SISTEMAS DE GESTIÓN DOCUMENTAL

RANKING

INTRODUCCIÓN

Mediante este trabajo pretendemos realizar un estudio general de una de las formas de monitorización de resultados de búsqueda en sistemas de gestión documental, el ranking.

El ranking se define, generalmente, como un conjunto de elementos con un orden; a los cuales se les otorga un valor de 0 a 1, mostrándose en primer lugar los más próximos al 1.

Dentro de los sistemas de recuperación de información, el ranking consiste en una lista de documentos ordenados en función de su relevancia respecto de una consulta dada. De este modo, el usuario realiza una consulta empleando una serie de términos que describen su necesidad de información; y recupera una lista de documentos ordenados según el grado de correspondencia entre el contenido de los mismos y los términos de dicha consulta.

CÁLCULO DEL RANKING

Para acercarnos al cálculo del ranking asumiremos que nos encontramos ante un sistema que emplea *n* términos únicos. De este modo, tanto las consultas realizadas por los usuarios, como los documentos contenidos en la base de datos; pueden representarse mediante un vector *(t1, t2, t3, . . . , tn)*, donde *ti* adquiere un valor 1 cuando el término *i* está presente, o un valor 0 cuando no lo está.

Así, el usuario realiza una consulta en lenguaje natural, de la que el sistema extraerá los términos válidos para la búsqueda, generando un vector representativo de la necesidad de información de este usuario. Del mismo modo generará un vector por cada documento contenido en la base de datos. De esta forma el sistema aplica una técnica comparativa entre el vector consulta y todos y cada uno de los vectores asociados a los documentos; mediante esta comparación el sistema asigna un valor a cada ítem, en función del cual se ordenarán los documentos dentro del ranking.

En el siguiente cuadro se puede observar un ejemplo de cómo funcionaría lo anteriormente descrito en una base de datos simple.

En la sección ***A*** todos los términos del set de datos

B consulta en lenguaje natural y traducción en vector conceptual

C representación conceptual de tres documentos

D se realiza el producto de los vectores. El resultado sirve para ordenar los documentos en el ranking

MODELOS

Los modelos de ranking se engloban en dos tipos: los que comparan la consulta con documentos individuales (uno por uno), y los que lo hacen con conjuntos de documentos relacionados (clusters). Dentro del primer tipo se hallan los más utilizados: el modelo del espacio vectorial y el modelo probabilístico.

Modelo del Espacio Vectorial

Este modelo sitúa los vectores de los documentos y de la consulta dentro de un espacio vectorial de *n* dimensiones; siendo *n* el número de términos únicos en la base de datos. De esta forma el sistema puede calcular el coseno del ángulo que forma el vector consulta con el resto, obteniendo un valor que determinará la similitud entre la consulta y cada documento, y permitirá al sistema ordenar los registros en función de esa similitud.

$$(d_j, q_k) = \frac{\sum_{i=1}^{n} (td_{ij} \times tq_{ik})}{\sqrt{\sum_{i=1}^{n} td_{ij}^2 \times \sum_{i=1}^{n} tq_{ik}^2}}$$

donde

tdij = el término *i* en el vector para el documento *j*

tqik = el término *i* en el vector para la consulta *k*

n = número de términos únicos en la base de datos

Este modelo ha sido utilizado como base para varios experimentos de recuperación con ranking, como es el caso de la serie de experimentos realizados en el sistema SMART por Salton y sus asociados. En estos experimentos se realizaron comparativas entre la función de similitud por solapamiento y la fórmula del coseno; así como pruebas con términos ponderados en función de su frecuencia de aparición en los documentos.

Modelo Probabilístico

El modelo probabilístico más usado hoy en día es el de Robertson y Sparck Jones (1976). Este modelo se basa en la premisa de que los términos que aparecen en los documentos relevantes recuperados previamente para una consulta dada, deben tener mayor peso que si no hubieran aparecido en esos documentos. De esta forma presentaron la siguiente tabla mostrando la distribución del término *t* en documentos relevantes y no relevantes para una consulta *q*.

		Document Relevance		
		+	−	
Document Indexing	+	r	$n - r$	n
	−	$R - r$	$N - n - R + r$	$N - n$
		R	$N - R$	N

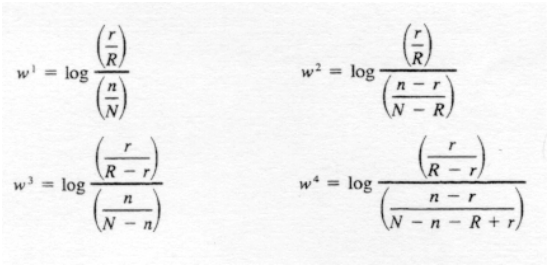
N = número total de documentos en la colección

R = número de documentos relevantes para la consulta *q*

n = número de documentos con el término *t*

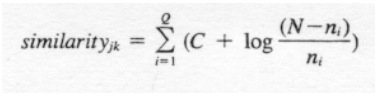
r = número de documentos relevantes con el término *t*

A partir de esta tabla desarrollaron cuatro fórmulas que reflejan la distribución relativa de los términos en documentos relevantes y no relevantes, y propusieron que esas fórmulas fueran usadas para la ponderación de términos.



Experimentaron con estas cuatro fórmulas sobre la colección Cranfield (indexada manualmente), usando todos los criterios de relevancia para ponderar términos y ver cuál sería el resultado óptimo. Tras estudiar los resultados verificaron de forma práctica la superioridad teórica de la cuarta fórmula (F4). Posteriormente, Sparck Jones investigó el uso de esta teoría como medida predictiva, obteniendo los mejores resultados con una versión ligeramente modificada de F4. Empleó esta medida de forma que simulase una típica sesión on−line (teniendo en cuenta el feedback de relevancia) y descubrió que la recuperación era más satisfactoria cuando sólo se tenían en cuenta los pesos de los dos primeros documentos relevantes recuperados en el ranking.

En ese momento la utilización de la indexación probabilística requería emplear al menos unos pocos documentos relevantes, haciendo que este modelo estuviera más relacionado con el feedback de relevancia que con los esquemas de términos pesados. Así, en 1979, Croft y Harper publicaron una serie de experimentos en los que se usaba indexación probabilística sin ningún tipo de información acerca de la relevancia. Partiendo de una reafirmación probabilística de F4, asumían que todos los términos de la consulta tenían la misma probabilidad de aparecer en documentos relevantes; y derivaron una nueva fórmula para la ponderación de términos.



donde

Q = número de términos coincidentes entre el documento *j* y la consulta *k*

C = constante

ni = número de documentos en la base de datos conteniendo el término *i*

N = número de documentos en la base de datos

Esta fórmula combina un peso basado en el número de términos coincidentes y una ponderación de términos similar a la medida IDF. La constante *C* permite adaptar la fórmula a la colección con la que se esté trabajando, tomando un valor cercano a 1 cuando los términos coincidentes de la consulta se consideran significantes (discriminatorios), o valores menores cuando no es el caso.

En 1983 Croft mejoró este esquema de ponderación combinada para incorporar los pesos por frecuencia de aparición en los documentos, añadiendo una constante K para la adaptación a distintas colecciones (ajusta la importancia relativa de los dos esquemas de ponderación).

$$similarity_{jk} = \sum_{i=1}^{Q} (C + IDF_i) * f_{ij})$$

donde

Q = número de términos coincidentes entre el documento *j* y la consulta *k*

IDFi = peso IDF para el término *i* en toda la colección

$$f_{ij} = K + (1 - K)\frac{freq_{ij}}{maxfreq_j}$$

donde

freqij = frecuencia del término *i* en el documento *j*

K = constante

maxfreqj = frecuencia máxima de cualquier término en el documento *j*

Otros Modelos de Ranking para Documentos Individuales

En el desarrollo de estas medidas para la ponderación de términos fueron utilizados varios modelos distintos, como la medida de frecuencia del documento invertido (Inverted Document Frequency Measure) desarrollada por Sparck Jones en 1972, medida que ha sido ampliamente empleada en la implementación del modelo probabilístico y del modelo del espacio temporal.

Otros han probado con distribuciones de términos más complejas, de las que cabe destacar el modelo 2−Poisson propuesto por Bookstein y Swanson (1974) y probado por Harter (1975) y Raghavan (1983). Este modelo ha demostrado ser menos provechoso por la dificultad que entraña estimar los múltiples parámetros necesarios para la implementación.

Modelos de Ranking Orientados a Conjuntos de Documentos

Los más conocidos de estos modelos orientados a conjuntos de documentos son los modelos de clustering. En ellos la consulta se realiza contra un conjunto de documentos relacionados agrupados jerárquicamente.

También cabe destacar los modelos basados en la teoría de conjuntos difusos, que, a pesar de haber sido ampliamente propuestos, no han recibido suficientes implementaciones experimentales como para ser usados en la práctica (excepto cuando son combinados con búsquedas Booleanas).

La teoría de conjuntos sin depurar (rough sets) ha sido también aplicada a la recuperación de información, pero tampoco ha sido desarrollada suficientemente para ser usada en la práctica.

OTROS EXPERIMENTOS DE RANKING

Comparación Directa de Medidas de Similitud y Esquemas de Ponderación de Términos

Se han realizado varios estudios que examinan los diferentes factores relacionados con el ranking, que no están basados en ningún modelo en particular sino que usan métodos de comparación directa de medidas de similitud y esquemas de ponderación de términos.

Es el caso del experimento de Sparck Jones (1973) que investigó diferentes tipos de ponderación de frecuencia de términos tomando la frecuencia de términos en un documento, la frecuencia de términos en una colección, las apariciones de términos en un documento (medida binaria), y las apariciones de términos en una colección; normalizando estas medidas para la longitud del documento. Para este experimento utilizó cuatro colecciones indexadas manualmente con palabras clave, construyendo todas las consultas a partir de ellas. Los resultados mostraron que el uso de frecuencias (o apariciones) de términos en una colección siempre mejoraba el rendimiento, mientras que el uso de frecuencias (o apariciones) de términos en un documento solamente mejoraba el rendimiento en algunos casos. Este experimento no combinó diferentes esquemas de ponderación de términos.

El experimento de McGill (1979) seleccionó 67 medidas de similitud y 39 esquemas de ponderación de términos y las utilizó para clasificar los resultados de recuperaciones Booleanas, usando indexaciones tanto controladas (manuales) como no controladas (texto completo). En ambos casos encontró una diferencia significativa en el comportamiento de las medidas de similitud, identificando un grupo de aproximadamente 15 de ellas que funcionaban bastante mejor que las demás. Este grupo incluía la fórmula del coseno y la función del producto interno usadas en los modelos probabilísticos. Los resultados con esquemas de ponderación de términos fueron más irregulares, ya que no se hallaron diferencias significativas cuando se usaba vocabulario controlado (manual), encontrándose, en cambio, diferencias importantes trabajando con vocabulario controlado (texto completo).

Harman (1986) utilizó un enfoque diferente, seleccionando cuatro factores de ponderación de términos que habían demostrado su importancia en investigaciones pasadas y ensayando con ellos diferentes combinaciones para llegar a un esquema óptimo de ponderación de términos. Utilizó indexación de texto completo con diferentes colecciones de prueba normalizadas, realizando las consultas con el mismo tipo de indexación. Los cuatro factores investigados fueron: el **número de coincidencias entre un documento y una consulta**; la **distribución de un término dentro de una colección**; la **frecuencia de un término en un documento**; y la **longitud del documento**. Se usaron dos medidas distintas para la distribución de un término en un documento, la medida IDF de Sparck Jones y una implementación revisada de la medida de ruido. El ruido, en este caso, se refiere a la capacidad de un término para devolver información útil y examina la concentración de términos dentro de los documentos en lugar del número de apariciones. Usando los factores independientemente, la **distribución de un término dentro de la colección** daba un resultado casi dos veces mejor (para la colección Cranfield) que empleando la **frecuencia de un término en un documento**. Combinando la **frecuencia en un documento** con el IDF o la medida de ruido, y normalizando para la **longitud del documento** los resultados mejoraron en más de dos veces respecto a la utilización del IDF o el ruido por separado (también para la colección Cranfield). Otras colecciones demostraron una mejora inferior a nivel de resultados, aunque los esquemas de ponderación de términos mostraron siempre una mayor eficacia. La medida de ruido mostró consistentemente un rendimiento ligeramente superior al IDF, aunque la diferencia no fue significativa.

Ranking Basado en la Estructura de Documentos

Algunos experimentos de ranking se han basado en la estructura documental (o intradocumental) en lugar de la ponderación de términos descrita anteriormente. Bernstein y Williamson (1984) construyeron un sistema de recuperación con ranking para una base de conocimientos altamente estructurada, la Base de Conocimientos sobre la Hepatitis. Sus algoritmos de ranking usaron ponderaciones basadas en la importancia de los términos en colecciones enteras y en documentos individuales, así como en la posición de los términos en la estructura del documento, por ejemplo la presencia del término en párrafos resumen frente a la presencia del término en párrafos de texto. Para este experimento se desarrolló un esquema de ponderación muy elaborado, adaptado a la estructura particular de la base de conocimientos.

En SIBRIS, un sistema operativo de recuperación de información (Wade 1989), se utilizan también estructuras de documento y de consulta para influenciar el ranking, incrementando los pesos para los términos

en los títulos de documentos y disminuyendo los pesos para términos añadidos a una consulta desde un tesauro.

Técnicas de Ranking Utilizadas en Sistemas Operativos

Existen varios sistemas operativos de recuperación que han implementado algoritmos de ranking como centro de sus mecanismos de búsqueda.

El sistema SIRE implementado en la Universidad de Siracusa (Noreault 1977) es un sistema híbrido que combina búsqueda booleana y un esquema de ranking basado en el modelo vectorial y que pondera mediante el uso de la frecuencia de términos en documentos. Una aplicación comercial de este sistema, comercializada como Personal Librarian, utiliza el ranking basado en diferentes factores, incluyendo el IDF y la frecuencia de un término en un documento. Este sistema asigna valores más altos a los documentos que presentan un mayor número de coincidencias con los términos de las consultas, de los que normalmente se asignarían en los esquemas de ranking experimentales.

El sistema CITE, diseñado como un interfaz con MEDLINE (Doszkocs 1982), clasificaba los documentos basándose solamente en la ponderación IDF, puesto que no había información disponible acerca de las frecuencias en documento de los archivos de MEDLINE.

El proyecto OPAKI (Walker y Jones 1987) trabajaba con catálogos on−line y utilizaba solamente medidas de IDF.

CÓMO SELECCIONAR UNA TÉCNICA DE RANKING

Observando los resultados de los experimentos anteriormente descritos se pueden identificar diferentes tendencias.

- El uso de ponderación de términos basado en la distribución de un término en una colección siempre mejora el rendimiento (o al menos, no lo perjudica). La medida IDF se utiliza comúnmente, sea en su forma original o normalizada de algún modo.

$$IDF_i = \log_2 \frac{N}{n_i} + 1 \qquad (Sparck\ Jones\ 1972)$$

$$IDF_i = \log_2 \frac{maxn}{n_i} + 1 \qquad (Sparck\ Jones\ 1979)$$

$$IDF_i = \log_2 \frac{N - n_i}{n_i} \qquad (Croft\ and\ Harper\ 1979)$$

donde

N = número de documentos en la colección

ni = número total de apariciones del término *i* en la colección

maxn = frecuencia máxima de cualquier término en la colección

Una alternativa posible es la medida de ruido o entropía ensayada en algunos experimentos.

Ruido normalizadoi = maxnoise − noisei

$$noise_i = \sum_{k=1}^{N} \frac{Freq_{ik}}{TFreq_i} \log_2 \frac{TFreq_i}{Freq_{ik}} \qquad (Harman\ 1986)$$

$$entropy_i = 1 - \frac{\sum_{k=1}^{N} \frac{Freq_{ik}}{TFreq_i} \log_2 \frac{TFreq_i}{Freq_{ik}}}{\log_2 N} \qquad (Lochbaum\ \&\ Streeter\ 1989)$$

donde

N = número de documentos en la colección

maxnoise = ruido (*noise*) más alto de cualquier término en la colección

Freqik = frecuencia del término *i* en el documento *k*

TFreqi = frecuencia total del término *i* en la colección

- La combinación de la frecuencia en el documento con la ponderación IDF a menudo ofrece una mejora aún más importante. No obstante es muy importante normalizar de algún modo la frecuencia en el documento para moderar el efecto de los términos de alta frecuencia y compensar la longitud del documento. Esta normalización es particularmente crítica para datos indexados manualmente. Cualquiera de las fórmulas normalizadas de frecuencia en documento que se muestran a continuación pueden ser utilizadas con garantías de bien funcionamiento.

$$cfreq_{ij} = K + (1 - K) \frac{freq_{ij}}{maxfreq_j} \qquad (Croft\ 1983)$$

$$nfreq_{ij} = \frac{\log_2 (freq_{ij} + 1)}{\log_2 length_j} \qquad (Harman\ 1986)$$

donde

freqij = frecuencia del término *i* en el documento *j*

maxfreqj = frecuencia máxima de cualquier término en el documento *j*

lengthj = número de términos únicos en el documento *j*

- Si se utiliza la frecuencia en el documento se pueden usar varios métodos para combinarla con la medida IDF. La combinación recomendada para la mayoría de las situaciones se detalla a continuación.

$$similarity\ (Q,D) = \frac{\sum_{i=1}^{t} (w_{iq} \times w_{ij})}{\sqrt{\sum_{i=1}^{t} (w_{iq})^2 \times \sum_{i=1}^{t} (w_{ij})^2}}$$

donde

$$w_{iq} = \left(0.5 + \frac{0.5\, freq_{iq}}{maxfreq_q}\right) \times IDF_i$$

y

wij = *freqij* x *IDFi*

freqiq = frecuencia del término *i* en la consulta *q*

maxfreqq = frecuencia máxima de cualquier término en la consulta *q*

IDFi = IDF del término *i* en toda la colección

freqij = frecuencia del término *i* en el documento *j*

Salton y Buckley sugieren reducir la ponderación de la consulta *wiq* a la frecuencia en el documento (*freqiq*) para consultas largas que contienen múltiples apariciones de términos y usar la ponderación binaria de documentos (*wij* = 1 o 0) para colecciones con documentos cortos o colecciones que usan vocabulario controlado.

IMPLEMENTACIÓN DEL RANKING EN UN SRI

En este apartado describiremos una simple pero completa implementación del ranking en un sistema de recuperación de información.

La implementación engloba dos partes muy interrelacionadas: la indexación del texto y el uso (búsqueda) de ese índice para recuperar una lista de identificadores de registros en forma de ranking.

El índice utilizado es un fichero invertido que se crea una vez por cada actualización del sistema (tan solo una vez en colecciones estáticas) y que permite realizar búsquedas con mayor rapidez. Aunque también es posible construir un SRI con ranking sin ningún tipo de índice, el uso de estos índices incrementa la eficiencia. Para conseguir esta eficiencia es necesario actualizar el índice cada vez se producen cambios en el conjunto de datos.

La descripción del proceso de búsqueda no incluye especificaciones acerca del interfaz ni del proceso real de búsqueda de datos. Se asume que la consulta en lenguaje natural pasa al proceso de búsqueda de alguna forma, y que la lista de identificadores de registros que es devuelta por el proceso de búsqueda es introducida como input en alguna rutina que relaciona dichos identificadores con los registros a los que hace referencia (localización de los datos), mostrando una lista de títulos o cortos descriptores de los datos para facilitar la selección del usuario.

Creación del Fichero Invertido

El uso de un sistema de recuperación con ranking en lugar de un sistema booleano tiene varias implicaciones importantes para mantener las estructuras del fichero invertido.

- La utilización del ranking implica que no existe una necesidad (o esta es muy pequeña) de emplear operaciones de búsqueda para términos adyacentes ni de restricción de campos, cosas necesarias en búsquedas booleanas. De este modo, sólo son almacenados los identificadores de los registros y la localización de cada palabra, creando un índice mucho menor que el de los sistemas booleanos.
- El uso del ranking significa que las estrategias necesitadas en los sistemas booleanos para incrementar la precisión no son necesarias y deberían ser descartadas a favor de estrategias que incrementen la recuperación a expensas de la precisión.

Aunque un fichero invertido con información de frecuencia puede ser usado directamente por una rutina de búsqueda, normalmente se procesa en un formato final mejorado.

term	recno	freq
ab	2	1
being	2	1
charact	2	1
human	2	1
index	1	1
literat	1	1
novel	1	1
pap	1	1
report	1	1
report	2	1
technique	1	2
.		
.		

Fichero invertido con información de frecuencia

Este formato está basado en los métodos de búsqueda y de ponderación, pero una técnica de búsqueda común suele usar una rutina de búsqueda binaria en el fichero para localizar las palabras que correspondan a la consulta. Esto implica que el fichero buscado debe ser tan pequeño como sea posible, y por eso el fichero simple (como el mostrado), que contiene los términos, identificadores de registro y frecuencias, es normalmente dividido en dos partes para la búsqueda. Una de ellas es el diccionario que contiene los términos, con alguna estadística sobre el término (nº apariciones, IDF), y un puntero hacia la localización del fichero de apariciones (postings file) de ese término. El fichero de apariciones contiene los identificadores de registro y los pesos de cada una de las apariciones del término. De esta manera el diccionario usado en la búsqueda binaria sólo tendrá una línea por cada término único.

Existen cuatro opciones principales a la hora de almacenar la ponderación de términos en el fichero de apariciones:

- Almacenar la frecuencia real (raw frequency). Produce la búsqueda más lenta pero el sistema es más flexible ya que los algoritmos de ponderación de términos pueden ser cambiados sin variar el índice.
- Almacenar la frecuencia normalizada. Esta operación se haría durante la creación del diccionario final y de los ficheros de apariciones, y su frecuencia normalizada sería insertada en los ficheros de apariciones en lugar de la frecuencia real. La ventaja de esta opción es que la actualización (asumiendo sólo la adición de nuevos registros y la no modificación de los viejos) no requiere que los ficheros de aparición sean cambiados. Esta opción mejorará considerablemente el tiempo de respuesta respecto de la opción 1.
- Almacenar el término ponderado completamente. Esta opción permite una adición simple de cada peso durante el proceso de búsqueda y proporciona un tiempo de respuesta muy rápido. La desventaja es que la actualización requiere cambiar todos los ficheros de apariciones porque el IDF forma parte del fichero de aparición (y las medidas de IDF cambian cuando se hace alguna adición a la base de datos).
- Si no se usa ponderación en los registros, los registros de apariciones no tendrán que almacenar pesos. Todo el proceso se desarrolla en las rutinas de búsqueda.

Buscando en el Fichero Invertido

Una forma de usar un fichero invertido para recuperar un output con ranking estadístico es, primero, recuperar todos los registros que contengan los términos de consulta, y luego utilizar la información de los pesos de cada término en esos registros para calcular el peso total de cada uno de los registros recuperados, y finalmente clasificar esos registros. El tiempo de búsqueda depende estrechamente del número de documentos recuperados. Resulta prohibitivo utilizando grandes bases de datos.

Este proceso puede hacerse más independientemente del número de documentos recuperados, usando el método desarrollado por Doszkocs para CITE, que emplea técnicas de hashing para acumular el total de los pesos.

La búsqueda comienza aplicando un método de parsing sobre la consulta, siendo cada término comprobado contra la lista detención para desechar términos comunes. Si el término de consulta no es común pasa a través de la rutina de stemming, extrayendo la raíz del término, que será utilizada para ejecutar una búsqueda binaria contra el diccionario. Si la raíz se halla en el diccionario, se recupera la dirección de la lista de apariciones para esa raíz, así como el correspondiente IDF y el número de apariciones. El siguiente paso es utilizar las direcciones y el número de apariciones para leer los identificadores de registros, añadiendo la ponderación de términos total de cada registro en su respectivo acumulador. Llegado este punto, dependiendo de la opción elegida a la hora de ponderar los términos, seguiremos uno de los siguientes procesos:

- Si hemos utilizado la opción 3, este total está inmediatamente disponible, siendo sólo necesaria una simple adición.
- Si hemos seguido la opción 2, el peso almacenado en las apariciones es la frecuencia normalizada de la raíz en ese registro, y necesita ser multiplicada por el IDF de la raíz antes de la adición.
- Si hemos tomado como opción la 1, la totalidad del peso de los términos debe ser calculada, ya que el peso almacenado en las apariciones es la frecuencia real de la raíz en ese registro. Es esencial cargar en la memoria las estadísticas de registros necesarias, como la longitud de registro, antes de la búsqueda y así mantener un tiempo de respuesta razonable.

Cada vez que un término de consulta es procesado sus apariciones causan sucesivas adiciones en los acumuladores.

Cuando todos los términos de consulta han sido procesados, los acumuladores con peso diferente de cero son clasificados para producir una lista de registros en forma de ranking.

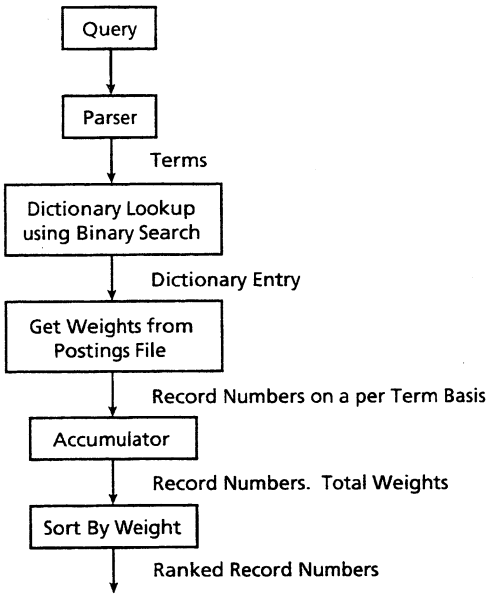


Diagrama de flujo del motor de búsqueda

Ranking 16

A

factores información ayuda humano operación recuperación sistemas

B

Consulta factores humanos en sistemas de recuperación de información

VECTOR (1 1 0 1 0 1 1)

C

Registro 1 contiene humano, factores, información, recuperación

VECTOR (1 1 0 1 0 1 0)

Registro 2 contiene humano, factores, ayuda, sistemas

VECTOR (1 0 1 1 0 0 1)

Registro 3 contiene factores, operación, sistemas

VECTOR (1 0 0 0 1 0 1)

D

Consulta (1 1 0 1 0 1 1)

Reg 1 (1 1 0 1 0 1 0)

(1 1 0 1 0 1 0) = 4

Consulta (1 1 0 1 0 1 1)

Reg 2 (1 0 1 1 0 0 1)

(1 0 0 1 0 0 1) = 3

Consulta (1 1 0 1 0 1 1)

Reg 3 (1 0 0 0 1 0 1)

(1 0 0 0 0 0 1) = 2