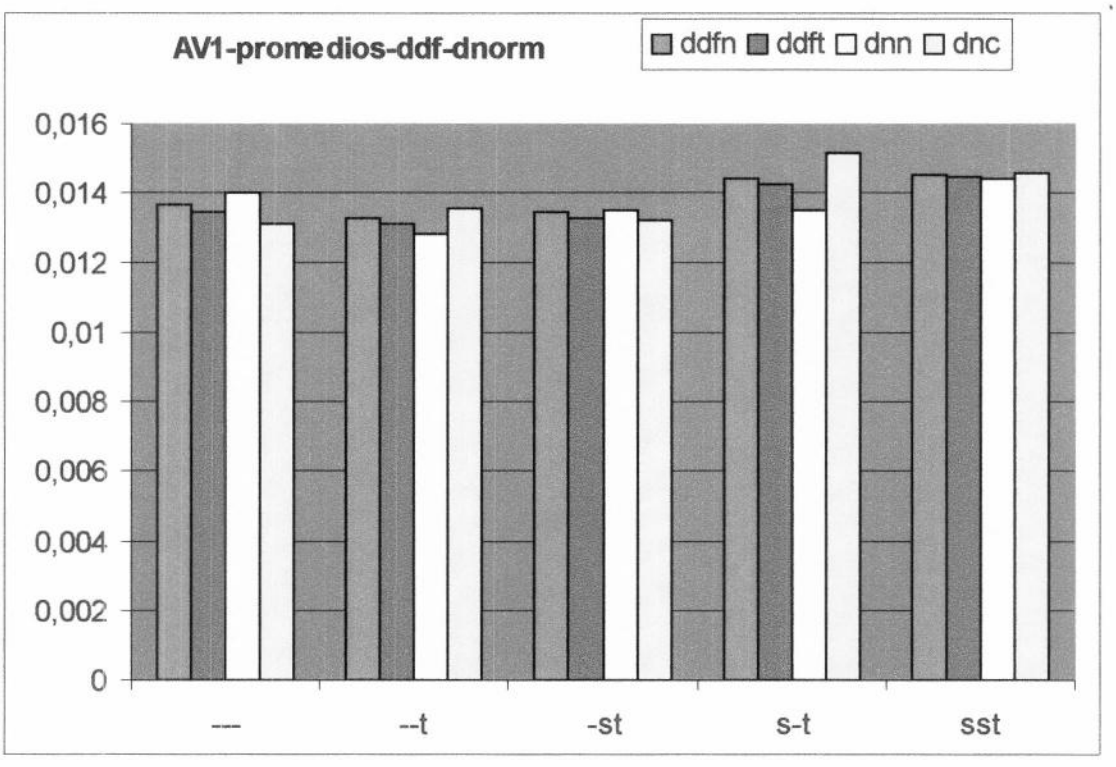
s-t

Consulta	ddfn	ddft	dnn	dnc
startling	0,03712147	0,03662895	0,03326579	0,04048463
proliferate	0,01518233	0,01578678	0,02390773	0,00706138
seagram	0,07382619	0,07342602	0,07827506	0,06897715
frustrate	-0,00781391	-0,00743109	0,00126103	-0,01650603
coreldraw	0,01106826	0,01099782	0,01713511	0,00493097
backdoor	-0,09359483	-0,09329123	-0,08004102	-0,10684505
sparring	0,04158859	0,04225682	0,04331772	0,04052769
discerning	-0,03526533	-0,03542894	-0,03671739	-0,03397688
summertime	0,08116769	0,08112721	0,05063957	0,11165533
colocation	-0,00218429	-0,00083318	0,02310395	-0,02612142
battleground	0,01241298	0,01068644	0,00318853	0,01991089
excavator	0,04305207	0,04173629	0,01282373	0,07196463
whomever	0,03502483	0,03433426	0,0168684	0,05249069
relaunch	0,07033292	0,0709937	0,08921359	0,05211303
beeper	0,03918694	0,04040164	0,05320281	0,02638577

invoicing	-0,04886332	-0,04851888	-0,06275834	-0,03462386
komputer	0,00649761	0,00574291	0,02383417	-0,01159364
mcclain	0,03862012	0,03771952	0,01386075	0,06247889
imaginable	-0,01686492	-0,01714248	0,00111023	-0,03511763
unanticipated	0,03252576	0,03249095	0,02337789	0,04163882
hypocrisy	0,0305342	0,03123637	0,0590505	0,00272006
iplanet	0,0407386	0,04001841	0,00617724	0,07457977
buxton	-0,0823073	-0,08287485	-0,08637359	-0,07880856
heros	0,00707094	0,00705091	0,01196517	0,00215667
stroud	0,0169521	0,01817043	0,01292119	0,02220134
impasse	-0,01718104	-0,01782171	-0,02546156	-0,0095412
quaint	-0,00932694	-0,00863535	-0,01125577	-0,00670651
dutchess	0,00449995	0,00202396	0,00493847	0,00158544
optimizer	0,08599476	0,08604299	0,09366217	0,07837559
outperform	0,02154375	0,02089611	0,01078509	0,03165477
PROMEDIO	**0,01438467**	**0,01425969**	**0,01350927**	**0,01513509**

sst				
Consulta	**ddfn**	**ddft**	**dnn**	**dnc**
startling	0,02859394	0,02906535	0,02888945	0,02876984
proliferate	0,02515929	0,02424195	0,03262601	0,01677523
seagram	0,07023257	0,07013056	0,07548177	0,06488136
frustrate	-0,00425599	-0,00302498	0,00394824	-0,01122921
coreldraw	0,01054875	0,0112972	0,02009369	0,00175225
backdoor	-0,09526797	-0,09429647	-0,07854329	-0,11102116
sparring	0,04758887	0,04867043	0,04884266	0,04741664
discerning	-0,04273015	-0,04165985	-0,04045321	-0,04393679
summertime	0,0804593	0,07960249	0,04904739	0,11101441
colocation	0,00195911	0,0024395	0,02673692	-0,02233832
battleground	0,00754664	0,00581314	-0,00146199	0,01482178
excavator	0,02465887	0,02471456	-0,00036202	0,04973545
whomever	0,03248759	0,03199876	0,0144941	0,04999224
relaunch	0,07713382	0,07780946	0,0992813	0,05566197
beeper	0,05600598	0,05541659	0,06572365	0,04569892
invoicing	-0,04936622	-0,04867043	-0,06490769	-0,03312896
komputer	0,00790639	0,00707262	0,02522857	-0,01024955
mcclain	0,0365586	0,03556653	0,01265057	0,05947456
imaginable	-0,02059873	-0,02059873	-0,00387259	-0,03732487
unanticipated	0,03366054	0,03418964	0,02457533	0,04327485
hypocrisy	0,03977149	0,04014643	0,06593586	0,01398206
iplanet	0,04338558	0,04215457	0,00624512	0,07929502
buxton	-0,08342904	-0,08419689	-0,08191336	-0,08571257
heros	0,0067037	0,00677715	0,00965493	0,00382592
stroud	0,01962545	0,02054442	0,01535784	0,02481203
impasse	-0,01419813	-0,0148388	-0,02233398	-0,00670295
quaint	-0,0136922	-0,01248668	-0,0062941	-0,01988478
dutchess	0,00300896	0,00048575	0,00631477	-0,00282006
optimizer	0,08573987	0,08596032	0,09385506	0,07784514
outperform	0,0198784	0,02007005	0,00749405	0,0324544
PROMEDIO	**0,01450251**	**0,01447982**	**0,01441117**	**0,01457116**

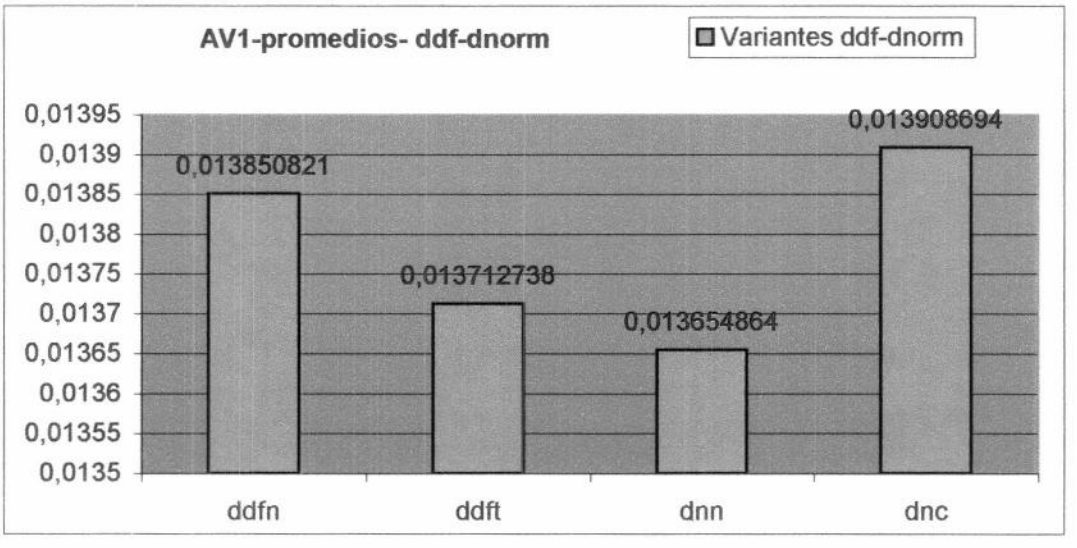
Gráficamente:



Consolidemos las 5 tablas anteriores:

Index id	ddfn	ddft	dnn	dnc	Prom
---	0,01365622	0,01346486	0,0140407	0,01308038	**0,01356054**
--t	0,01328554	0,01309845	0,0128294	0,0135546	**0,013192**
-st	0,01342516	0,01326086	0,01348378	0,01320224	**0,01334301**
s-t	0,01438467	0,01425969	0,01350927	0,01513509	**0,01432218**
sst	0,01450251	0,01447982	0,01441117	0,01457116	**0,01449117**
Promedio	**0,01385082**	**0,01371274**	**0,01365486**	**0,01390869**	

Gráficamente:



Componente df: vemos que ante la variante n, que significa aplicar un factor 1, es decir, no modificar el peso aportado por los demás componentes, la variante t no sólo no mejora sino que disminuye las evaluaciones.

Componente de normalización: se observa que aplicar la normalización del coseno (variante c) mejora el desempeño, respecto de no aplicar ninguna normalización. El efecto de normalizar los pesos de los términos de un documento es amortiguar la importancia de los documentos demasiado extensos, en donde las frecuencias absolutas de los términos (cantidad de apariciones) pueden ser muy grandes, lo que provocaría que ese documento tuviera un puntaje alto sólo por ser extenso.

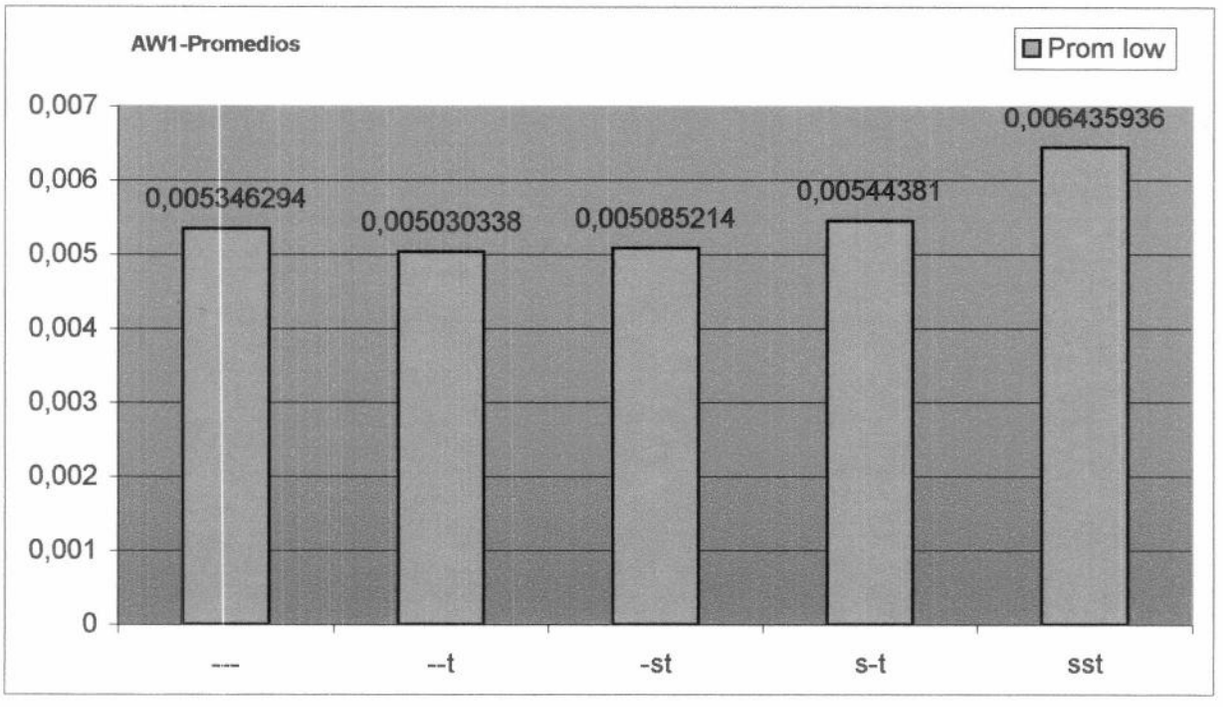
Terminamos así nuestro análisis de AltaVista.

5.2.4.3. AlltheWeb: variantes de indexación

Analicemos las mismas 30 consultas, en el buscador AlltheWeb (AW) o también llamado FAST. Los mismos comentarios que se hicieron para describir las tablas del análisis de AltaVista, se aplican aquí. La siguiente tabla promedia las 30 consultas en las 5 variantes de indexación estudiadas.

Consulta	—	–t	-st	s-t	sst
backdoor	-0,029360325	-0,043224194	-0,039846424	-0,042815108	-0,039437339
sparring	0,000402855	-0,015005371	-0,007738643	-0,010808011	-0,002915899
discerning	-0,016149425	-0,007422003	-0,001075534	-0,013813629	-0,009906609
summertime	0,02987896	0,035643436	0,028843942	0,042180582	0,034971393
startling	-0,016919671	-0,013178512	-0,009509229	-0,012312403	-0,009214536
colocation	0,055140454	0,045468929	0,048204516	0,039349584	0,042105585
battleground	0,03731179	0,047158642	0,033183322	0,053510716	0,043470342
excavator	0,065314808	0,048418453	0,044868042	0,027656428	0,026699394
whomever	-0,004671266	-0,003092532	-0,001176948	-0,002844968	0,001347403
proliferate	0,03954554	0,048251051	0,050648107	0,052994162	0,055410834
seagram	-0,018470278	-0,017497332	-0,026069456	-0,016818624	-0,024998431
frustrate	0,033447036	0,041042143	0,043886107	0,028065768	0,029510855
relaunch	-0,048031389	-0,051400947	-0,048551242	-0,042123277	-0,038803048
coreldraw	-0,017124382	-0,002027027	-0,021516274	-0,006469776	-0,021904969
beeper	0,033172192	0,012868324	0,021784837	-0,000832566	0,005248619
invoicing	0,04513426	0,042916429	0,04622929	0,085785247	0,094807338
komputer	-0,003847403	-0,003019481	-0,0015625	-0,014971591	-0,014330357
mcclain	0,001638779	-0,004438359	0,002702547	-0,003669283	0,007874044
imaginable	-0,017785527	-0,024048242	-0,022929381	-0,016971811	-0,016230747
unanticipated	0,000833563	0,006754616	0,004860154	0,007619179	0,006561725
hypocrisy	0,032677685	0,029166667	0,036111111	0,026295779	0,033096679
iplanet	0,011299493	0,01897224	0,016477149	0,018314382	0,01877795
buxton	0,033653846	0,045003153	0,038011169	0,046812136	0,042259653
heros	0,012753223	0,000218692	0,002812308	-0,023449969	-0,02199202
stroud	0,016426853	0,012212484	0,011404134	0,014993912	0,013943336
impasse	0,047283351	0,046073147	0,044398907	0,046726509	0,045675933
quaint	-0,032914012	-0,031957561	-0,034858915	-0,033682017	-0,036426926
dutchess	0,006158678	0,009926645	0,012610626	0,006719233	0,009547992
optimizer	-0,105196093	-0,100124853	-0,09829245	-0,071702409	-0,063715482
outperform	-0,031214779	-0,022748485	-0,021352846	-0,020423887	-0,018354642
Promedio	**0,005346294**	**0,005030338**	**0,005085214**	**0,00544381**	**0,006435936**

Gráficamente:



Cada barra representa la evaluación promedio de las 30 consultas en cada variante de indexación. Nuevamente, nos interesa observar los cambios o tendencias al aplicar diferentes variantes, tanto de indexación como de relevancia.

Analicemos la influencia de *stemming*, *stopwords* y *tags*.

Variante	**Id post**	**Id pre**	**Diferencia**	**Pares mejorados**
Stemming	s-t	--t	0,000413471	4,11403821
	sst	-st	0,001350721	13,439678
Stopwords	-st	--t	5,48759E-05	0,54601522
	sst	s-t	0,000992126	9,87165498
Stop-stem	sst	--t	0,001405597	13,9856932
Tags	--t	---	-0,000315955	-3,14375676

Aplicar *stemming* resulta positivo, y vemos que la cantidad de pares mejorados por *ranking*, en este caso 4.11 y 13.43 son valores disímiles, por lo que no es lo mismo aplicar *stemming* antes de eliminar las *stopwords*, que después de descartar tales palabras.

Lo expresado acerca de la remoción de *stopwords* al hacer el análisis de variantes de indexación en AltaVista se aplica también aquí: pierde importancia en consultas de una sola palabra.

La aplicación conjunta de descarte de *stopwords* y lematización de palabras (*stemming*) produce la mayor diferencia, con 13.98 pares.

Tener en cuenta el texto de atributos en algunos tags, no aporta sino que más bien empeora las evaluaciones.

5.2.4.4. AlltheWeb: variantes de relevancia

Abordaremos, tal como hicimos con AltaVista, el análisis de la influencia de las variantes de cada componente en las fórmulas de cálculo de pesos de términos en consulta y páginas.

Recordemos que un esquema de relevancia se identifica con dos ternas, una para consultas y otra para documentos. Cada terna tiene 3 componentes: tf, df y normalización.

El componente tf tiene a {b, n, a, l} como posibles variantes. El df a {n, t}; y el de normalización a {n, c}. Las fórmulas asociadas a cada variante fueron expuestas oportunamente.

5.2.4.4.1. Componente tf de consultas

Al igual que con AV, mostraremos las 5 tablas con las evaluaciones para las 5 variantes de indexación. Cada columna se refiere a una variante del componente tf: {b, n, a, l}. Cada valor promedia 64 evaluaciones realizadas con cada variante y consulta. Es evidente que, en una misma fila, tenemos 4 evaluaciones promedio que abarcan los 256 esquemas de relevancia que fueron probados por cada consulta y variante de indexación. En la columna qtfb se promedian los esquemas b??-???, en la qtfn los n??-???, y así. Vemos que los esquemas que contempla cada variante no se solapan entre sí, pues son variantes del mismo componente.

Consulta	qtfb	qtfn	qtfa	qtfl
backdoor	-0,02936033	-0,02936033	-0,02936033	-0,02936033
sparring	0,00040285	0,00040285	0,00040285	0,00040285
discerning	-0,01614943	-0,01614943	-0,01614943	-0,01614943
summertime	0,02987896	0,02987896	0,02987896	0,02987896
startling	-0,01691967	-0,01691967	-0,01691967	-0,01691967
colocation	0,05514045	0,05514045	0,05514045	0,05514045
battleground	0,03731179	0,03731179	0,03731179	0,03731179
excavator	0,06531481	0,06531481	0,06531481	0,06531481
whomever	-0,00467127	-0,00467127	-0,00467127	-0,00467127
proliferate	0,03954554	0,03954554	0,03954554	0,03954554
seagram	-0,01847028	-0,01847028	-0,01847028	-0,01847028
frustrate	0,03344704	0,03344704	0,03344704	0,03344704
relaunch	-0,04803139	-0,04803139	-0,04803139	-0,04803139
coreldraw	-0,01712438	-0,01712438	-0,01712438	-0,01712438
beeper	0,03317219	0,03317219	0,03317219	0,03317219
invoicing	0,04513426	0,04513426	0,04513426	0,04513426
komputer	-0,0038474	-0,0038474	-0,0038474	-0,0038474
mcclain	0,00163878	0,00163878	0,00163878	0,00163878
imaginable	-0,01778553	-0,01778553	-0,01778553	-0,01778553
unanticipated	0,00083356	0,00083356	0,00083356	0,00083356
hypocrisy	0,03267768	0,03267768	0,03267768	0,03267768
iplanet	0,01129949	0,01129949	0,01129949	0,01129949
buxton	0,03365385	0,03365385	0,03365385	0,03365385
heros	0,01275322	0,01275322	0,01275322	0,01275322
stroud	0,01642685	0,01642685	0,01642685	0,01642685
impasse	0,04728335	0,04728335	0,04728335	0,04728335
quaint	-0,03291401	-0,03291401	-0,03291401	-0,03291401

dutchess	0,00615868	0,00615868	0,00615868	0,00615868
optimizer	-0,10519609	-0,10519609	-0,10519609	-0,10519609
outperform	-0,03121478	-0,03121478	-0,03121478	-0,03121478
Promedio	**0,00534629**	**0,00534629**	**0,00534629**	**0,00534629**

-t

Consulta	**qtfb**	**qtfn**	**qtfa**	**qtfl**
backdoor	-0,04322419	-0,04322419	-0,04322419	-0,04322419
sparring	-0,01500537	-0,01500537	-0,01500537	-0,01500537
discerning	-0,007422	-0,007422	-0,007422	-0,007422
summertime	0,03564344	0,03564344	0,03564344	0,03564344
startling	-0,01317851	-0,01317851	-0,01317851	-0,01317851
colocation	0,04546893	0,04546893	0,04546893	0,04546893
battleground	0,04715864	0,04715864	0,04715864	0,04715864
excavator	0,04841845	0,04841845	0,04841845	0,04841845
whomever	-0,00309253	-0,00309253	-0,00309253	-0,00309253
proliferate	0,04825105	0,04825105	0,04825105	0,04825105
seagram	-0,01749733	-0,01749733	-0,01749733	-0,01749733
frustrate	0,04104214	0,04104214	0,04104214	0,04104214
relaunch	-0,05140095	-0,05140095	-0,05140095	-0,05140095
coreldraw	-0,00202703	-0,00202703	-0,00202703	-0,00202703
beeper	0,01286832	0,01286832	0,01286832	0,01286832
invoicing	0,04291643	0,04291643	0,04291643	0,04291643
komputer	-0,00301948	-0,00301948	-0,00301948	-0,00301948
mcclain	-0,00443836	-0,00443836	-0,00443836	-0,00443836
imaginable	-0,02404824	-0,02404824	-0,02404824	-0,02404824
unanticipated	0,00675462	0,00675462	0,00675462	0,00675462
hypocrisy	0,02916667	0,02916667	0,02916667	0,02916667
iplanet	0,01897224	0,01897224	0,01897224	0,01897224
buxton	0,04500315	0,04500315	0,04500315	0,04500315
heros	0,00021869	0,00021869	0,00021869	0,00021869
stroud	0,01221248	0,01221248	0,01221248	0,01221248
impasse	0,04607315	0,04607315	0,04607315	0,04607315
quaint	-0,03195756	-0,03195756	-0,03195756	-0,03195756
dutchess	0,00992665	0,00992665	0,00992665	0,00992665
optimizer	-0,10012485	-0,10012485	-0,10012485	-0,10012485
outperform	-0,02274849	-0,02274849	-0,02274849	-0,02274849
Promedio	**0,00503034**	**0,00503034**	**0,00503034**	**0,00503034**

-st

Consulta	**qtfb**	**qtfn**	**qtfa**	**qtfl**
backdoor	-0,03984642	-0,03984642	-0,03984642	-0,03984642
sparring	-0,00773864	-0,00773864	-0,00773864	-0,00773864
discerning	-0,00107553	-0,00107553	-0,00107553	-0,00107553
summertime	0,02884394	0,02884394	0,02884394	0,02884394
startling	-0,00950923	-0,00950923	-0,00950923	-0,00950923
colocation	0,04820452	0,04820452	0,04820452	0,04820452
battleground	0,03318332	0,03318332	0,03318332	0,03318332
excavator	0,04486804	0,04486804	0,04486804	0,04486804
whomever	-0,00117695	-0,00117695	-0,00117695	-0,00117695
proliferate	0,05064811	0,05064811	0,05064811	0,05064811
seagram	-0,02606946	-0,02606946	-0,02606946	-0,02606946
frustrate	0,04388611	0,04388611	0,04388611	0,04388611
relaunch	-0,04855124	-0,04855124	-0,04855124	-0,04855124

coreldraw	-0,02151627	-0,02151627	-0,02151627	-0,02151627
beeper	0,02178484	0,02178484	0,02178484	0,02178484
invoicing	0,04622929	0,04622929	0,04622929	0,04622929
komputer	-0,0015625	-0,0015625	-0,0015625	-0,0015625
mcclain	0,00270255	0,00270255	0,00270255	0,00270255
imaginable	-0,02292938	-0,02292938	-0,02292938	-0,02292938
unanticipated	0,00486015	0,00486015	0,00486015	0,00486015
hypocrisy	0,03611111	0,03611111	0,03611111	0,03611111
iplanet	0,01647715	0,01647715	0,01647715	0,01647715
buxton	0,03801117	0,03801117	0,03801117	0,03801117
heros	0,00281231	0,00281231	0,00281231	0,00281231
stroud	0,01140413	0,01140413	0,01140413	0,01140413
impasse	0,04439891	0,04439891	0,04439891	0,04439891
quaint	-0,03485891	-0,03485891	-0,03485891	-0,03485891
dutchess	0,01261063	0,01261063	0,01261063	0,01261063
optimizer	-0,09829245	-0,09829245	-0,09829245	-0,09829245
outperform	-0,02135285	-0,02135285	-0,02135285	-0,02135285
Promedio	**0,00508521**	**0,00508521**	**0,00508521**	**0,00508521**

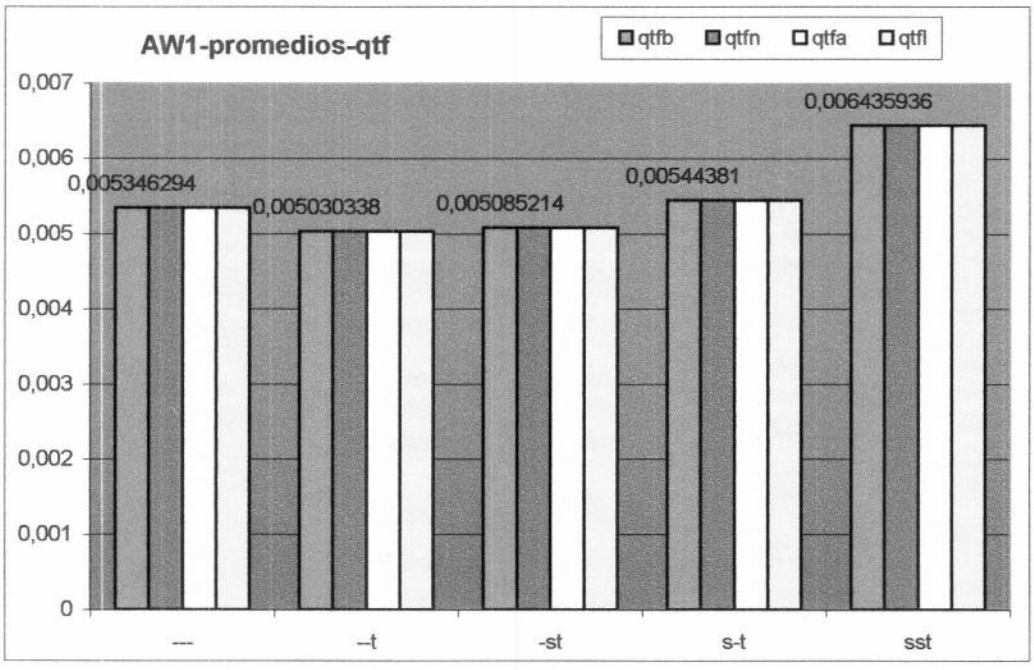
s-t

query	**qtfb**	**qtfn**	**qtfa**	**qtfl**
backdoor	-0,04281511	-0,04281511	-0,04281511	-0,04281511
sparring	-0,01080801	-0,01080801	-0,01080801	-0,01080801
discerning	-0,01381363	-0,01381363	-0,01381363	-0,01381363
summertime	0,04218058	0,04218058	0,04218058	0,04218058
startling	-0,0123124	-0,0123124	-0,0123124	-0,0123124
colocation	0,03934958	0,03934958	0,03934958	0,03934958
battleground	0,05351072	0,05351072	0,05351072	0,05351072
excavator	0,02765643	0,02765643	0,02765643	0,02765643
whomever	-0,00284497	-0,00284497	-0,00284497	-0,00284497
proliferate	0,05299416	0,05299416	0,05299416	0,05299416
seagram	-0,01681862	-0,01681862	-0,01681862	-0,01681862
frustrate	0,02806577	0,02806577	0,02806577	0,02806577
relaunch	-0,04212328	-0,04212328	-0,04212328	-0,04212328
coreldraw	-0,00646978	-0,00646978	-0,00646978	-0,00646978
beeper	-0,00083257	-0,00083257	-0,00083257	-0,00083257
invoicing	0,08578525	0,08578525	0,08578525	0,08578525
komputer	-0,01497159	-0,01497159	-0,01497159	-0,01497159
mcclain	-0,00366928	-0,00366928	-0,00366928	-0,00366928
imaginable	-0,01697181	-0,01697181	-0,01697181	-0,01697181
unanticipated	0,00761918	0,00761918	0,00761918	0,00761918
hypocrisy	0,02629578	0,02629578	0,02629578	0,02629578
iplanet	0,01831438	0,01831438	0,01831438	0,01831438
buxton	0,04681214	0,04681214	0,04681214	0,04681214
heros	-0,02344997	-0,02344997	-0,02344997	-0,02344997
stroud	0,01499391	0,01499391	0,01499391	0,01499391
impasse	0,04672651	0,04672651	0,04672651	0,04672651
quaint	-0,03368202	-0,03368202	-0,03368202	-0,03368202
dutchess	0,00671923	0,00671923	0,00671923	0,00671923
optimizer	-0,07170241	-0,07170241	-0,07170241	-0,07170241
outperform	-0,02042389	-0,02042389	-0,02042389	-0,02042389
Promedio	**0,00544381**	**0,00544381**	**0,00544381**	**0,00544381**

sst

query	qtfb	qtfn	qtfa	qtfl
backdoor	-0,03943734	-0,03943734	-0,03943734	-0,03943734
sparring	-0,0029159	-0,0029159	-0,0029159	-0,0029159
discerning	-0,00990661	-0,00990661	-0,00990661	-0,00990661
summertime	0,03497139	0,03497139	0,03497139	0,03497139
startling	-0,00921454	-0,00921454	-0,00921454	-0,00921454
colocation	0,04210559	0,04210559	0,04210559	0,04210559
battleground	0,04347034	0,04347034	0,04347034	0,04347034
excavator	0,02669939	0,02669939	0,02669939	0,02669939
whomever	0,0013474	0,0013474	0,0013474	0,0013474
proliferate	0,05541083	0,05541083	0,05541083	0,05541083
seagram	-0,02499843	-0,02499843	-0,02499843	-0,02499843
frustrate	0,02951085	0,02951085	0,02951085	0,02951085
relaunch	-0,03880305	-0,03880305	-0,03880305	-0,03880305
coreldraw	-0,02190497	-0,02190497	-0,02190497	-0,02190497
beeper	0,00524862	0,00524862	0,00524862	0,00524862
invoicing	0,09480734	0,09480734	0,09480734	0,09480734
komputer	-0,01433036	-0,01433036	-0,01433036	-0,01433036
mcclain	0,00787404	0,00787404	0,00787404	0,00787404
imaginable	-0,01623075	-0,01623075	-0,01623075	-0,01623075
unanticipated	0,00656172	0,00656172	0,00656172	0,00656172
hypocrisy	0,03309668	0,03309668	0,03309668	0,03309668
iplanet	0,01877795	0,01877795	0,01877795	0,01877795
buxton	0,04225965	0,04225965	0,04225965	0,04225965
heros	-0,02199202	-0,02199202	-0,02199202	-0,02199202
stroud	0,01394334	0,01394334	0,01394334	0,01394334
impasse	0,04567593	0,04567593	0,04567593	0,04567593
quaint	-0,03642693	-0,03642693	-0,03642693	-0,03642693
dutchess	0,00954799	0,00954799	0,00954799	0,00954799
optimizer	-0,06371548	-0,06371548	-0,06371548	-0,06371548
outperform	-0,01835464	-0,01835464	-0,01835464	-0,01835464
Promedio	**0,00643594**	**0,00643594**	**0,00643594**	**0,00643594**

Gráficamente:



Confirmamos que los promedios por variante de indexación coinciden con los promedios que obtuvimos sin separar en variantes de relevancia.

Volvemos a observar empíricamente que las variantes de tf en consultas no influyen en las evaluaciones (fue probado analíticamente en la sección correspondiente de análisis de AltaVista).

5.2.4.4.2. Componentes df y normalización en consultas

Deberíamos observar influencia nula de las variantes de estos componentes. Estas son las evaluaciones.

Consulta	qdfn	qdft	qnn	qnc
backdoor	-0,02936033	-0,02936033	-0,02936033	-0,02936033
sparring	0,00040285	0,00040285	0,00040285	0,00040285
discerning	-0,01614943	-0,01614943	-0,01614943	-0,01614943
summertime	0,02987896	0,02987896	0,02987896	0,02987896
startling	-0,01691967	-0,01691967	-0,01691967	-0,01691967
colocation	0,05514045	0,05514045	0,05514045	0,05514045
battleground	0,03731179	0,03731179	0,03731179	0,03731179
excavator	0,06531481	0,06531481	0,06531481	0,06531481
whomever	-0,00467127	-0,00467127	-0,00467127	-0,00467127
proliferate	0,03954554	0,03954554	0,03954554	0,03954554
seagram	-0,01847028	-0,01847028	-0,01847028	-0,01847028
frustrate	0,03344704	0,03344704	0,03344704	0,03344704
relaunch	-0,04803139	-0,04803139	-0,04803139	-0,04803139
coreldraw	-0,01712438	-0,01712438	-0,01712438	-0,01712438
beeper	0,03317219	0,03317219	0,03317219	0,03317219
invoicing	0,04513426	0,04513426	0,04513426	0,04513426

komputer	-0,0038474	-0,0038474	-0,0038474	-0,0038474
mcclain	0,00163878	0,00163878	0,00163878	0,00163878
imaginable	-0,01778553	-0,01778553	-0,01778553	-0,01778553
unanticipated	0,00083356	0,00083356	0,00083356	0,00083356
hypocrisy	0,03267768	0,03267768	0,03267768	0,03267768
iplanet	0,01129949	0,01129949	0,01129949	0,01129949
buxton	0,03365385	0,03365385	0,03365385	0,03365385
heros	0,01275322	0,01275322	0,01275322	0,01275322
stroud	0,01642685	0,01642685	0,01642685	0,01642685
impasse	0,04728335	0,04728335	0,04728335	0,04728335
quaint	-0,03291401	-0,03291401	-0,03291401	-0,03291401
dutchess	0,00615868	0,00615868	0,00615868	0,00615868
optimizer	-0,10519609	-0,10519609	-0,10519609	-0,10519609
outperform	-0,03121478	-0,03121478	-0,03121478	-0,03121478
Promedio	**0,00534629**	**0,00534629**	**0,00534629**	**0,00534629**

–t

Consulta	**qdfn**	**qdft**	**qnn**	**qnc**
backdoor	-0,04322419	-0,04322419	-0,04322419	-0,04322419
sparring	-0,01500537	-0,01500537	-0,01500537	-0,01500537
discerning	-0,007422	-0,007422	-0,007422	-0,007422
summertime	0,03564344	0,03564344	0,03564344	0,03564344
startling	-0,01317851	-0,01317851	-0,01317851	-0,01317851
colocation	0,04546893	0,04546893	0,04546893	0,04546893
battleground	0,04715864	0,04715864	0,04715864	0,04715864
excavator	0,04841845	0,04841845	0,04841845	0,04841845
whomever	-0,00309253	-0,00309253	-0,00309253	-0,00309253
proliferate	0,04825105	0,04825105	0,04825105	0,04825105
seagram	-0,01749733	-0,01749733	-0,01749733	-0,01749733
frustrate	0,04104214	0,04104214	0,04104214	0,04104214
relaunch	-0,05140095	-0,05140095	-0,05140095	-0,05140095
coreldraw	-0,00202703	-0,00202703	-0,00202703	-0,00202703
beeper	0,01286832	0,01286832	0,01286832	0,01286832
invoicing	0,04291643	0,04291643	0,04291643	0,04291643
komputer	-0,00301948	-0,00301948	-0,00301948	-0,00301948
mcclain	-0,00443836	-0,00443836	-0,00443836	-0,00443836
imaginable	-0,02404824	-0,02404824	-0,02404824	-0,02404824
unanticipated	0,00675462	0,00675462	0,00675462	0,00675462
hypocrisy	0,02916667	0,02916667	0,02916667	0,02916667
iplanet	0,01897224	0,01897224	0,01897224	0,01897224
buxton	0,04500315	0,04500315	0,04500315	0,04500315
heros	0,00021869	0,00021869	0,00021869	0,00021869
stroud	0,01221341	0,01221156	0,01221156	0,01221341
impasse	0,04607315	0,04607315	0,04607315	0,04607315
quaint	-0,03195756	-0,03195756	-0,03195756	-0,03195756
dutchess	0,00992665	0,00992665	0,00992665	0,00992665
optimizer	-0,10012485	-0,10012485	-0,10012485	-0,10012485
outperform	-0,02274849	-0,02274849	-0,02274849	-0,02274849
Promedio	**0,00503037**	**0,00503031**	**0,00503031**	**0,00503037**

-st

Consulta	**qdfn**	**qdft**	**qnn**	**qnc**
backdoor	-0,03984642	-0,03984642	-0,03984642	-0,03984642
sparring	-0,00773864	-0,00773864	-0,00773864	-0,00773864

discerning	-0,00107553	-0,00107553	-0,00107553	-0,00107553
summertime	0,02884394	0,02884394	0,02884394	0,02884394
startling	-0,00950923	-0,00950923	-0,00950923	-0,00950923
colocation	0,04820452	0,04820452	0,04820452	0,04820452
battleground	0,03318332	0,03318332	0,03318332	0,03318332
excavator	0,04486804	0,04486804	0,04486804	0,04486804
whomever	-0,00117695	-0,00117695	-0,00117695	-0,00117695
proliferate	0,05064811	0,05064811	0,05064811	0,05064811
seagram	-0,02606946	-0,02606946	-0,02606946	-0,02606946
frustrate	0,04388611	0,04388611	0,04388611	0,04388611
relaunch	-0,04855124	-0,04855124	-0,04855124	-0,04855124
coreldraw	-0,02151627	-0,02151627	-0,02151627	-0,02151627
beeper	0,02178484	0,02178484	0,02178484	0,02178484
invoicing	0,04622929	0,04622929	0,04622929	0,04622929
komputer	-0,0015625	-0,0015625	-0,0015625	-0,0015625
mcclain	0,00270255	0,00270255	0,00270255	0,00270255
imaginable	-0,02292938	-0,02292938	-0,02292938	-0,02292938
unanticipated	0,00486015	0,00486015	0,00486015	0,00486015
hypocrisy	0,03611111	0,03611111	0,03611111	0,03611111
iplanet	0,01647715	0,01647715	0,01647715	0,01647715
buxton	0,03801117	0,03801117	0,03801117	0,03801117
heros	0,00281231	0,00281231	0,00281231	0,00281231
stroud	0,01140413	0,01140413	0,01140413	0,01140413
impasse	0,04439891	0,04439891	0,04439891	0,04439891
quaint	-0,03485891	-0,03485891	-0,03485891	-0,03485891
dutchess	0,01261063	0,01261063	0,01261063	0,01261063
optimizer	-0,09829245	-0,09829245	-0,09829245	-0,09829245
outperform	-0,02135285	-0,02135285	-0,02135285	-0,02135285
Promedio	**0,00508521**	**0,00508521**	**0,00508521**	**0,00508521**

s-t

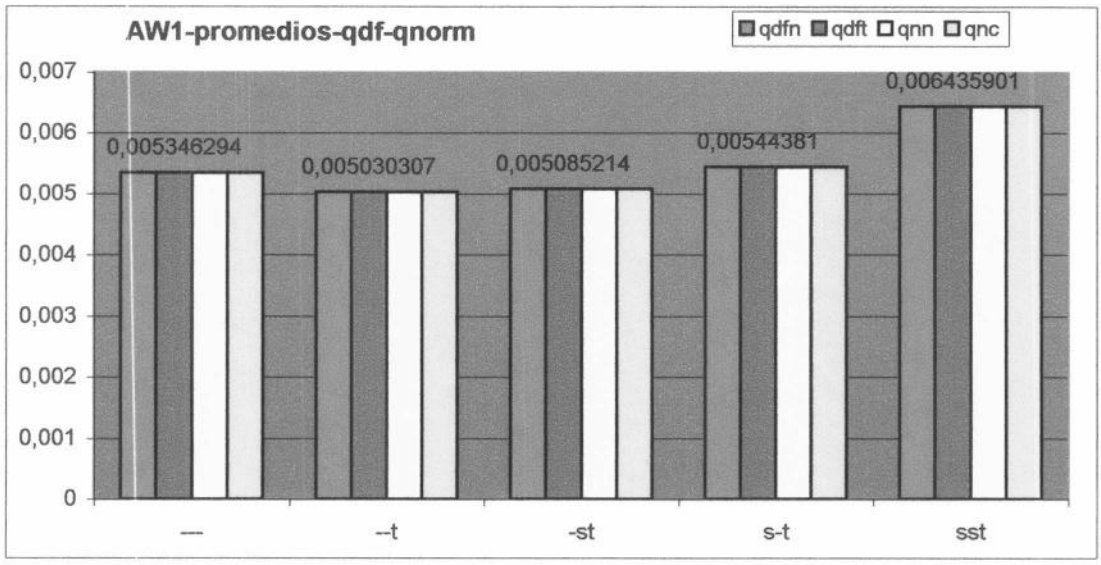
Consulta	**qdfn**	**qdft**	**qnn**	**qnc**
backdoor	-0,04281511	-0,04281511	-0,04281511	-0,04281511
sparring	-0,01080801	-0,01080801	-0,01080801	-0,01080801
discerning	-0,01381363	-0,01381363	-0,01381363	-0,01381363
summertime	0,04218058	0,04218058	0,04218058	0,04218058
startling	-0,0123124	-0,0123124	-0,0123124	-0,0123124
colocation	0,03934958	0,03934958	0,03934958	0,03934958
battleground	0,05351072	0,05351072	0,05351072	0,05351072
excavator	0,02765643	0,02765643	0,02765643	0,02765643
whomever	-0,00284497	-0,00284497	-0,00284497	-0,00284497
proliferate	0,05299416	0,05299416	0,05299416	0,05299416
seagram	-0,01681862	-0,01681862	-0,01681862	-0,01681862
frustrate	0,02806577	0,02806577	0,02806577	0,02806577
relaunch	-0,04212328	-0,04212328	-0,04212328	-0,04212328
coreldraw	-0,00646978	-0,00646978	-0,00646978	-0,00646978
beeper	-0,00083257	-0,00083257	-0,00083257	-0,00083257
invoicing	0,08578525	0,08578525	0,08578525	0,08578525
komputer	-0,01497159	-0,01497159	-0,01497159	-0,01497159
mcclain	-0,00366928	-0,00366928	-0,00366928	-0,00366928
imaginable	-0,01697181	-0,01697181	-0,01697181	-0,01697181
unanticipated	0,00761918	0,00761918	0,00761918	0,00761918
hypocrisy	0,02629578	0,02629578	0,02629578	0,02629578
iplanet	0,01831438	0,01831438	0,01831438	0,01831438

buxton	0,04681214	0,04681214	0,04681214	0,04681214
heros	-0,02344997	-0,02344997	-0,02344997	-0,02344997
stroud	0,01499391	0,01499391	0,01499391	0,01499391
impasse	0,04672651	0,04672651	0,04672651	0,04672651
quaint	-0,03368202	-0,03368202	-0,03368202	-0,03368202
dutchess	0,00671923	0,00671923	0,00671923	0,00671923
optimizer	-0,07170241	-0,07170241	-0,07170241	-0,07170241
outperform	-0,02042389	-0,02042389	-0,02042389	-0,02042389
Promedio	**0,00544381**	**0,00544381**	**0,00544381**	**0,00544381**

sst

Consulta	**qdfn**	**qdft**	**qnn**	**qnc**
backdoor	-0,03943734	-0,03943734	-0,03943734	-0,03943734
sparring	-0,0029159	-0,0029159	-0,0029159	-0,0029159
discerning	-0,00990558	-0,00990764	-0,00990764	-0,00990558
summertime	0,03497139	0,03497139	0,03497139	0,03497139
startling	-0,00921454	-0,00921454	-0,00921454	-0,00921454
colocation	0,04210559	0,04210559	0,04210559	0,04210559
battleground	0,04347034	0,04347034	0,04347034	0,04347034
excavator	0,02669939	0,02669939	0,02669939	0,02669939
whomever	0,0013474	0,0013474	0,0013474	0,0013474
proliferate	0,05541083	0,05541083	0,05541083	0,05541083
seagram	-0,02499843	-0,02499843	-0,02499843	-0,02499843
frustrate	0,02951085	0,02951085	0,02951085	0,02951085
relaunch	-0,03880305	-0,03880305	-0,03880305	-0,03880305
coreldraw	-0,02190497	-0,02190497	-0,02190497	-0,02190497
beeper	0,00524862	0,00524862	0,00524862	0,00524862
invoicing	0,09480734	0,09480734	0,09480734	0,09480734
komputer	-0,01433036	-0,01433036	-0,01433036	-0,01433036
mcclain	0,00787404	0,00787404	0,00787404	0,00787404
imaginable	-0,01623075	-0,01623075	-0,01623075	-0,01623075
unanticipated	0,00656172	0,00656172	0,00656172	0,00656172
hypocrisy	0,03309668	0,03309668	0,03309668	0,03309668
iplanet	0,01877795	0,01877795	0,01877795	0,01877795
buxton	0,04225965	0,04225965	0,04225965	0,04225965
heros	-0,02199202	-0,02199202	-0,02199202	-0,02199202
stroud	0,01394334	0,01394334	0,01394334	0,01394334
impasse	0,04567593	0,04567593	0,04567593	0,04567593
quaint	-0,03642693	-0,03642693	-0,03642693	-0,03642693
dutchess	0,00954799	0,00954799	0,00954799	0,00954799
optimizer	-0,06371548	-0,06371548	-0,06371548	-0,06371548
outperform	-0,01835464	-0,01835464	-0,01835464	-0,01835464
Promedio	**0,00643597**	**0,0064359**	**0,0064359**	**0,00643597**

Gráficamente:



Confirmamos empíricamente que los promedios coinciden con los calculados en general para cada variante de indexación. Asimismo, vemos que no influye cambiar de variante ni en el componente df ni en la normalización de consultas.

Estas observaciones, lejos de ser redundantes, nos confirman que las evaluaciones calculadas son consistentes entre sí, y a la vez con la teoría sobre la que se apoyan.

5.2.4.4.3. Componente tf de documentos

Cambiamos nuestro foco de análisis hacia la terna de documentos. Comenzamos con el componente tf.

Los valores calculados son los siguientes.

Consulta	dtfb	dtfn	dtfa	dtfl
backdoor	-0,00965292	-0,03873176	-0,02874857	-0,04030805
sparring	0,0026857	-0,01341314	0,02747084	-0,01513198
discerning	-0,0053202	-0,02449918	-0,00591133	-0,028867
summertime	0,05958542	1,44E-05	0,05313093	0,00678512
startling	-0,00635386	-0,01886033	-0,01995285	-0,02251164
colocation	-0,01020741	0,08121019	0,06857341	0,08098563
battleground	0,03963026	0,02814037	0,0517121	0,02976443
excavator	0,03428812	0,08276287	0,06294662	0,08126163
whomever	0,00868506	0,00194805	-0,03113636	0,00181818
proliferate	0,04120896	0,0205574	0,07083673	0,02557906
seagram	-0,00968238	-0,01787396	-0,02940807	-0,0169167
frustrate	0,02601156	0,03992472	0,02609558	0,04175628
relaunch	-0,03729282	-0,03280007	-0,08757817	-0,0344545
coreldraw	0,01547515	-0,04157222	-0,00732345	-0,03507701
beeper	0,02105586	0,02903622	0,04637815	0,03621854
invoicing	0,05100616	0,03656446	0,0521488	0,04081762
komputer	-0,02058442	0,01396104	-0,01988636	0,01112013
mcclain	-0,03071991	0,02683859	-0,00047438	0,01091082
imaginable	-0,01868643	-0,00261552	-0,04795118	-0,00188899

unanticipated	0,00925875	-0,01295123	0,01837972	-0,01135299
hypocrisy	0,01876164	0,03382992	0,04495655	0,03316263
iplanet	-0,00846695	0,02585079	0,00833061	0,01948353
buxton	0,03745571	0,02329911	0,0468234	0,02703717
heros	0,02085635	0,015531	-0,0031768	0,01780233
stroud	0,02176883	0,00666726	0,03162865	0,00564267
impasse	0,05403599	0,02277857	0,09099549	0,02132335
quaint	-0,02926954	-0,0259273	-0,05489817	-0,02156104
dutchess	-0,01326028	0,01639344	0,02242219	-0,00092065
optimizer	-0,03243243	-0,14211222	-0,12019683	-0,12604289
outperform	-0,00200761	-0,04575232	-0,03589039	-0,04120879
Promedio	**0,00759441**	**0,00293997**	**0,00767656**	**0,00317423**

--t

Consulta	**dtfb**	**dtfn**	**dtfa**	**dtfl**
backdoor	-0,00465382	-0,0643878	-0,04484177	-0,05901339
sparring	0,00356047	-0,03850522	0,01058932	-0,03566605
discerning	-0,01568144	-0,00563218	-0,00279146	-0,00558292
summertime	0,06095107	0,01055143	0,05764476	0,01342648
startling	-0,00495946	-0,0143465	-0,0169628	-0,01644529
colocation	0,01053405	0,06061163	0,05281316	0,05791687
battleground	0,03742942	0,04371319	0,06629834	0,04119361
excavator	0,0467033	0,04272503	0,06299165	0,04125383
whomever	0,00931818	0,00311688	-0,02694805	0,00214286
proliferate	0,04251146	0,03493189	0,07661164	0,03894922
seagram	-0,01191074	-0,01759149	-0,02244052	-0,01804658
frustrate	0,02747345	0,04935139	0,03409396	0,05324976
relaunch	-0,03383219	-0,04451764	-0,08117297	-0,04608099
coreldraw	0,01858471	-0,01698634	0,00258646	-0,01229294
beeper	0,02125537	-0,00541743	0,02995703	0,00567833
invoicing	0,04442011	0,04132546	0,04449946	0,04142068
komputer	-0,00668831	0,00186688	-0,00461039	-0,0026461
mcclain	-0,0271836	0,01526652	-0,00533322	-0,00050313
imaginable	-0,01838128	-0,01579483	-0,04736995	-0,0146469
unanticipated	0,01125654	-0,00440893	0,02507578	-0,00490493
hypocrisy	0,01080074	0,03432651	0,04112353	0,03041589
iplanet	-0,00989856	0,03708551	0,01938809	0,02931392
buxton	0,05715186	0,02281871	0,07246442	0,02757761
heros	0,01833947	-0,0024248	-0,01709638	0,00205648
stroud	0,03055952	-0,00561668	0,02216976	0,00173735
impasse	0,05549121	0,02096698	0,08954027	0,01829413
quaint	-0,01851747	-0,03029355	-0,05228126	-0,02673797
dutchess	-0,00038608	0,00528629	0,04065693	-0,00585056
optimizer	-0,03360752	-0,13513514	-0,11016451	-0,12159224
outperform	-0,00114469	-0,03405889	-0,0259228	-0,02986757
Promedio	**0,01064986**	**-0,00037244**	**0,00968562**	**0,00015832**

-st

Consulta	**dtfb**	**dtfn**	**dtfa**	**dtfl**
backdoor	-0,00355792	-0,06085991	-0,03835645	-0,05661142
sparring	0,00587784	-0,0339779	0,0247698	-0,02762431
discerning	-0,01059113	-0,00192118	0,0070936	0,00111658
summertime	0,06072106	0,00448508	0,04045196	0,00971767
startling	-0,00257317	-0,01013455	-0,01230522	-0,01302398

colocation	0,01155479	0,06404132	0,05428303	0,06293892
battleground	0,03365005	0,03123672	0,03239026	0,03545626
excavator	0,04454152	0,0413439	0,05637122	0,03721552
whomever	0,0087013	0,01064935	-0,02834416	0,00428571
proliferate	0,04337455	0,03932584	0,07956186	0,04033017
seagram	-0,01467265	-0,02620677	-0,03894922	-0,02444919
frustrate	0,02856567	0,0509309	0,04019357	0,05585428
relaunch	-0,03405986	-0,04120879	-0,07428207	-0,04465424
coreldraw	0,01397849	-0,03613775	-0,0436501	-0,02025574
beeper	0,02751688	-0,00035298	0,04350829	0,01646716
invoicing	0,03954802	0,04116676	0,06938361	0,03481876
komputer	-0,0087987	0,00818182	-0,00425325	-0,00137987
mcclain	-0,02512794	0,02532919	0,00943017	0,00117877
imaginable	-0,02378669	-0,00948852	-0,04010462	-0,01833769
unanticipated	0,01023698	-0,00644806	0,02234775	-0,00669606
hypocrisy	0,00876785	0,04039417	0,0632216	0,03206083
iplanet	-0,01048484	0,03944426	0,00692627	0,03002291
buxton	0,05833784	0,01589804	0,05392422	0,02388459
heros	0,01792511	-0,00099754	-0,00893186	0,00325353
stroud	0,02934189	-0,0085531	0,02135305	0,0034747
impasse	0,05506058	0,02129366	0,08291756	0,01832383
quaint	-0,02050859	-0,03333713	-0,0547275	-0,03086244
dutchess	0,0017225	0,00417261	0,04475529	-0,00020789
optimizer	-0,03535546	-0,13748531	-0,09923619	-0,12109283
outperform	-0,0014969	-0,03386517	-0,0231051	-0,02694421
Promedio	**0,0102803**	**-0,0001027**	**0,00955458**	**0,00060868**

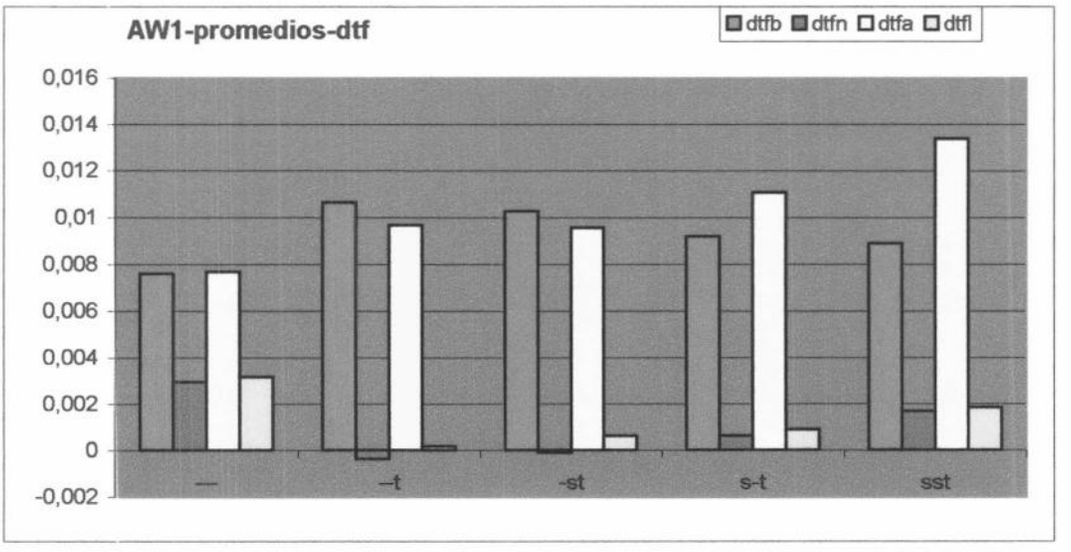
s-t

Consulta	**dtfb**	**dtfn**	**dtfa**	**dtfl**
backdoor	-0,00277728	-0,06216598	-0,05053144	-0,05578574
sparring	0,00320749	-0,0286372	0,0092388	-0,02704113
discerning	-0,02568144	-0,00852217	-0,00981938	-0,01123153
summertime	0,06211546	0,0197516	0,06394112	0,02291415
startling	-0,0083664	-0,00825139	-0,01955034	-0,01308148
colocation	0,01345337	0,04497387	0,04793402	0,05103707
battleground	0,04117843	0,05374598	0,06520551	0,05391294
excavator	0,04114874	0,01282051	0,04041314	0,01624332
whomever	0,00676948	0,00600649	-0,02724026	0,00308442
proliferate	0,04233884	0,04001632	0,08426966	0,04535183
seagram	-0,00827004	-0,01923922	-0,01958446	-0,02018078
frustrate	0,02100417	0,0297587	0,02559148	0,03590872
relaunch	-0,03272418	-0,03087244	-0,06895453	-0,03594196
coreldraw	0,01989247	-0,02380122	-0,00595757	-0,01601279
beeper	0,01135666	-0,01757213	0,01368938	-0,01080417
invoicing	0,04513426	0,10118708	0,10163144	0,09518822
komputer	-0,00967532	-0,01706169	-0,01074675	-0,0224026
mcclain	-0,02502731	0,01302398	-0,00199816	-0,00067564
imaginable	-0,01280151	-0,00913978	-0,04077303	-0,00517291
unanticipated	0,01237255	-0,0049876	0,02772114	-0,00462937
hypocrisy	0,00769708	0,03171943	0,03817505	0,02759156
iplanet	-0,00966678	0,03536758	0,01982439	0,02773233
buxton	0,06031946	0,02655678	0,06772053	0,03265177
heros	0,00615408	-0,03914979	-0,0199202	-0,04088398
stroud	0,02903005	-0,00329651	0,03015859	0,00408351

impasse	0,0556991	0,02239249	0,09063911	0,01817534
quaint	-0,01979747	-0,0328109	-0,0528786	-0,0292411
dutchess	-0,0008167	0,00179675	0,03302447	-0,00712758
optimizer	-0,04400705	-0,08551704	-0,07555817	-0,08172738
outperform	-0,00355734	-0,02977952	-0,02349253	-0,02486616
Promedio	**0,0091901**	**0,00061043**	**0,01107241**	**0,0009023**

sst				
Consulta	**dtfb**	**dtfn**	**dtfa**	**dtfl**
backdoor	-0,00294241	-0,06003423	-0,04283012	-0,05194259
sparring	0,00618478	-0,02490792	0,02447821	-0,01741866
discerning	-0,01986864	-0,00622742	-0,00899836	-0,00453202
summertime	0,06105169	0,01288023	0,04732333	0,01863033
startling	-0,00603761	-0,00465758	-0,01489276	-0,0112702
colocation	0,01339213	0,05071044	0,04672954	0,05759023
battleground	0,0402981	0,04287839	0,04079898	0,0499059
excavator	0,0385366	0,01161953	0,04505194	0,0115895
whomever	0,00769481	0,01396104	-0,02181818	0,00555195
proliferate	0,04384533	0,04230745	0,08828699	0,04720357
seagram	-0,0113615	-0,02846651	-0,03512021	-0,02504551
frustrate	0,02181073	0,03167428	0,02839763	0,03616077
relaunch	-0,03402951	-0,02650112	-0,06112258	-0,03355898
coreldraw	0,01618715	-0,03940715	-0,04135426	-0,02304563
beeper	0,01477901	-0,01401166	0,01909147	0,00113567
invoicing	0,03972259	0,1070431	0,14219514	0,09026852
komputer	-0,0098539	-0,01277597	-0,01275974	-0,02193182
mcclain	-0,0225979	0,02626358	0,02440918	0,00342131
imaginable	-0,01778553	-0,00075559	-0,03911654	-0,00726533
unanticipated	0,01164233	-0,0044916	0,02557178	-0,00647561
hypocrisy	0,00665736	0,03777157	0,06002483	0,02793296
iplanet	-0,00940772	0,0381899	0,01685209	0,02947753
buxton	0,06185072	0,01992134	0,05674653	0,03052003
heros	0,00465009	-0,03730816	-0,01591467	-0,03939533
stroud	0,03081195	-0,00583571	0,02525837	0,00553873
impasse	0,05413994	0,02366952	0,08660014	0,01829413
quaint	-0,02184549	-0,03528558	-0,05637729	-0,03219934
dutchess	0,00132157	0,00072761	0,03839986	-0,00225707
optimizer	-0,0487074	-0,08112515	-0,04559342	-0,07943596
outperform	-0,00353973	-0,02782474	-0,01937165	-0,02268245
Promedio	**0,00888665**	**0,00166673**	**0,01336487**	**0,00182549**

Gráficamente:



Consolidemos las 5 tablas anteriores:

Index id	dtfb	dtfn	dtfa	dtfl
—	0,00759441	0,00293997	0,00767656	0,00317423
–t	0,01064986	-0,00037244	0,00968562	0,00015832
-st	0,0102803	-0,0001027	0,00955458	0,00060868
s-t	0,0091901	0,00061043	0,01107241	0,0009023
sst	0,00888665	0,00166673	0,01336487	0,00182549
Promedio	**0,00932026**	**0,0009484**	**0,01027081**	**0,0013338**

Vemos que la mejor variante es la {a}, seguida de cerca por {b}, y con evaluaciones 10 veces más pequeñas {n}, {l}.

El factor 1 (b) o la frecuencia de un término normalizada por la máxima frecuencia en cada documento (a), son 10 veces mejores que tomar el logaritmo de la frecuencia (l) o simplemente la frecuencia de un término (n).

No deja de sorprender que no considerar la frecuencia de los términos (b) pueda ser más apropiado que considerarla, en dos (n, l) de las otras tres variantes.

5.2.4.4.4. Componentes df y normalización en documentos

Las evaluaciones concernientes a los restantes componentes del esquema de relevancia de documentos, son las siguientes:

—

Consulta	ddfn	ddft	dnn	dnc
backdoor	-0,02965682	-0,02906383	-0,02453011	-0,03419054
sparring	0,00025322	0,00055249	-0,02661142	0,02741713
discerning	-0,01650246	-0,01579639	-0,02119869	-0,01110016
summertime	0,03034616	0,02941176	0,00623886	0,05351906
startling	-0,01721436	-0,01662498	-0,01076706	-0,02307228
colocation	0,05208844	0,05819247	0,05518128	0,05509962
battleground	0,03754326	0,03708032	0,01690851	0,05771507
excavator	0,06537861	0,06525101	0,05390921	0,07672041
whomever	-0,00448052	-0,00486201	0,0288474	-0,03818994

proliferate	0,03951415	0,03957693	-0,00065909	0,07975017
seagram	-0,01829766	-0,0186429	-0,00624568	-0,03069487
frustrate	0,03371589	0,03317818	0,03785791	0,02903616
relaunch	-0,04759122	-0,04847156	-0,01100419	-0,08505859
coreldraw	-0,01641965	-0,01782912	-0,01929672	-0,01495205
beeper	0,0326504	0,03369398	0,02340393	0,04294045
invoicing	0,04457088	0,04569764	0,01779026	0,07247826
komputer	-0,00293019	-0,00476461	0,01310065	-0,02079545
mcclain	0,0018616	0,00141596	0,01332586	-0,0100483
imaginable	-0,01734961	-0,01822145	0,0128451	-0,04841616
unanticipated	0,00074401	0,00092312	-0,01275834	0,01442546
hypocrisy	0,03298417	0,0323712	0,01722533	0,04813004
iplanet	0,01043712	0,01216187	0,02023342	0,00236557
buxton	0,0337777	0,03352999	0,01136432	0,05594337
heros	0,01317526	0,01233118	0,03166053	-0,00615408
stroud	0,01700968	0,01584402	0,0017225	0,03113121
impasse	0,04701235	0,04755435	0,00830067	0,08626604
quaint	-0,03297446	-0,03285357	-0,01169075	-0,05413727
dutchess	0,00616239	0,00615497	0,0093401	0,00297725
optimizer	-0,10460488	-0,10578731	-0,11294066	-0,09745153
outperform	-0,03134686	-0,0310827	-0,02747253	-0,03495703
Promedio	**0,00532855**	**0,00536403**	**0,00313602**	**0,00755657**

–t

Consulta	**ddfn**	**ddft**	**dnn**	**dnc**
backdoor	-0,04345313	-0,04299526	-0,04041314	-0,04603525
sparring	-0,01540823	-0,01460252	-0,04424494	0,01423419
discerning	-0,00792282	-0,00692118	-0,00732348	-0,00752053
summertime	0,03586625	0,03542062	0,01325398	0,05803289
startling	-0,01337617	-0,01298085	-0,0085389	-0,01781812
colocation	0,04430018	0,04663768	0,02966275	0,06127511
battleground	0,04753051	0,04678678	0,03547143	0,05884585
excavator	0,0487825	0,0480544	0,02765268	0,06918423
whomever	-0,00309253	-0,00309253	0,03082792	-0,03701299
proliferate	0,04824713	0,04825497	0,00797188	0,08853022
seagram	-0,01726194	-0,01773272	-0,00696755	-0,02802712
frustrate	0,0408153	0,04126899	0,04419277	0,03789152
relaunch	-0,05127952	-0,05152237	-0,02454314	-0,07825876
coreldraw	-0,00178001	-0,00227405	-0,00335658	-0,00069747
beeper	0,01270718	0,01302947	-0,00601596	0,03175261
invoicing	0,04263474	0,04319812	0,02234495	0,06348791
komputer	-0,00246753	-0,00357143	-0,00181818	-0,00422078
mcclain	-0,00446352	-0,0044132	0,0029613	-0,01183802
imaginable	-0,02358326	-0,02451322	0,00351642	-0,0516129
unanticipated	0,00655828	0,00695095	-0,00877652	0,02228575
hypocrisy	0,02880975	0,02952359	0,01550279	0,04283054
iplanet	0,01771106	0,02023342	0,02816863	0,00977585
buxton	0,04492434	0,04508197	0,01525251	0,0747538
heros	0,00081338	-0,000376	0,01137201	-0,01093462
stroud	0,01207049	0,01235448	-0,00932526	0,03375022
impasse	0,04607686	0,04606943	0,00651877	0,08562752
quaint	-0,03243401	-0,03148111	-0,01540278	-0,04851234
dutchess	0,0098524	0,01000089	0,00016334	0,01968995
optimizer	-0,099552	-0,10069771	-0,10995887	-0,09029083

outperform	-0,02321076	-0,02228621	-0,01953015	-0,02596682
Promedio	**0,00494716**	**0,00511351**	**-0,00037938**	**0,01044005**

-st

Consulta	**ddfn**	**ddft**	**dnn**	**dnc**
backdoor	-0,03980514	-0,03988771	-0,03903201	-0,04066084
sparring	-0,00761971	-0,00785758	-0,04085328	0,025376
discerning	-0,00103448	-0,00111658	-0,00522167	0,00307061
summertime	0,02906676	0,02862113	0,00733138	0,05035651
startling	-0,00945892	-0,00955954	-0,00698637	-0,01203209
colocation	0,04726033	0,0491487	0,03158174	0,06482729
battleground	0,03364246	0,03272418	0,02261551	0,04375114
excavator	0,04529214	0,04444394	0,02754759	0,06218849
whomever	-0,00083604	-0,00151786	0,03050325	-0,03285714
proliferate	0,05067949	0,05061672	0,01018455	0,09111167
seagram	-0,02655201	-0,02558691	-0,01235013	-0,03978878
frustrate	0,04399953	0,04377268	0,0473854	0,04038681
relaunch	-0,04825906	-0,04884342	-0,02076377	-0,07633872
coreldraw	-0,02140366	-0,02162889	-0,01910782	-0,02392473
beeper	0,02212247	0,02144721	-0,00397483	0,04754451
invoicing	0,04569764	0,04676093	0,03254936	0,05990922
komputer	-0,00137175	-0,00175325	0,00022727	-0,00335227
mcclain	0,00280317	0,00260192	0,01009143	-0,00468633
imaginable	-0,02218105	-0,02367771	0,00928509	-0,05514385
unanticipated	0,00495316	0,00476715	-0,01077432	0,02049463
hypocrisy	0,03607232	0,03614991	0,02351024	0,04871198
iplanet	0,01497055	0,01798375	0,02396924	0,00898506
buxton	0,03833393	0,0376884	0,0067105	0,06931184
heros	0,00302333	0,00260129	0,01520872	-0,0095841
stroud	0,01162687	0,0111814	-0,01155263	0,03436089
impasse	0,04442118	0,04437663	0,00310347	0,08569435
quaint	-0,03501536	-0,03470247	-0,01533166	-0,05438616
dutchess	0,01294844	0,01227281	-0,00105429	0,02627554
optimizer	-0,09763514	-0,09894976	-0,10496475	-0,09162015
outperform	-0,02176669	-0,020939	-0,01680051	-0,02590518
Promedio	**0,00513249**	**0,00503794**	**-0,00023211**	**0,01040254**

s-t

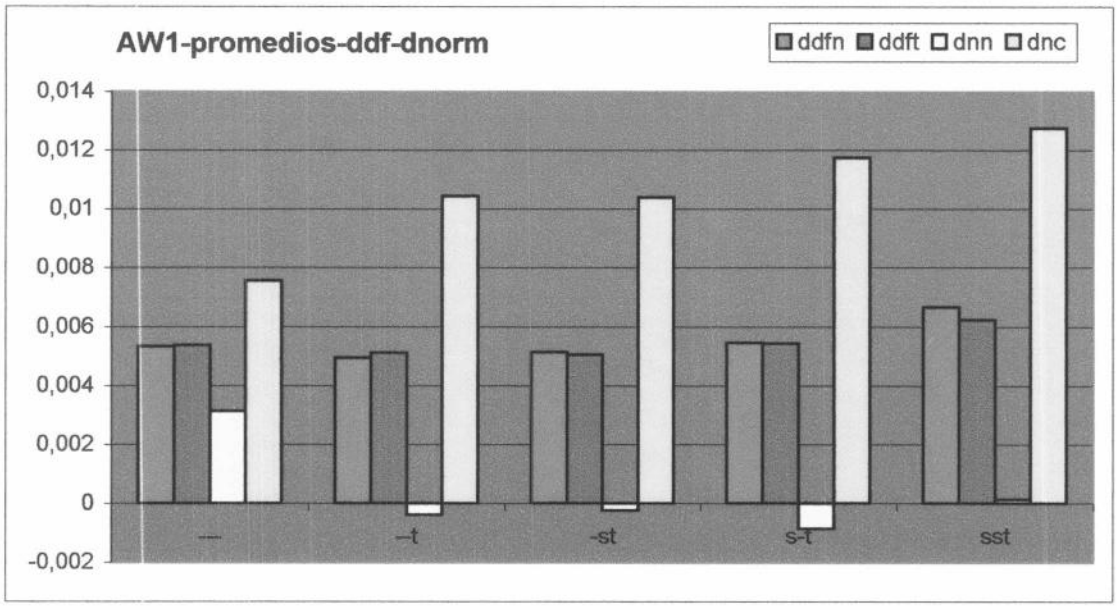
Consulta	**ddfn**	**ddft**	**dnn**	**dnc**
backdoor	-0,04259743	-0,04303279	-0,04111872	-0,0445115
sparring	-0,0110574	-0,01055862	-0,037876	0,01625998
discerning	-0,01426108	-0,01336617	-0,01279146	-0,0148358
summertime	0,04226324	0,04209792	0,01723593	0,06712524
startling	-0,01226209	-0,01236272	-0,00244379	-0,02218101
colocation	0,03937	0,03932917	0,0150049	0,06369427
battleground	0,05422409	0,05279734	0,04081416	0,06620727
excavator	0,02771272	0,02760013	-0,00025521	0,05556807
whomever	-0,00275162	-0,00293831	0,03094156	-0,03663149
proliferate	0,05288431	0,05310401	0,01289938	0,09308895
seagram	-0,01621838	-0,01741887	-0,00929006	-0,02434718
frustrate	0,02833882	0,02779271	0,02439844	0,0317331
relaunch	-0,04186904	-0,04237751	-0,01451035	-0,0697362
coreldraw	-0,00616827	-0,00677129	-0,00935774	-0,00358181
beeper	-0,00106661	-0,00059853	-0,01414979	0,01248465

invoicing	0,08573764	0,08583286	0,06181362	0,10975687
komputer	-0,01429383	-0,01564935	-0,0187013	-0,01124188
mcclain	-0,00372319	-0,00361538	0,00309068	-0,01042925
imaginable	-0,01649956	-0,01744406	0,01059285	-0,04453647
unanticipated	0,00749518	0,00774318	-0,00801874	0,0232571
hypocrisy	0,02588454	0,02670701	0,01340782	0,03918374
iplanet	0,01667485	0,01995392	0,02769143	0,00893734
buxton	0,04685342	0,04677085	0,01532757	0,0782967
heros	-0,02302793	-0,02387201	-0,02398711	-0,02291283
stroud	0,01504217	0,01494565	-0,00726123	0,03724905
impasse	0,04631445	0,04713857	0,00705334	0,08639968
quaint	-0,03406958	-0,03329446	-0,01490499	-0,05245904
dutchess	0,0067712	0,00666726	-0,00267284	0,01611131
optimizer	-0,07101939	-0,07238543	-0,07104877	-0,07235605
outperform	-0,0211063	-0,01974148	-0,01782192	-0,02302585
Promedio	**0,0054525**	**0,00543512**	**-0,00086461**	**0,01175223**

sst

Consulta	**ddfn**	**ddft**	**dnn**	**dnc**
backdoor	-0,03973008	-0,0391446	-0,03949739	-0,03937729
sparring	-0,00258594	-0,00324586	-0,03442296	0,02859116
discerning	-0,01014163	-0,00967159	-0,01463054	-0,00518268
summertime	0,03504686	0,03489592	0,01185958	0,0580832
startling	-0,00902766	-0,00940141	-0,00092002	-0,01750906
colocation	0,04291197	0,0412992	0,01539278	0,06881839
battleground	0,04415336	0,04278732	0,031601	0,05533969
excavator	0,02684952	0,02654927	0,00397826	0,04942052
whomever	0,00206169	0,00063312	0,03422078	-0,03152597
proliferate	0,05517544	0,05564622	0,01531605	0,09550562
seagram	-0,02478658	-0,02521028	-0,01379386	-0,036203
frustrate	0,03028801	0,0287337	0,02742304	0,03159867
relaunch	-0,03840083	-0,03920527	-0,01029081	-0,06731528
coreldraw	-0,0221084	-0,02170154	-0,02090962	-0,02290032
beeper	0,00589319	0,00460405	-0,01586863	0,02636587
invoicing	0,09483114	0,09478353	0,0766362	0,11297848
komputer	-0,01409091	-0,01456981	-0,01774351	-0,01091721
mcclain	0,0083089	0,00743919	0,01381462	0,00193347
imaginable	-0,01579483	-0,01666667	0,01331008	-0,04577158
unanticipated	0,00718517	0,00593828	-0,01107743	0,02420088
hypocrisy	0,03299969	0,03319367	0,02122905	0,04496431
iplanet	0,01715887	0,02039703	0,02683246	0,01072344
buxton	0,04266499	0,04185432	0,01008827	0,07443103
heros	-0,02145488	-0,02252916	-0,02173112	-0,02225292
stroud	0,0141438	0,01374287	-0,01032015	0,03820682
impasse	0,04569821	0,04565366	0,00533084	0,08602103
quaint	-0,03675759	-0,03609626	-0,0145921	-0,05826175
dutchess	0,01037212	0,00872387	-0,00313317	0,02222915
optimizer	-0,06283049	-0,06460047	-0,05891598	-0,06851498
outperform	-0,01863201	-0,01807728	-0,01537405	-0,02133524
Promedio	**0,0066467**	**0,00622517**	**0,00012706**	**0,01274482**

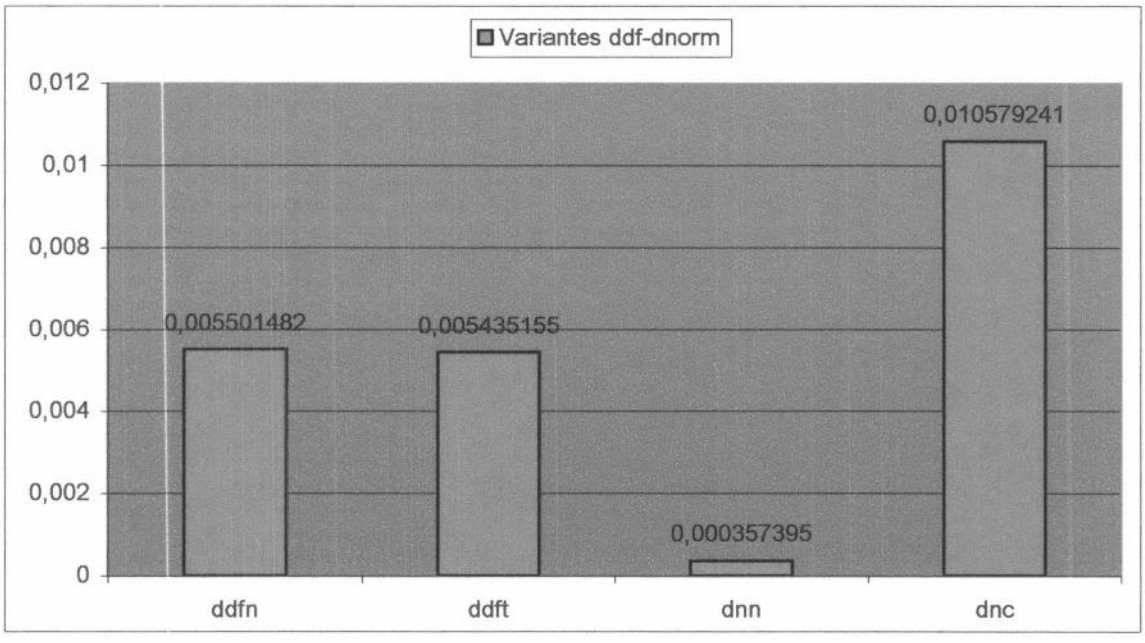
Gráficamente:



Consolidando:

Index id	ddfn	ddft	dnn	dnc
---	0,00532855	0,00536403	0,00313602	0,00755657
--t	0,00494716	0,00511351	-0,00037938	0,01044005
-st	0,00513249	0,00503794	-0,00023211	0,01040254
s-t	0,0054525	0,00543512	-0,00086461	0,01175223
sst	0,0066467	0,00622517	0,00012706	0,01274482
Promedio	**0,00550148**	**0,00543515**	**0,0003574**	**0,01057924**

Gráficamente:



Componente df: aplicar la variante {t} empeora la evaluación respecto de no modificar el peso de los términos con este componente (n). La diferencia es casi imperceptible, pues implica una merma de menos de 0.7 pares por *ranking*.

Componente normalización: aplicar la normalización de documentos en el cálculo de los pesos de los términos en documentos, es 35 veces mejor que no aplicarla. Esta es una diferencia muy pronunciada, y es un gran indicio del uso de alguna forma de normalización en AlltheWeb. La diferencia de más de 0.01 en las evaluaciones, nos habla de más de 100 pares de páginas mejoradas en cada *ranking*.

Culmina así nuestro análisis de AlltheWeb.

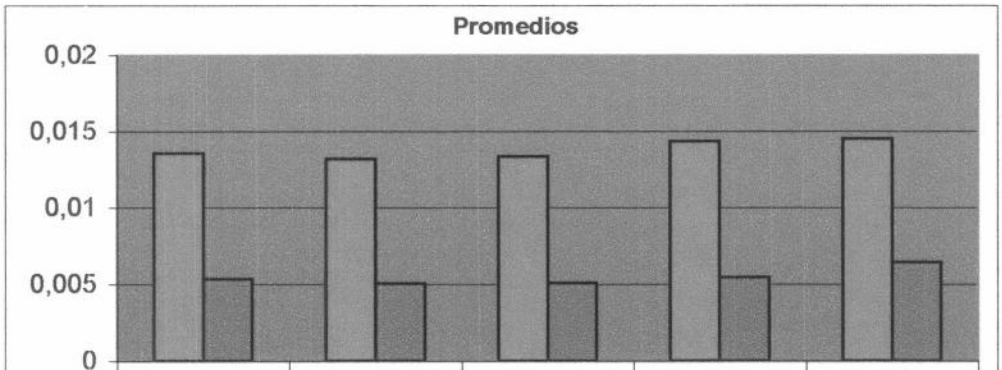
5.2.4.5. Comparación de las evaluaciones en AltaVista y AlltheWeb

Compararemos en esta sección los resultados observados para los dos buscadores en estudio.

Recordemos que las mismas 30 consultas fueron analizadas en los dos MBs, y que en AV la cantidad de resultados por consulta es de 70000 y en AW es de 120000.

Por la cantidad de resultados, y de acuerdo con lo discutido en la sección acerca de la elección de las consultas, es razonable suponer que las evaluaciones en AW sean menores en promedio a las de AV.

Veamos primero los promedios generales en las variantes de indexación.



	---	--t	-st	s-t	sst
AV	0,01356054	0,013191999	0,01334301	0,014322182	0,014491166
AW	0,005346294	0,005030338	0,005085214	0,00544381	0,006435936

Constatamos la suposición previa acerca de la inferioridad de las evaluaciones de AW respecto de las de AV. Éstas últimas son mayores al doble de las primeras.

Otra de las explicaciones posibles de esta diferencia está en suponer que los esquemas de relevancia de modelo vectorial que estamos evaluando, se adaptan mejor a los rankings de AV que a los de AW.

En ambos MBs, la aplicación de *stemming* fue positiva, aunque en distinta medida pues en AW, si bien en un caso mejora más de 13 pares, en promedio mejora menos de 9 contra más de 11 de AV.

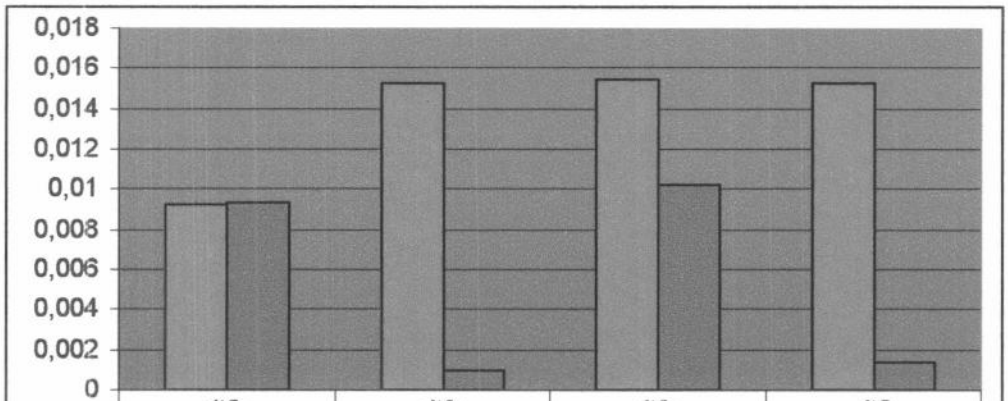
La eliminación de *stopwords*, también benefició a las evaluaciones en ambos MBs, aunque esta vez en donde más pares se reordenaron fue en AW con más de 5 en promedio, contra menos de 2 para AV.

El uso conjunto de stemming y stopwords mejoró casi la misma cantidad de pares en los dos MBs, 12.92 de AV, contra 13.98 de AW.

La utilización de texto de atributos de tags HTML fue perniciosa en ambos casos, con más de 3 pares desmejorados.

Comparar las variantes de componentes de consultas no aporta demasiado, pues ya dijimos que no variaban las evaluaciones, por lo que sería repetir el análisis previo.

Veamos qué sucede con el componente tf de documentos.



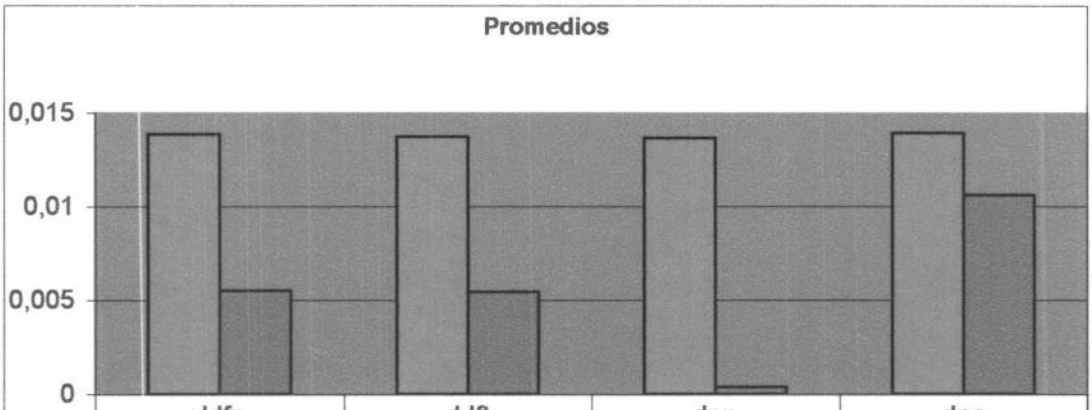
	dtfb	dtfn	dtfa	dtfl
AV	0,009203389	0,015276005	0,015411886	0,015235837
AW	0,009320264	0,000948399	0,010270809	0,001333801

Claramente, donde se observa una mayor diferencia es en las variantes {n} y {l}, donde las evaluaciones de AV son 15 veces mejores que las de AW.

La variante {b} es casi tan apropiada para AV como para AW, y en la variante {a} vemos una diferencia también a favor de AV, pero no tan pronunciada como en {n} y {l}.

Mientras que en AV hay tres variantes con similar desempeño, en AW las mejores evaluaciones se reparten entre sólo 2 de las 4 variantes.

Pasemos a comparar los componente df y normalización de documentos.



	ddfn	ddft	dnn	dnc
AV	0,013850821	0,013712738	0,013654864	0,013908694
AW	0,005501482	0,005435155	0,000357395	0,010579241

En el componente df, las luchas entre {n} y {t} en ambos buscadores es pareja. En cada variante, AV supera al doble de AW, manteniendo la tendencia de los promedios generales que vimos por cada variante de indexación.

En el componente de normalización, la lucha es mucho más pareja en AV que en AW, donde la normalización de pesos de términos en documentos parece ser el rasgo más prominente

de este experimento. Recordemos que esta diferencia implica una mejora de más 100 pares en AW, la mayor de todas las observadas. La diferencia entre {c} y {n} en AV es de sólo 2.5 pares.

Hemos evaluado 30 consultas en 2 motores de búsqueda de la web. Analizamos la influencia en el resultado de las evaluaciones, por parte de diversas variantes de indexación y de relevancia. Luego comparamos los resultados de ambos buscadores para destacar similitudes y diferencias.

Constatamos repetidas veces algunos resultados coherentes con las fórmulas de pesos teóricos y con los demás resultados arrojados por el experimento. De esta manera, se confirmó que el funcionamiento del sistema EVALSE, utilizado para arribar a todos estas evaluaciones, es el esperado.

Las conclusiones a las que se llegó al observar los datos tabulados y graficados, tienen un basamento de cientos de miles de evaluaciones. Cada una de tales evaluaciones, recordemos, corresponde a un *ranking* construido por el sistema evaluador. Para confeccionar cada *ranking* se tuvo que asignar un puntaje a cada una de las 200 páginas por consulta (12000 en total).

Para calcular el puntaje de cada página, se utilizó uno de los 256 esquemas posibles de relevancia, y una de las 5 variantes de indexación.

Cada esquema de evaluación se subdivide a su vez en subesquemas para consultas y documentos, que a su vez se subdividen en componentes, que a su turno implementan diferentes variantes.

Toda esta recopilación sirve para sustentar los resultados de este experimento, que se realizó con consultas reales en dos de los buscadores de la web más populares y con índices de páginas más numerosos de la web.

5.2.5. Limitaciones, posibles extensiones y aplicaciones de EVALSE

El sistema evaluador presentado en esta sección no es un sistema terminado, ni es un sistema cerrado, pero obtuvimos algunos resultados de su aplicación que nos sirvieron para entender mejor a los sistemas evaluados. Sirve como un primer prototipo de lo que se puede conseguir mediante la evaluación de motores de búsqueda, ya sea de la web o de cualquier otra colección de documentos.

En lo que sigue, cuando se mencione una limitación, se deberá interpretar como una oportunidad de extender la funcionalidad del sistema para ampliar su utilidad en la evaluación de los SRI.

Las decisiones de diseño no son definitivas pues existen posibilidades concretas de construir más y mejores variantes con el fin de poder aproximarse mejor a los sistemas evaluados.

Por ejemplo, cada módulo podría variar su funcionamiento de la siguiente manera:

Recolección:

Ya se han puntualizado algunas limitaciones asociadas a considerar consultas de una sola palabra, como las relacionadas con la eliminación de *stopwords*. Realizar el análisis con consultas más sofisticadas puede ayudar a establecer, por ejemplo, si el algoritmo de asignación de relevancia varía de acuerdo a la forma de la consulta.

Puede obviarse este paso por completo si ya se dispone de los documentos de la colección.

Indexación:

Ya hemos visto cómo se pueden probar diversas variantes como *stemming* y *stopwords*. El algoritmo utilizado para lematizar términos, puede ser cambiado y arrojar resultados diferentes. También podemos probar con otras listas posibles de *stopwords*. Aquí utilizamos el idioma inglés, pero estudiar consultas en otros idiomas con algoritmos de lematización, y listas de palabras vacías apropiados es también otro desafío.

También se utilizó texto de atributos de tags HTML. Los tags utilizados pueden variar, así como también puede considerarse a un texto extraído de un tag, con más peso que el texto extraído en otro lugar de la página. Las variantes posibles son innumerables, por ejemplo si se considera que el TITLE puede ser más valioso que el texto de BODY. Cuánto más valioso es otro punto a tener en cuenta.

El énfasis del texto, dado por tags HTML específicos, o la ubicación dentro de un mismo tag, pueden ser factores de influencia en el cálculo del peso de un término.

Si consideramos algoritmos de relevancia que tienen en cuenta la red de hipervínculos entre páginas web, puede ser útil tener en cuenta al texto de anclaje (*anchor text*) que circunda los enlaces de una página a otra. También a este tipo de texto se puede asignar un peso especial.

Los pasos seguidos para obtener las palabras o *tokens* del texto, pueden ser revisados y los signos de puntuación o los separadores posibles, cambiados.

La frecuencia en documentos (df) de los términos se calculó a partir de las páginas de cada consulta. Se podría calcular de otras maneras, para aproximar quizás mejor a las dfs del índice del buscador evaluado. Una posibilidad es hacer consultas por sitio, y extraer del sitio completo a las dfs. Otra variante sería utilizar la información brindada por la interfaz del MB analizado, como por ejemplo AltaVista, que brinda no sólo la cantidad de resultados por consulta realizada, sino la cantidad de documentos en donde aparece cada uno de los términos presentes en la consulta.

Otras transformaciones del texto pueden ser evaluadas, como por ejemplo, se puede citar el caso del algoritmo *soundex*, que dado un término lo transforma o normaliza hacia su representación fonética. Podemos aplicar este proceso y observar cómo influye en las evaluaciones, de la misma manera en que lo hicimos con el *stemming*, por ejemplo.

Relevancia:

Las páginas con puntajes arbitrarios, como pueden ser los posicionamientos pagos, no pueden ser contemplados por el sistema evaluador.

No se consideraron tampoco las cuestiones asociadas a las medidas de defensa ante la PMB, como pueden ser topes a cantidad de repeticiones de un término, o penalizaciones varias.

Tampoco fue tenida en cuenta la estructura de un documento, en cuanto al peso diferenciado que puede tener un término por aparecer en una región determinada (e.g. título) del mismo.

Las variaciones posibles a los esquemas de relevancia son innumerables. Siempre dentro del modelo de espacio vectorial, podemos considerar más componentes, o más variantes (fórmulas) para los mismos componentes. También podemos usar otras funciones de similitud entre consultas y documentos [BUC/96].

Se podría también cambiar de modelo y probar por ejemplo, funciones de asignación de relevancia del modelo probabilístico. Incluso se pueden explorar los componentes de pesos en este u otros modelos.

Evaluación:

La función de distancias puede ser cambiada, para tener en cuenta diferentes aspectos al comparar dos *rankings*. Quizás se preferirá dar énfasis a las primeras posiciones de los *rankings*, por sobre el resto, pensando que allí es donde interesa que se asemejen. O quizás utilizando, como hace SMART, medidas de precision-recall en 11 niveles, o restringirse a 6 niveles.

Reporte:

Podemos estar interesados, además de los promedios, en los máximos y mínimos, o en las medianas. O ver una clasificación de los esquemas de relevancia según su evaluación decreciente. Las posibilidades aquí dependen de lo que se desee analizar con más detalle, y de la posible utilización posterior del sistema evaluador.

Vistas las posibles mejoras y/o extensiones a este sistema, pensemos en las posibles **aplicaciones** de sus resultados, las evaluaciones.

Comprender la manera en que un MB indexa los documentos, o asigna pesos a términos en consultas y documentos, o calcula la similitud entre ellos, es sin duda deseable, y complejo.

Aproximar alguno de estos aspectos nos puede ayudar, por ejemplo, a integrar los resultados de tal MB en un **metabuscador**. Entre las acciones que estos sistemas deben realizar, se encuentran la elección de los buscadores (*database selection*) que pueden dar una respuesta relevante a determinada consulta. Luego deben procesar la consulta (transformada si es necesario) en dichas fuentes, e integrar los *rankings* resultantes (*result merging*).

El proceso de selección de fuentes de información o bases de datos apropiadas podría hacer uso de la información que le brinde el sistema evaluador. Un metabuscador podría tener información acerca de qué tipo de consultas son mejor evaluadas en cada MB, donde el tipo puede depender del tema sobre el que gira la consulta, o quizás referirse a la forma sintáctica que posee.

En [GRA/97] se presenta el protocolo STARTS, que define qué datos necesita brindar una fuente de información (*resource metadata*), para poder integrarla en un metabuscador. En el caso de fuentes no cooperativas, como los MBs de la web, puede inferirse parte de esa información a partir de las evaluaciones que se hagan a dichos MBs.

Otra posible uso de EVALSE, es en una herramienta de **ingeniería de sitios** y posicionamiento de los mismos ante un MB en determinado conjunto de consultas C. Los esquemas de relevancia E que mejor se adapten al MB, darán indicios de cómo modificar o diseñar páginas y sitios para que *rankeen* mejor en C. De hecho, una práctica posible sería verificar la posición de una página en un *ranking* "local" r(c,e), antes de enviarla al MB. La medida de evaluación *ev* en (c, e, ev) da también una idea numérica de la precisión con que se logrará posicionar a la página en c. Esta medida puede interpretarse también como la susceptibilidad de un MB a ser manipulado usando el esquema e en la consulta c.

De hecho, cualquier algoritmo o método destinado a comprender y descubrir criterios de relevancia en buscadores, puede ser utilizado (y abusado) con fines de manipulación de los *rankings*. Es decir, para practicar persuasión de motores de búsqueda (PMB).

Una última posibilidad, es aplicar un sistema evaluador, como el presentado en esta tesis, en una **metaheurística**.

Cada esquema de relevancia de partida *e* puede ser tomado como una solución inicial local, que se puede mejorar iterativamente hasta encontrar un óptimo local. Luego se pueden comparar las evaluaciones de los óptimos locales y devolver el esquema que sea óptimo a nivel global. En este sentido, que un esquema *e1* inicial local sea mejor que otro *e2* también inicial

local, no asegura que el óptimo local *o1* correspondiente a *e1* sea mejor (i.e. describa mejor al MB) que el óptimo local *o2* correspondiente a *e2*.

Los métodos de mejoramiento de un *e* aplicados en la búsqueda local del esquema óptimo local *o*, pueden también variar: agregando componentes al esquema, cambiando componentes (fórmulas), optimizando constantes de componentes [LIU/00], etc.

.Iterar sobre todos los e de la knowledge base E.
.Armar solución inicial local SL a partir de e.
.Optimizar SL para obtener óptimo local OL.
.Fin Iterar
.Devolver mejor OL respecto del MB para C.

Capítulo 6. Conclusiones

A lo largo de todo esta tesis, tanto en lo que quedó plasmado en este informe como en las innumerables lecturas, experimentos, ideas e intentos muchas veces fallidos, siempre se buscó poner luz sobre un tema sumamente importante para la comunidad, como es el de acceder a la inmensa cantidad de información que nos brinda la Web hoy.

Se puntualizó que los únicos sistemas preparados para manejar actualmente, y hacer frente en el futuro, a tal volumen de información sin estructura son los motores de búsqueda de recolección automática.

Sobre esta premisa, se avanzó hacia el punto crucial en el diseño de tales sistemas, como es el algoritmo de asignación de puntajes a páginas, ante las consultas de usuario.

Se realizó un panorama totalizador, desde los sistemas de recuperación clásicos, hasta las últimas innovaciones que hacen uso del potencial de hiperenlaces, y la estructura HTML.

También se dedicó un capítulo completo a un tema que aparece solamente en la Web, como la persuasión de motores de búsqueda, y al que no se le dio hasta aquí la importancia que merece por su innegable y determinante influencia en el cálculo de los puntajes.

En el objetivo apuntado de comprender y descubrir el funcionamiento de estos recopiladores y servidores de información global, se pensaron y diseñaron técnicas con diferentes fundamentos. A partir de tales métodos, se ejecutaron experimentos y se eligió exponer en el presente trabajo los referidos a la validación de hipótesis y al sistema evaluador.

Reiteraremos en este capítulo las principales conclusiones a las que se arribó como consecuencia de tales experimentos. Además, extenderemos dichas conclusiones.

En cuanto a la validación de **hipótesis**, se debe apuntar que, aun con las trabas impuestas por el proceso de *submission* de páginas, se pueden hacer cosas para mejorar los resultados.

En primer lugar, se podrían realizar lotes para más consultas, algo que se intentó sin éxito. De esta manera, se podrían sacar conclusiones más abarcadoras.

Dichas consultas deben tener una popularidad semejante, y el envío de las páginas debe hacerse en un período de tiempo lo más corto posible, aunque evitando el rechazo justamente por abusar del servicio de envío manual de páginas. Asimismo, debe utilizarse la mayor cantidad de dominios posible, para evitar el rechazo por exceso de páginas por dominio.

También se podría aumentar la cantidad de páginas por lote para validar más fuertemente las hipótesis. Por ejemplo, se podrían tener 7 páginas con 7 tamaños de font diferentes.

Otro de los intentos fallidos, pero útiles sin duda, implica el envío de páginas a otros buscadores, como se intentó con FAST, obteniendo la indexación de sólo 3 páginas sobre 2 lotes de 28 páginas cada uno.

Yendo a las conclusiones del experimento sobre AltaVista, vemos que:

- El tag TITLE es muy importante, más que la frecuencia de un término en cualquier otro lugar de la página.
- El sublote con 2 o más apariciones del término-consulta, supera generalmente a las páginas que tienen una sola aparición de tal término.
- Las mismas páginas en una misma consulta, cambiaron el orden de aparición, por lo que, o algunas páginas tienen puntajes muy similares y se las ordenó

arbitrariamente, o bien se aplicó otro esquema de relevancia, temporaria o definitivamente.

- La popularidad de un hipervínculo, la descripción de un hipervínculo, el uso de sinónimos, la descripción de una imagen, el nombre de una imagen y el mayor tamaño de la fuente pueden ser considerados factores que tienden a mejorar la posición de una página.
- El uso de marcos (*frames*), la acumulación excesiva de apariciones de un término, ya sea para mejorar su frecuencia o porcentaje, no parecen ayudar a mejorar el puntaje de una página.

En lo referente al sistema **EVALSE**, debemos aclarar nuevamente su principal cometido.

Es el de descubrir tendencias, analizar variantes o aspectos de diversos métodos de indexación o relevancia. No apunta a aproximar lo más posible en términos absolutos a los *rankings* de un buscador. Si lo logra, tanto mejor, pero dada la cantidad de obstáculos que impiden que ello se logre, alcanza con comparar las evaluaciones entre los esquemas locales.

No se pretende modelar con exactitud a un MB, porque hay factores arbitrarios que no podrían conocerse (posicionamiento pago), y otros muy difíciles de conocer (penalizaciones), o de descubrir (algoritmos que usan todo el grafo de la web, como PageRank; o algunas medidas estadísticas a partir de sus índices desconocidos).

Se puntualizó oportunamente la importancia de ampliar los experimentos con consultas de más de una palabra.

Los resultados para consultas en inglés, nos indican que AltaVista:

- Utiliza algún algoritmo de *stemming* relacionado con el de Porter.
- No da relevancia al texto presente en el interior de tags considerados en el experimento.
- Tiene en cuenta la información de la cantidad de apariciones de un término.
- Las df obtenidas de las consultas no son apropiadas para explicar el componente de frecuencia en documentos, de la manera en que éste se aplique.
- No parece aplicar ningún tipo de normalización de documentos.

Para FAST, podemos aseverar que:

- Utiliza algún algoritmo de *stemming* relacionado con el de Porter.
- No da relevancia al texto presente en el interior de tags considerados en el experimento.
- No es fácilmente entendible la manera en que considera la frecuencia de un término.
- Las df obtenidas de las consultas no son apropiadas para explicar el componente de frecuencia en documentos, de la manera en que éste se aplique.
- Es muy importante la aplicación de la normalización en documentos (recordemos los más de 100 pares mejorados en promedio).

Capítulo 7. Referencias web y bibliográficas

7.1. Bibliografía

[BAE/99] http://www.dcc.ufmg.br/irbook.
R.Baeza-Yates, B.Ribeiro-Neto. Modern Information Retrieval. Addison-Wesley. ACM Press. 1999.

[BRI/98] http://www7.scu.edu.au/programme/fullpapers/1921/short321.htm
Sergey Brin y Lawrence Page, The Anatomy of a Large-Scale Hypertextual Web Search Engine, 1998.
Explica el funcionamiento interno de un buscador de gran escala: Google. Es una de las pocas descripciones de este tipo que se hizo pública. Es importante el peso que se le da al grafo de hipervínculos en el cálculo de la relevancia de las páginas: la medida PageRank.

[BUC/96] C. Buckley et al. Pivoted document length normalization. *ACM SIGIR Conference*, Zurich. 1996.

[CHA/98] http://www7.scu.edu.au/programme/fullpapers/1898/com1898.html
S. Chakrabarti, B. Dom, D. Gibson, J. Kleinberg, P. Raghavan, S. Rajagopalan. Automatic Resource Compilation by Analyzing Hyperlink Structure and Associated Text. Proc. 7th International World Wide Web Conference, 1998.

[CUT/97]

http://www.usenix.org/publications/library/proceedings/usits97/full_papers/cutler/cutler_html/cutler.html. Michal Cutler, Yungming Shih, Weiyi Meng. Using the Structure of HTML Documents to Improve Retrieval. Proceedings of the Symposium on Internet Technologies and Systems, Monterey, California. December 8-11, 1997.

[GAR/98]
N. Shivakumar; H. García Molina. Finding near-replicas of documents on the Web. Workshop on Web Databases, Valencia, España, Marzo 1998.

[GOO/54] L. A.Goodman, W.H. Kruskal. Measures of association for cross classifications. Journal of American Statistics Association, 49:732-764. 1954.

[GRA/97] L. Gravano et al. STARTS: Stanford proposal for Internet metasearching. *Proceedings of the 1997 ACM SIGMOD Conference*, Mayo 1997.

[HEN/98]
Krshua Bharat, Monika Henzinger. Improved algorithms for topic distillation in a hyperlinked environment. En 21nd SIGIR Conference on Research and Developmente in Information Retrieval. 1998.

[KLE/98]
Kleinberg, Jon M. Authoritative sources in a hyperlinked environment. ACM-SIAM Symposium on Discrete Algorithms.
Se proponen y ofrecen resultados acerca de algoritmos que utilizan la estructura de red de los hiperenlaces para obtener información útil en el descubrimiento de documentos autoridades y concentradores (authorities and hubs) respecto del tema de interés en una búsqueda.

[LIU/00] Liu, Meng, Yu. Discovery of similarity computations of Search Engines.

[MAR/96]
Marchiori, Massimo. Security of World Wide Web Search Engines. Department of Pure and Applied Mathematics, University of Padova. 1996.

[MAR/97] http://www.scope.gmd.de/info/www6/technical/paper222/paper222.html
Marchiori, Massimo. The quest for correct information on the Web: Hyper Search Engines. Department of Pure and Applied Mathematics, University of Padova. 1997.

[PAG/98]
S. Brin y L. Page. The PageRank citation ranking: bringing order to the web. Technical Report. 1998.

[POR/80] Porter, M.F. An Algorithm For Suffix Stripping. Program 14 (3), July 1980, pp. 130-137.

[PRI/98] http://mapping.cs.monash.edu.au/~pringle/WWW7/96.html
Pringle, G.; Allison, L.; Dowe, D.L. What is a tall poppy among Web pages? Computer Networks And Isdn Systems , Vol: 30, Issue: 1-7, April 1998 , pp. 369-377.

[RAG/97] V.Gudivada, V.Raghavan, W. Grosky and R.Kasanagottu. Information retrieval on the World Wide Web. IEEE Internet Computing, Oct-Nov:58-68, 1997.

[SAL/74] Gregory Salton, A.Wong, C.S.Yang. A vector space model for automatic indexing. Department of CS, Cornell University. TR74-218. 1974.

[SAL/83] G. Salton; M. J. McGill. Introduction to modern information retrieval. McGraw-Hill. 1983.

7.2. Webliografía

7.2.1. Páginas Web con citas referentes a ranking

[w1] Infoseek, a partir del año pasado, habría dado más peso a páginas de sites que están listadas en su directorio confeccionado por humanos y a la popularidad de las mismas.
Gradualmente se irá quitando la importancia relativa de la aparición de las palabras en las páginas para la confección de los rankings.

[w2] En su interior, los principales MBs usan el método de ubicación/frecuencia para determinar la relevancia.
Direct Hit usa la retroalimentación del usuario para mejorar la relevancia.
Clever (MB de IBM basado en HITS) considera que las páginas apuntadas desde otras páginas "importantes" deben tener más peso.
Google utiliza además el texto en los hipervínculos.
Hay un creciente énfasis en considerar aspectos no tradicionales como ser los enlaces o la retroalimentación del usuario para dar sentido a la web, en oposición a considerar sólo las palabras en una página. Esto es esencial en un contexto en el que hay páginas con texto no confiable y páginas imposibles de indexar.

[w3] No se puede comprar una mejor posición en los rankings de los principales MBs. Ninguno lo permitirá, luego de que la experiencia de Open Text en 1996 produjera una publicidad adversa.

[w4] Hotbot e Infoseek privilegian las páginas con palabras clave en sus meta tags. Pero Lycos ni siquiera las lee.

[w5] Use Títulos con las primeras letras del alfabeto. Todos los MBs que usé muestran los resultados con el mismo puntaje en orden alfabético. La mayoría de los MBs dan más peso a una página si las palabras clave aparecen cerca del comienzo de la misma.

[w6] El tag meta keywords sirve para Infoseek y Altavista, mientras Excite y muchos otros enfatizan el contenido y los títulos.
A veces son necesarias palabras clave para ayudar a los MBs a clarificar el contexto.

[w7] La WWW puede parecer un medio visual, pero es indexada textualmente.

7.2.2. Páginas Web con citas referentes a web spamming

[w8] Un número pequeño de sites envían regularmente una gran cantidad de páginas a Altavista con el fin de aparecer arriba en nuestras páginas de resultados. La técnica usual es enviar páginas con numerosas palabras clave, o no relacionadas con el contenido real de las mismas. Algunas personas envían páginas a nuestro spider que difieren de lo que se verá en un navegador. No permitiremos envíos de quienes hagan spam en nuestro índice. En casos extremos, excluiremos todas sus páginas del índice.

[w9] Cada documento obtiene una graduación según cuántos términos de búsqueda contenga, dónde estén en el documento, y cuán cerca estén unos de otros. Repetir una palabra muchas veces en una página web, lo que se conoce como "spamming", tiene un efecto negativo en el ranking del sitio.

[w10] (Al respecto del servicio Direct Hit usado por algunos MBs para rankear según popularidad). ¿Los dueños de los sitios pueden simplemente hacer click para llevar sus páginas a la cima?

[w11] Sección SPAM: tag meta refresh, texto invisible, texto pequeño.

[w12] Los MBs enfrentan mucho spam, y el peor es el oportunista. Esto ocurre cuando alguien decide tomar ventaja de un tema candente y crea una página destinada a atraer a los que buscan información al

respecto, aunque el contenido ofrecido es generalmente muy malo en términos de calidad.

[w13] Si encuentras que no hay meta keywords en un sitio, hay chances de que esté usando una técnica para ocultar keywords, como por ejemplo scripts de servidor que envían páginas diferentes para cada MB que las rastree.
Estos Stealth scripts fueron designados ya para proteger las keywords de nosotros (la posible competencia o empresas de web design & promotion), o para ofrecer a los MBs contenido cargado de keywords con el fin de provocar una mejor posición en el ranking.
...el MB revisitará y recargará tarde o temprano los sitios registrados sin que el dueño lo sepa, con el fin de expulsar páginas borradas o para evitar el spamming, así que estos sitios desaparecerán de una manera u otra.

[w14] Y si se usa una sola palabra clave, y luego se la repite en otra cadena, también puede ser penalizado.
Infoseek asevera "Para mejores resultados, se deben evitar las siguientes técnicas de publicación por web:

Uso excesivo o repetición de keywords
Uso de keywords no relacionadas con el contenido del site
Uso de meta refresh rápido
Uso de texto del mismo color del fondo
Duplicación de páginas con diferentes URLs
Uso de páginas diferentes que conducen a la misma URL"

...al menos un MB ha añadido lógica de indexado que considera spamming a la exacta duplicación de porciones sustanciales de la lista de keywords, y borrará las páginas a la semana de su agregado. Para sortear esto, se debe convertir a la lista de keywords en oraciones, aun cuando no tengan mucho sentido, de tal forma de evitar pattern matching con las keywords.

[w15] http://www.northernwebs.com/.../alta_vista.htm
Altavista considera truculentas las siguientes técnicas:
Envíos cíclicos de páginas. Meta tags no relacionados con el contenido. Múltiples páginas de entrada. Engaño al spider (mostrar en el navegador un archivo distinto al que detecta el spider).
Cada documento se clasifica según cuántos términos de búsqueda contiene, dónde están las palabras, y cuán cerca están entre sí.
Altavista usa también la popularidad del sitio para aumentar el ranking de un sitio web.

Ranking:
[w1]http://www.searchenginewatch.com/sereport/9903-relevancy.html
[w2]http://www.searchenginewatch.com/sereport/9808-clicks.html
[w3]http://www.searchenginewatch.com/sereport/9708-retailers.html
[w4]http://www.searchenginewatch.com/webmasters/rank.html
[w5]http://webreference.com/content/search/how.html
[w6]http://searchenginewatch.internet.com/webmasters/tips.html
[w7]http://www.northernwebs.com/set/index.html
[w15]http://www.northernwebs.com/.../alta_vista.htm

Spamming:
[w8]http://www.altavista.com/cgi-bin/query?pg=addurl
[w9]http://www.altavista.com/av/content/ques_howto.htm
[w10]http://www.searchenginewatch.com/sereport/9808-clicks.html
[w11]http://www.searchenginewatch.com/webmasters/features.html
[w12]http://www.searchenginewatch.com/reports/spam.html
[w13]http://www.bruceclay.com/web_rank.htm
[w14]http://searchenginewatch.internet.com/webmasters/tips.html

7.2.3. Otros sitios web

[SMART] ftp://anonymous@ftp.cs.cornell.edu/~ftp/pub/smart/smart.11.0.tar.Z

[MG] http://www.kbs.citri.edu.au/mg

[AV-submit] http://www.altavista.com/sites/search/addurl
[AV-trunc] http://doc.altavista.com/adv_search/ast_haw_pageindex.html
"...Check at off hours, because during an unexpected traffic crunch, your search might be truncated, and the number might not be accurate...".
[SEW] http://www.searchenginewatch.com. Sitio de información acerca de los motores de búsqueda de la web.
[SESH] http://www.searchengineshowdown.com .
[Archive] www.archive.org.
[Deepweb] http://www.completeplanet.com/Tutorials/DeepWeb/index.asp.
[Cyveil] http://www.cyveillance.com/web/us/newsroom/releases/2000/2000-07-10.htm.
[Netsizer] http://www.netsizer.com .
[ISC] http://www.isc.org .
[OCLC] http://www.oclc.org .
[StatMarket] http://www.statmarket.com .
[Nielsen] http://www.nielsen-netratings.com .
[MedMet] http://www.mediametrix.com .
[GVU] http://www.cc.gatech.edu/gvu/user_surveys/ .
[NPD] http://www.npd.com .

Capítulo 8. Glosario

WWW o Web	*World Wide Web*. La telaraña de documentos que componen Internet.
HTML	*Hypertext Markup Language*. Lenguaje para crear páginas web.
MB	Motor de Búsqueda (de la web si no se aclara).
Ranking	Secuencia de resultados devueltos por un MB ante una consulta.
IR	*Information retrieval*. Ver RI.
RI	Recuperación de Información.
SRI	Sistema de Recuperación de Información.
Tag	Elemento del HTML para delimitar texto y definir funcionalidad.
Hiperenlace	Enlace o *link*: ver hipervínculo.
Hipervínculo	Conexión o apuntador desde una página HMTL hacia otra.
Spamdexing	Repetición excesiva de un término en una página HTML, con el objetivo de aumentar su relevancia para las consultas que contengan a dicho término.
PMB	Persuasión de Motores de Búsqueda. El proceso mediante el cual se intenta manipular un MB para aumentar la relevancia de determinadas páginas.
SEP	*Search Engine Persuasion*. Ver PMB.
Recall	Medida usada en la evaluación de un SRI. Es la proporción de documentos relevantes que fueron recuperados.
Precisión	Medida usada en la evaluación de un SRI. Es la proporción de documentos recuperados que son relevantes.
Recuperación	Ver recall.
Index term	Ver Término índice.
Término índice	Término utilizado para representar a un documento en el índice de un SRI.
Stopword	Palabra demasiado frecuente en una colección, poco representativa de un documento en particular.
Stem	Raíz gramatical de un término luego de quitarle prefijos o sufijos.
Thesaurus	Estructura jerárquica de conceptos, destilados a partir de términos.
VSM	*Vector-space model*. Modelo utilizado por un SRI, que representa documentos y consultas con vectores.
TREC	*Text Retrieval Conferences*. Conferencias sobre Recuperación de Texto.
Tf	*Term frequency*. Frecuencia relativa de un término en un documento.
Df	*Document frequency*. Frecuencia de un término en una colección.
Idf	*Inverse document frequency*. Frecuencia inversa en documentos de una colección.
URL	*Uniform Resource Location*. Dirección universal de acceso a un recurso web.
Crawling	Proceso de recolección de páginas para indexarlas en un MB.
Indexación	Procesamiento de documentos para obtener una representación de los mismos en la base de datos de un MB.