Interface de recuperación para catálogos en línea con salidas ordenadas por probable relevancia

Gustavo Gabriel Archuby
Licenciatura en Informática, Estudiante. Facultad de Informática, Universidad Nacional de La Plata.
E-mail: gustavoa@huma.fahce.unlp.edu.ar

Julián Cellini
Analista en Computación, expedido por la Facultad de Informática de la Universidad Nacional de La Plata. Promedio: 7,625.
E-mail: juliancellini@gmx.net

Claudia Marcela González
Bibliotecaria Documentalista, expedido por la Facultad de Humanidades y Ciencias de la Educación de la Universidad Nacional de La Plata, 1988.
E-mail: claudia@huma.fahce.unlp.edu.ar

Mónica Gabriela Pené
Bibliotecaria Documentalista, expedido por la Facultad de Humanidades y Ciencias de la Educación de la Universidad Nacional de La Plata, con fecha 20 de diciembre de 1996. Promedio: 9,67
E-mail: mpene@huma.fahce.unlp.edu.ar

Resumen

Se presenta el desarrollo de una interface de recuperación de información para catálogos en línea de acceso público (plataforma CDS/ISIS), basada en el concepto de similaridad para generar los resultados de una búsqueda ordenados por posible relevancia. Se expresan los fundamentos teóricos involucrados, para luego detallar la forma en que se efectuó su aplicación tecnológica, explícita a nivel de programación. Para finalizar se esbozan los problemas de implementación según el entorno.

Palabras-claves

Interfaces de recuperación de información; Ponderación de términos; Medida de similaridad; CDS/ISIS.

Opacs retrieval interface with ranked outputs

Abstract

Presents an information retrieval interface model for Public Access Catalogs (OPALs) on CDS/ISIS platform, based on the similarity principle. The proposal aims at ordering the results of vector queries according to their relevance. The underlying theoretical principle is described, as well as the implementation of the model.

Keywords

Information retrieval interfaces; Term weighting; Similarity measure; CDS/ISIS.

INTRODUCCIÓN

En estos últimos 50 años ha existido en los países desarrollados una preocupación constante por investigar e implementar técnicas que permitan recuperar información precisa. Desde mediados de siglo, los esfuerzos convergentes de distintas disciplinas: informática, lingüística, psicología, ciencias de la información, han dado origen a sistemas automáticos de recuperación de información de diferente nivel de complejidad. En el ámbito de la documentación, los más difundidos y utilizados internacionalmente son los que aplican técnicas basadas en la equiparación exacta *(exact matching)*, proximidad y álgebra de Boole[1].

Quizá uno de los principales problemas de estos sistemas tradicionales, provenga de la falta de asignación de grados de posible relevancia en las respuestas[2]. Esto es: el operador AND es demasiado restrictivo, todos los documentos que no cumplen con las condiciones de búsqueda establecidas quedan excluidos; y en contraposición, el operador OR es demasiado inclusivo provocando generalmente un problema de sobrerecuperación. Por ejemplo, si al momento de plantear una estrategia de búsqueda sencilla, se opta por relacionar dos términos con un AND, el usuario perderá la posibilidad de ver los documentos que sólo contienen uno de los términos. Si la misma búsqueda se plantea con un OR, el sistema traerá todos los registros que contienen al menos uno, pero no es capaz de ordenar los registros recuperados de forma que se muestren primero los que contienen ambos términos.

El fundamento teórico que permitió desarrollar las técnicas de salidas ordenadas por probable relevancia (*ranked output)*, correspondió al ámbito de la psicolingüística, concretamente a los trabajos de Zipf. Sus estudios establecieron que si se ordenan las palabras de un corpus textual en un rango de forma descendente por su frecuencia de aparición, y luego se multiplica el rango por la frecuencia, se obtiene un valor aproximadamente constante. La observación de que dicho valor era más estable en las frecuencias intermedias, le permitió concluir que es en esas palabras donde se deposita la significación de un texto[2, 3, 4].

frecuencia * rango ≅ constante

A fines de los 50, Luhn aplica esta idea a un sistema documental concluyendo, de forma similar, que el poder de resolución *(resolving power)* de un término de indización asignado en una base de datos, está en los términos de frecuencias intermedias. En este caso, el concepto de poder de resolución de un término – o "peso" como se denominará de aquí en más –, está relacionado con su capacidad de identificar material relevante dentro del corpus documental[5].

A partir de los trabajos de Luhn comienzan a desarrollarse funciones matemáticas que modelizan, con mayor rigor, el peso de un término. Algunas de estas funciones sirven para aplicar en sistemas que utilizan lenguaje controlado y otras para sistemas que emplean la lengua natural. Una de las más simples, y apropiada para la experiencia aquí planteada, fue propuesta por Sparck Jones [2, 5].

$$\textit{Peso del término}\ \mathbf{t = Log}_{2}\ \mathbf{(n\ /\ f) + 1}$$

donde:

• **n** es la cantidad de registros de la base de datos

• **f** es la frecuencia del término en la base de datos

Esta función supone que el peso de un término es inversamente proporcional a la cantidad de documentos que lo poseen (véase tabla 1). Puede decirse entonces que, cuanto más frecuente es un término en una base de datos, menor es la información que proporciona y, por ende, más bajo su peso.

Gerald Salton utiliza este concepto de peso en su modelo de recuperación basado en el espacio vectorial (proyecto SMART, 1968). En dicho modelo, se forma una matriz término / documento que representa la base de datos. Cada vector de la matriz representa un documento; cada elemento del vector tendrá valor 0 (cero) si dicho documento no contiene el término; o el valor del peso del término si lo contiene.

Vectorizando a su vez la expresión de búsqueda formulada por el usuario *(query vector)*, desarrolla un nuevo modelo matemático para la recuperación de información basado en el cálculo del coeficiente de ***similaridad*** entre vectores. Este coeficiente permite determinar las similitudes y diferencias entre los documentos de una base de datos y / o entre éstos y la expresión de búsqueda introducida por el usuario.

Para realizar el cálculo de la similaridad entre dos vectores

TABLA 1

(peso alto = mucha Inf.) ***(peso bajo = poca inf.)***	Frecuencia **alta** en la **base**	Frecuencia **baja** en la **base**
Frecuencia **alta** en el **documento**	**Peso medio**	**Peso alto**
Frecuencia **baja** en el **documento**	**Peso bajo**	**Peso medio**

existen diversas funciones, siendo las más conocidas la del producto escalar de dos vectores y los coeficientes del coseno, Dice y Jaccard. Para el presente desarrollo se ha seleccionado el coeficiente de Dice, ya que la bibliografía lo presenta como una de las funciones para aplicar en recuperación de información [5].

Coeficiente de **Dice**:

$$\frac{\mathbf{2 * \Sigma\ (PQi * PDi)}}{\mathbf{\Sigma\ (PQi\ ) + \Sigma\ (PDi)}}$$

Donde:

• **PQi:** es el peso del término i en el documento Q, o 0 (cero) si el documento Q no tiene el término.

• **PDi:** es el peso del término i en el documento D, o 0 (cero) si el documento D no tiene el término.

Al hacer el cálculo del coeficiente de similaridad de los documentos y del vector de búsqueda, y someterlos a una comparación sistemática, se está en condiciones de establecer un orden descendente, colocando en primer término el documento cuyo valor es más cercano al del vector de búsqueda y así hasta concluir con todos los registros resultantes. Estos registros son los mismos que se obtienen al hacer un OR entre todos los términos que se utilizan en la interrogación.

CARACTERÍSTICAS DE LA INTERFACE

La presente interface ha sido desarrollada con la finalidad de realizar una aplicación concreta de algunos de los principios teóricos de la recuperación de información en forma automática. No obstante, podría aportar en la práctica una posibilidad sustancial de mejorar el acceso a la información de cualquier OPAC sobre plataforma CDS/ISIS en Internet.

Una de las herramientas que permite la operación de bases MicroIsis desde la Web es el WWWISIS 4.0, también conocido como WXIS, desarrollado por el grupo de programación de BIREME. Mediante la interfaz CGI (*Common Gateway Interface*), este software opera como nexo entre las bases de datos y un servidor www, permitiendo la visualización de los registros de las bases ISIS en formato HTML en el visualizador del cliente.

La solicitud que realiza el cliente se hace a través de un formulario que completa el usuario. Los datos enviados son tomados por el script CGI (en este caso por un script WWWISIS) que recupera la cadena de caracteres que tiene los datos y la utiliza como expresión de búsqueda en la base de datos. Es el mismo script el que se encarga de devolver los registros resultantes en formato HTML al cliente.

La interface web desarrollada para este estudio (véase fig. 1) consta de un formulario con las siguientes opciones:

a) el cuadro de texto: donde el usuario registra todos y cada uno de los términos que representan el concepto que está buscando. Dichos términos se registran uno debajo de otro.

b) porcentajes de similaridad documental: donde el usuario establece el grado de similaridad mínimo entre la información que busca (indicada en los términos ingresados – que componen el vector de búsqueda –) y los documentos recuperados.

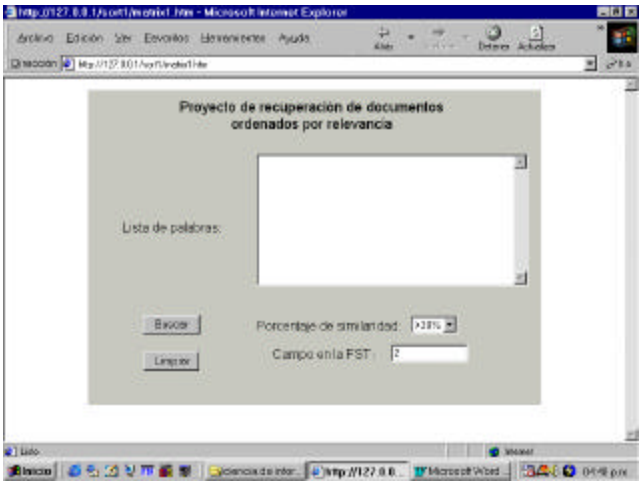
c) campo de la FST: campo de la base de datos sobre el que se efectúa la búsqueda.

d) el botón de Buscar: inicia la acción.

e) el botón Limpiar: borra los datos registrados en el formulario, preparándolo para recibir nueva información.

Un detalle interesante de esta interface es que brinda la posibilidad de que el usuario determine si desea recuperar sólo documentos con un alto porcentaje de similaridad o bien visualizar todos los registros recuperados (listados en orden decreciente según el porcentaje de similaridad).

FIGURA 1
Formulario web diseñado para este estudio



Haciendo una descripción más profunda, debe mencionarse también que fue necesario generar otra base de datos complementaria al OPAC. El ambiente de programación WWWISIS 4.0 no provee una función que calcule logaritmos, con lo cual el cálculo constante que debe realizarse para conocer el "peso" de los términos hace poco óptimo el rendimiento de la aplicación. Se optó, entonces, por implementar una tabla de logaritmos (en base DOS) como un archivo maestro ISIS, en el cual se guarde el logaritmo de 1 en el registro 1, el logaritmo de 2 en el registro 2, y así sucesivamente. A dicha base se accede mediante la función REF del lenguaje de formateo.

DESCRIPCIÓN GENERAL DEL SCRIPT

El funcionamiento de la aplicación comienza cuando el usuario envía los términos a buscar. Dichos términos se utilizan para generar el vector de búsqueda, y además para seleccionar de la base de datos todos los documentos que contienen por lo menos uno de ellos en el campo descriptores.

Luego se toma cada documento recuperado y se calcula la similaridad con el vector de búsqueda mediante la función de similaridad. Si ésta es menor que el mínimo estipulado por el usuario, el documento es descartado; de lo contrario, es insertado en una lista ordenada en forma decreciente de acuerdo al grado de similaridad con el vector de búsqueda. Una vez procesados todos los documentos, son devueltos al usuario.

El script consta de un cuerpo principal y dos funciones. Una de las funciones calcula la similaridad entre dos documentos, y la otra, obtiene el peso del término en la base de datos. Esta última función se aplica al campo descriptores de la base de datos, porque se necesita que el término sea único por registro (véase *Apéndice*).

DISCUSIÓN

La realización de este desarrollo con la única finalidad de profundizar en el estudio del concepto de similaridad documental, permite plantear algunos interrogantes:

1. Si al efectuar búsquedas específicas, con más de 2 términos, no se obtienen resultados numerosos de nuestros OPACs, ¿se justifica el esfuerzo del desarrollo de una interface que los ordene por posible relevancia? Si un resultado de búsqueda ofrece 10 registros para visualizar, ¿es significativo que los devuelva ordenados?

2. Con CDS/ISIS como herramienta de desarrollo —reconocida es la potencia de su motor para realizar búsquedas booleanas—, ¿no sería factible obtener resultados ordenados, sin aplicar cálculos de similaridad documental, sólo haciendo variaciones de AND y OR entre los términos de búsqueda?

3. Los usuarios de nuestros OPACs, ¿reclamarán interfaces simples, sin uso de operadores, y resultados "rankeados" tal como los que ofrecen los actuales buscadores de Internet?

Finalmente cabe reflexionar que el presente estudio se enmarca dentro del modelo vectorial, el cual conforma, junto con el modelo probabilístico, el área "dura" de la teoría de recuperación de información. Si partimos de suponer que la búsqueda de información es el problema clave de la Ciencia de la Información, esto implica que el análisis de cualquiera de los temas claves de la Bibliotecología debería plantearse a la luz de sus teorías. Estas teorías provienen de diferentes áreas del conocimiento, principalmente de las que estudian cómo el hombre conoce y procesa información, por lo cual, los modelos matemáticos conforman una pequeña porción dentro de las posibles elecciones para abordar el problema de la recuperación de información.

BIBLIOGRAFÍA

1. FERNÁNDEZ MOLINA, J. C.; MOYA ANEGÓN, F. de. *Los catálogos de acceso público en línea* : el futuro de la recuperación de información bibliográfica. Granada : Asociación Andaluza de bibliotecarios, 1988.
2. MOYA ANEGÓN, F. de L*os sistemas integrados de gestión bibliotecaria:* estructura de datos y recuperación de información. Madrid : ANABAD, 1995.
3. MOYA ANEGÓN, F. de; LÓPEZ GIJÓN, J.; GARCÍA CARO, C. *Técnicas cuantitativas aplicadas a la biblioteconomía y documentación*. Madrid : Síntesis, 1996.
4. RIJSBERGEN, C.J. van. *Information retrieval*. 2. ed. London : Butterworths, 1979.
5. SALTON, G.; McGILL, M. *Introduction to modern information retrieval.* New York : McGraw-Hill, 1983.

Artigo aceito para publicação em 12-09-2000

Apéndice
Funciones explícitas en el script

IsisScript	**Descripción**
<function name=PesoCont action=replace tag=”100"> <do task=keyrange>	Encabezado de la función que calcula el peso
<!— En el campo 100 se distinguen los siguientes subcampos: ^t término ^b base de datos ^n tag —> <parm name=db><pft>v100^b</pft></parm> <parm name=from><pft>v100^t</pft></parm> <parm name=count>1</parm> <parm name=posttag><pft>v100^n</pft></parm> <field action=define tag=1001>Isis_Current</field> <field action=define tag=1>Isis_Key</field> <field action=define tag=2>Isis_Postings</field> <field action=define tag=3>Isis_Posting</field>	*Definición de constantes y variables necesarias para el cálculo del peso.* Se asigna la BD (v100^b), el Término a recuperar (100^t). Identificador del campo Descriptores En los campos siguientes: 1= término 2= cantidad de postings del término 3= datos del posting
   <field action=statusdb tag=4><pft>v100^b</pft </field> <field action=replace tag=5><pft> ref([‘e:\inetpub\wwwroot\cgi-in\wxis\logs\log2’]val(v4^n)-1,v1) </pft></field>	*Cálculo del logaritmo de la cantidad de registros de la base de datos.* Se obtiene información de la BD (campo 4 subcampo n = cantidad de registros + 1) Se calcula el logaritmo de la cantidad de registros (v4^n) y se le resta 1, guardándose el resultado en el campo 5.
  <loop> <field action=import tag=5>5</field> <field action=replace tag=120><pft> f(val(v5)-val(ref([‘e:\inetpub\wwwroot\cgi-bin\wxis\logs\log2’]val(v2),v1))+1,1,4),/ </pft></field> <field action=export tag=120>120</field> </loop>	*Cálculo del peso.* v5= Logaritmo de la cantidad de registros La expresión señalada con verde calcula el logaritmo de la frecuencia del término en la BD. En el campo 120 se obtiene el peso que le corresponde al término en cuestión.
</do> <return action=export tag=100>120</return> </function> <!— ————Fin Función para obtener pesos——— —>	Como resultado de la función se devuelve, en el campo 100, el peso del término.

IsisScript	**Descripción**
<function name=SimilDice action=replace tag="100" split=occ>	*Encabezado de la función que calcula la similaridad*
<! En el campo 100 se distinguen dos ocurrencias una por cada documento a comparar – con los correspondientes subcampos: ^d para un término del primer documento ^q para un término del segundo documento ^b base de datos ^t tag en la FST del cual se extrae el término –> <field action=add tag=2000><pft>(v100^b)</pft></field> <field action=add tag=2004><pft>(v100^t)</pft></field> <list action=delete>now</list> <list action=load type=freq><pft>mhu (v100^d/)</pft></list> <list action=load type=freq><pft>mhu (v100^q/)</pft></list>	*Definición de constantes y variables necesarias para el cálculo de la similaridad.* Se asigna la BD (v100^b). Se definen las características de la lista de términos de los documentos a comparar.
<do task=list> <parm name=reverse>On</parm> <parm name=sort><pft>f(val(v2),10,0)</pft></parm> <field action=define tag=1001>Isis_Current</field> <field action=define tag=1003>Isis_Total</field> <field action=define tag=1>Isis_Item</field> <field action=define tag=2>Isis_Value</field> <field action=replace tag=1101>0</field> <field action=replace tag=1102>0</field> <loop> <!—Nombre de la base de datos —> <field action=import tag=2000>2000</field> <!—Tag de la Fst del que se extrae el término —> <field action=import tag=2004>2004</field> <!—Campo donde se genera el divisor —> <field action=import tag=1101>1101</field> <!—Campo donde se genera el dividendo—> <field action=import tag=1102>1102</field>	*Generación de la lista* Se genera una lista que contiene los términos de ambos documentos. $\frac{2 * \Sigma (PQi * PDi) \quad \text{Dividendo}}{\Sigma (PQi) + \Sigma (PDi) \quad \text{Divisor}}$ En el campo 1101 se genera el dividendo. En el campo 1102 se genera el divisor. Las instrucciones encerradas entre <loop> y </loop> se repiten para cada elemento de la lista.

<!–Cálculo del peso del término que se esta computando—>

```
<call name=PesoCont>
        <pft>'^t'v1, '^b'v2000, '^n'v2004</pft>                    Se llama a la función que calcula el peso del término.
</call>

<!–Generación del divisor –>
<field action=replace tag=1101><pft>                               Se obtiene el divisor (v1101).
        f(val(v100)*val(v2)+val(v1101),3,4),
</pft></field>

<!–Generación del dividendo –>
<field action=replace tag=1102>                                    Se obtiene el dividendo (v1102).
  <pft>
    if val(v2)>1 then
        f(val(v100)*val(v100)+val(v1102),1,4),
    fi,
  </pft>
</field>

    <field action=export tag=1101>1101</field>
    <field action=export tag=1102>1102</field>

    </loop>
</do>

<!–Cálculo de la similaridad –>
                                                                   Cálculo de la similaridad
<field action=replace tag=100>
    <pft>
f(val(v1102)*2/val(v1101),4,4)                                     Similaridad: 2*v1102 / v1101
    </pft>
</field>

<return action=export tag=1100>100</return>
  </function>                                                      El resultado se devuelve en el campo 1100.

<!— ———————— Fin Dice———————— —>
```

Cuerpo principal del script

IsisScript	Descripción
<section> <display><pft>'content-type: text/html'/#</pft> </display> <display><HTML></display> <display><BODY></display>	*Marcas HTML que generan la página de resultados.*
<field action=cgi tag=2000>base</field> <field action=cgi tag=2002 split=occ>palabras</field> <field action=cgi tag=2004>FstTag</field> <field action=cgi tag=2005>descriptores</field> <field action=cgi tag=6010>similaridad</field> <field action=replace tag=2002 split=occ><pft>v2002</pft></field>	*Parámetros de entrada*
<display><pft>(v2002| |)</pft></display> <display><pft>'Base: 'v2000' '</pft></display> <display><pft>'Date: 'date</pft></display> <display><pft>'<hr>'</pft></display>	*Presentación de datos para la interface*
<do task=search> <parm name=expression><pft>(v2002+| or |)</pft> </parm> <parm name=db><pft>v2000</pft></parm> <parm name=gizmo>d:\isis\data\gizmo\mayuscu</parm> <loop> <field action=import tag=1200>1200</field> <field action=import tag=2000>2000</field> <field action=import tag=2002>2002</field> <field action=import tag=2004>2004</field> <field action=import tag=2005>2005</field> <field action=import tag=6010>6010</field> <field action=replace tag=2003 split=occ> <pft>(|^q|v2002/)(|^d|v53/)</pft> </field> <call name=SimilDice> <pft>(v2003/)'^b'v2000'^t'v2004</pft> </call> <field action=add tag=1200> <pft>if val(1100)>val(v6010) then '^k'v1100'^m'mfn fi,</pft> </field> <field action=export tag=1200>1200</field> </loop> </do>	*Búsqueda / Cálculo de similaridad* Se realiza la búsqueda estableciendo una operación de unión (OR) entre los términos que ingresa el usuario. Para cada registro recuperado, se calcula la similaridad entre los términos de búsqueda y los términos del registro recuperado, asignando dicho valor al campo 1100. Si el valor de similaridad es mayor que el porcentaje ingresado por el usuario, se agrega como una ocurrencia al campo 1200, junto con el MFN del registro. *Ordenación de los resultados*

```
 <list action=delete>now</list>
<list action=load><pft>(v1200/)</pft></list>

<do task=list>

<parm name=reverse>On</parm>

<field action=define tag=1001>Isis_Current</field>
<field action=define tag=1002>Isis_Itens</field>
<field action=define tag=3000>Isis_Item</field>
<parm name=sort><pft>v3000^k</pft></parm>
    <loop>
        <field action=import tag=2000>2000</field>

        <display><pft>'<br>',v1001,'/',v1002,'simil:'
v3000^k,c10,ref([v2000]val(v3000^m),mdl,'mfn: 'mfn'
',v10,('<br>'v53)/)'<hr>'</pft></display>
    </loop>
 </do>
```

Se genera una lista que se ordena, en forma descendente, por el valor de similaridad.

Se visualizan los registros recuperados, ordenados.

Marcas HTML que cierran la página de resultados.

```
        <display></BODY></display>
        <display></HTML></display>

 </section>

</IsisScript>
```