

José Luis López Pino

Estudiante de Ingeniería Informática de la Universidad de Granada; Finalista del IV Concurso Universitario de Software Libre en la categoría "Comunidad"

<jllopezpino@gmail.com>

1. Introducción

1.1. Objetivos propuestos

Un metabuscador es un buscador que, en vez de indexar contenidos, realiza consultas a otros buscadores y los clasifica y muestra como una única lista (en el caso de Visuse, de forma visual), consiguiendo así mayor cantidad de resultados sin que el usuario tenga que consultar uno por uno cada buscador.

Los objetivos que nos propusimos al plantear este proyecto fueron los siguientes:

- Lograr la intercomunicación con los distintos buscadores (diseño modular).
- Organizar la información proveniente de los buscadores.
- Puntuación de los distintos resultados, según la cadena de búsqueda introducida.
- Mostrar los resultados como un "muro", aprovechando al máximo la ventana del navegador.
- Mostrar los mejores contenidos con un tamaño mayor para darles mayor relevancia.

1.2. Antecedentes

Si consultamos cualquier lista de sitios más visitados de la red, encontramos que la clasificación está encabezada por buscadores, redes sociales y sitios de contenidos multimedia tal y como se puede apreciar en Alexa [1].

Es lógico que desde que los buscadores indexan la red y ayudan a los usuarios a encontrar recursos hayan surgido distintos buscadores especializados en contenidos multimedia y los buscadores tradicionales también hayan desarrollado sus versiones centradas en dichos contenidos. Sin embargo, estos buscadores siguen mostrando estos resultados de forma similar a los textuales, organizándolos en páginas con muy pocos resultados y añadiéndoles *thumbnails* (imágenes en miniatura) haciendo la búsqueda lenta e incómoda.

Recientemente ha habido algunos intentos para mejorar estas interfaces:

- El buscador Bing añadió un servicio llamado Visual Search [2] cuyo objetivo es sustituir las tradicionales búsquedas de texto por búsquedas a través de imágenes clasificadas en categorías y subcategorías, y para ello diseñaron una nueva interfaz utilizando Silverlight. Sin embargo, el objetivo de este servicio no es encontrar imágenes u otros contenidos multimedia, sino reemplazar las tradicionales búsquedas de texto por otras visuales.
- oSkope [3] es otro buscador que permite, mediante Flash, visualizar de distintas maneras los resultados de otros buscadores como Google Images o Youtube. Sus principales

Visualizando los resultados de búsqueda a través de Visuse

Resumen: Visuse es un acrónimo de VISUal Search Engine, ya que el proyecto consiste en un metabuscador que clasifica y muestra los resultados obtenidos de distintos buscadores y sitios web de forma visual, centrándose sobre todo en contenidos multimedia como imágenes y vídeo. Esto permite visualizar de una forma más cómoda los resultados, además de resultar muy útil para niños, personas que tengan problemas para leer, o dispositivos en los que sea incómodo leer.

Palabras clave: algoritmo voraz, búsquedas en Web, enfriamiento simulado, metabuscadores, presentación visual, relevancia de resultados, visualización de resultados, Visuse.

Autor

José Luis López Pino es estudiante de Ingeniería Informática de la Universidad de Granada (UGR) y se encuentra actualmente estudiando Computing (Web Systems Development) en Glasgow (Escocia) gracias a una beca Erasmus. Visuse es su Proyecto Fin de Carrera bajo la dirección del profesor Juan Julián Merelo Guervós. Su principal afición es la Informática, en especial todo lo que tenga que ver con la web (es socio fundador y secretario de la Asociación de Webmasters de Granada) y el software libre (colabora con la Oficina de Software Libre de la UGR).

desventajas son que no permite combinar los resultados de distintos buscadores, no ordena por puntuación los resultados en el mural y no permite visualizar los contenidos multimedia en el propio buscador.

- El servicio Speszify [4] sí que permite combinar resultados de distintos buscadores multimedia y visualizarlos en el propio buscador. Sin embargo, no realiza ninguna ordenación según la importancia de los resultados, y su distribución en un mural con grandes huecos por el que tenemos que ir desplazándonos hace muy difícil encontrar los resultados que deseamos.

- Recientemente, Google ha modificado la interfaz de su servicio de imágenes, para que éstas cobren más protagonismo en las búsquedas, mostrándolas también en forma de muro, aunque los resultados más relevantes simplemente aparecen más arriba.

Como podemos apreciar, ninguno de los buscadores existentes ha trabajado la disposición de las imágenes en función de su importancia ni la disposición óptima de los resultados para formar un muro que no desaproveche espacios. Estas observaciones constituyeron un incentivo para la formulación de los objetivos de nuestro proyecto.

1.3. Requisitos

Para llevar a cabo estos objetivos planteamos una serie de requisitos que la aplicación sería conveniente que cumpliera:

- Que funcionase en la mayor cantidad de navegadores posible, pero sin renunciar a funcionalidades importantes.
- Capacidad de configuración tanto del lado del cliente como del servidor.
- Fácil extensibilidad mediante módulos para los distintos buscadores.

2. Aspectos técnicos

El funcionamiento de Visuse en su interacción con el usuario, consiste en los siguientes pasos:

- El usuario introduce un término en el cuadro de texto.
- El cliente (JavaScript), solicita al servidor de Visuse que busque dicho término en una serie de buscadores. Se trata de peticiones HTTP corrientes.
- El servidor Visuse a su vez transmite a los distintos buscadores la petición del usuario y recibe la respuesta. La forma de comunicarse con cada uno de los buscadores y la información que nos comunican es completamente distinta, por ello del lado del servidor se ejecuta un módulo para cada uno de los distintos buscadores utilizados.
- Conforme recibe las respuestas, determina la importancia de cada resultado y los devuelve en un formato común, en este caso JSON, aunque fácilmente se podría implementar en XML u otro.
- El servidor va enviando los resultados al cliente, como respuesta a las peticiones HTTP que había realizado.
- El cliente estudia la mejor manera de mostrar los resultados y los muestra al usuario, paginándolos si es necesario.

3. Funcionalidades

3.1. Características

Las principales características que en este momento incluye el proyecto son:

- Módulos: Youtube, Picasa, Wikicommons, Flickr, Google Images y Yahoo Search.
- Un algoritmo "voraz" para mostrar los resultados.
- Algoritmo de puntuación para tres de los módulos.
- Ordenación y visualización de los resulta-

dos mostrados según la puntuación.

- Paginación de los resultados.
- Direcciones estáticas y amigables para las búsquedas.

3.2. Disposición de los resultados

Uno de los aspectos más innovadores del proyecto ha sido el de la disposición de los resultados en pantalla. Conociendo ya las experiencias en problemas similares, era necesario elegir una alternativa. El principal factor a tener en cuenta era la velocidad, ya que un tiempo de carga bajo es fundamental para una aplicación web y, durante la carga de las imágenes, se desea llamar repetidas veces al algoritmo, al tiempo que van llegando los resultados.

En un principio, el planteamiento era usar un algoritmo genético, aunque se descartó debido a que para obtener buenos resultados es necesaria una gran diversidad y un número elevado de iteraciones. La alternativa elegida para solucionar el problema del tiempo fue utilizar un algoritmo *greedy* (voraz o goloso) y, si la cantidad de huecos es muy elevada, realizar una búsqueda local para mejorar el resultado.

Para la representación de la disposición de las imágenes existen dos alternativas: almacenar las coordenadas en las que está situada la imagen o representar el espacio disponible como una rejilla y almacenar en una matriz si cada posición está disponible o no. La segunda alternativa supone almacenar más información pero es preferible a la primera debido a que si únicamente almacenamos las coordenadas es muy difícil comprobar los huecos para asignar las siguientes imágenes.

En diciembre de 2009, James Padolsey [5] utilizó un algoritmo "voraz" para resolver un problema muy similar al que estábamos estudiando: él trata de crear un muro a partir de imágenes, redimensionándolas a partir de un tamaño mínimo y máximo.

El algoritmo voraz que usa tiene como principales desventajas:

- Que no se encuentra en absoluto documentado.

- Los tamaños máximos y mínimos no son para cada imagen.
- La solución obtenida en algunos casos se encontraba muy lejos de ser la óptima.

Debido a que todas estas desventajas eran fácilmente evitables y a que el trabajo está publicado mediante una licencia libre, partimos de ese trabajo para aplicarlo al problema del proyecto. Para mejorar las soluciones obtenidas planteamos además aplicar la técnica del enfriamiento simulado [6] (también conocido como '*simulated annealing*' o 'recocido simulado'). La ventaja que presenta el enfriamiento simulado frente a la búsqueda local es que evita analizar la ejecución en óptimos locales.

Por lo tanto, partimos del trabajo de James Padolsey adaptándolo de forma que:

- Los tamaños varían en función de la importancia del resultado; se toma la puntuación del resultado para establecer unos tamaños máximos y mínimos de la imagen.
- Se utilizan dichos tamaños máximos y mínimos para cada imagen.
- Se organizan los resultados antes de empezar a utilizarlos.
- Se impide la reutilización de imágenes, ya que no tiene mucho sentido mostrar resultados por duplicado.

4. Conclusiones del proyecto

El proyecto ha logrado cumplir todos los objetivos básicos que nos habíamos propuesto, respetando las especificaciones planteadas. Hemos desarrollado buscadores para diferentes módulos que obtienen la información, la organizan y la puntúan según distintos clientes. Todo esto se realiza utilizando una estructura modular que permite una fácil extensibilidad de la aplicación, lo que ha permitido que terceras personas colaboren con el proyecto e implementen sus propios módulos para distintos buscadores, utilizando distintos sistemas de comunicación con ellos. Esta estructura modular también permite que los errores en unos módulos no afecten al resto.

Por otra parte, del lado del cliente se ha creado una agradable, a la par que simple, interfaz que muestra los resultados de una forma paginada y que da la sensación de rapidez al ir mostrando los resultados conforme son recibidos del servidor. Esto permite que la experiencia del usuario sea satisfactoria, además de que se le permite que no tenga que abandonar el buscador para consultar los resultados (ver **figura 1**). El resultado ha sido probado en distintos navegadores modernos con satisfactorio resultado, gracias a que ha sido desarrollado empleando estándares y tecnologías abiertas. Además, independientemente del navegador utilizado, los resultados se ajustan al espacio dejado libre por la ventana del navegador.

En cuanto a los métodos de disposición de los resultados hemos conseguido grandes resultados con el algoritmo voraz implementado al ejecutarse en un tiempo muy reducido, que además permite representar los resultados más interesantes (mejor puntuados) a un tamaño mayor y en las primeras páginas. La aplicación del enfriamiento simulado no ha sido satisfactoria, pero su estudio ha permitido sacar interesantes conclusiones sobre la conveniencia de aplicar este tipo de búsqueda y sobre cómo aplicarla.

Por último, la madurez del software desarrollado ha permitido que sea publicado e instalado para el acceso público en <<http://visuse.com>>. Visuse funciona actualmente en los navegadores Firefox, Chrome, Opera y Safari.

El desarrollo del proyecto también ha supuesto una interesante contribución a la comunidad de software libre, que puede reutilizar este trabajo para cualquier otro fin relacionado y que ha reconocido dicha contribución.

Enlaces de interés

Acceso público a Visuse para pruebas: <<http://visuse.com/>>.

Código fuente: <<https://forja.rediris.es/projects/cusl4-visuse>>.

Memoria del proyecto fin de carrera: <<http://www.slideshare.net/jlpino/memoria-4677729>>.

Blog del proyecto: <<http://visuse.wordpress.com/>>.

Cuenta Twitter: <[@visuse](https://twitter.com/visuse)>.

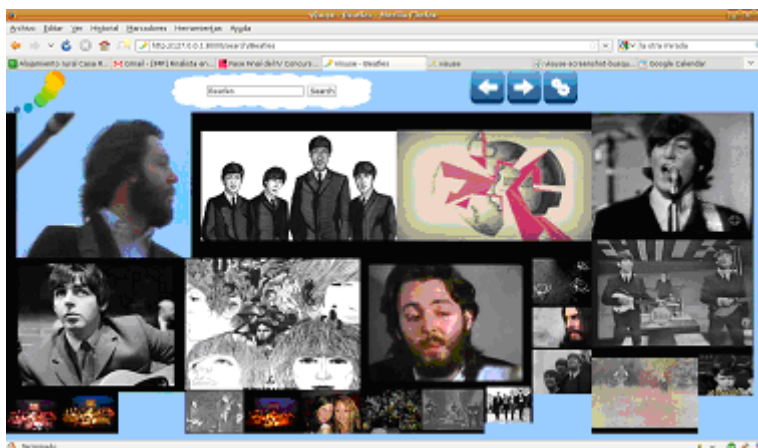


Figura 1. Resultados de buscar "Beatles" en Visuse.

Referencias

- [1] Alexa. <<http://www.alexa.com/topsites>>.
- [2] Bing. <<http://www.bing.com/visualsearch>>.
- [3] oSkope. <<http://www.oskope.com/>>.
- [4] Spezify. <<http://spezify.com/>>.
- [5] James Padosley. Gapsless Wall of Images, <<http://james.padosley.com/javascript/gapsless-wall-of-images/>>.
- [6] S. Kirkpatrick, C.D. Gelatt Jr, M.P. Vecchi. Optimization by simulated annealing. Science, 220(4598):671, 1983.