ReCIBE, Año 5 No. 2, Mayo 2016

Optimización del desempeño de un sistema de recomendación de documentos de texto basado en la configuración de los servidores.

Rodrigo Vences Nava
Universidad Autónoma de Yucatán
vnava@correo.uady.mx

Víctor Hugo Menéndez Domínguez
Universidad Autónoma de Yucatán
mdoming@correo.uady.mx

Alfredo Zapata González
Universidad Autónoma de Yucatán
zgonzal@correo.uady.mx

Resumen: Cuando se habla de sistemas informáticos en la Web, una de las configuraciones más utilizadas por los desarrolladores es la combinación de Apache y MySql, independientemente del sistema operativo sobre el cuál se desarrolle. En este trabajo se muestra cómo optimizar los tiempos de procesamiento y despliegue de información en el navegador Web de un Sistema de Recomendación de trabajos de titulación, aplicando sencillas configuraciones en el servidor Apache y siguiendo algunas buenas prácticas de programación al momento de ejecutar llamadas a la base de datos. Las configuraciones realizadas son específicamente en cuanto a la

compresión de datos y el manejo de caché. Este ahorro de tiempo repercute directamente en una mejor percepción por parte del usuario con respecto a la usabilidad del sistema.
Palabras clave: optimización, servidor Web, Apache, MySql.

Improvement performance of a recommendation system for text documents based on servers configuration

Abstract: When we speak about computer systems on the Web, one of the configurations more used by developers is the combination of Apache and MySql, regardless of operating system on which it develops. This paper shows how to optimize processing times and display of information in the Web browser for a recommendation system of graduation thesis using simple settings in the Apache server and following some good programming practices when executing calls to the database. The settings made are specifically data compression and cache management. This time saving directly affects in a better perception by the user regarding the usability of the system.
Keywords: Keywords: optimization, Web server, Apache, MySql

1. Introducción

Los Sistemas de Recomendación (SR) recopilan información sobre las preferencias de los usuarios sobre un conjunto de elementos o ítems (películas, canciones, libros, etc.), desde diferentes fuentes de información para proporcionar a los usuarios predicciones y recomendaciones de elementos similares, tratando de equilibrar factores como exactitud, novedad, dispersión y estabilidad en las recomendaciones (Bobadilla, Ortega, Hernando, & Gutiérrez, 2013).

Dentro de los esfuerzos que se han realizado para gestionar la información textual se encuentran principalmente los del área de recuperación de información (RI), la cual trata con la representación, almacenamiento, organización y acceso de elementos de información. Los documentos recuperados deben satisfacer las

necesidades de información del usuario expresadas en lenguaje natural (Baeza-Yates & Ribeiro-Neto, 1999).

La RI significa encontrar material (usualmente documentos) de una naturaleza no estructurada (usualmente texto) que satisface una necesidad de información desde dentro de grandes colecciones (generalmente se encuentran almacenados en computadoras) (Manning & Schutze, 1999).

Existen diversas herramientas para poder manejar el gran volumen de información, por ejemplo: clasificadores temáticos de textos, sistemas de detección de plagio, detección de reputación en redes sociales. Sin embargo todos tienen en común conocer el grado de similitud textual entre un par de textos (Álvarez Carmona, 2014).

Surge entonces la necesidad de procesar automáticamente ese gran volumen de información y para ello se ha recurrido a una de las ramas de la Inteligencia Artificial: el Procesamiento de Lenguaje Natural (PLN) (Manning, Raghavan, & Schütze, 2009).

El enfoque principal de esta área consiste en crear métodos, técnicas y herramientas computacionales que permitan realizar análisis de información escrita u oral y que faciliten la búsqueda y organización de dicha información (Karin Verspoor, 2013).

Algunos de los problemas que se presentan al desarrollar soluciones para manejar grandes cantidades de información textual, son los tiempos tanto de espera en el proceso de indexación, como de respuesta de la aplicación y la adecuada configuración de los servidores donde reside la aplicación, o las características de conectividad existentes dentro de una institución (Souders, 2007), los cuales impactan en el rendimiento del sistema y la satisfacción del usuario final.

Una posible solución para mejorar el desempeño de este tipo particular de sistemas como son los sistemas de recuperación de información y los sistemas de recomendación consiste en realizar modificaciones particulares a su código fuente, según recomendaciones o directrices de uso común (Souders, 2007). Sin embargo, esto origina problemas de mantenimiento, en especial cuando se trata de aplicaciones de libre distribución, donde las actualizaciones resultan muy

importantes. Por otro lado, adecuar la conectividad no siempre es posible dentro de una institución, dadas las políticas y normativas preestablecidas.

En este trabajo se propone una solución a esta problemática utilizando técnicas (manejo de caché y compresión de datos) que se centran en la configuración de los servidores donde se ejecuta nuestro caso de prueba, que es el Sistema de Recomendación de Trabajos de Titulación de la Facultad de Ciencias Antropológicas de la Universidad Autónoma de Yucatán (Vences, Menéndez, & Zapata, 2015) y que al aplicarlas en conjunto optimizan el desempeño del mismo en poco más del 50% en cuanto a velocidad de transferencia, considerando un esquema sólo de compresión de datos y hasta un 90.8% en cuanto a volumen de transferencia considerando un esquema de compresión de datos y manejo de caché. Además de optimizar hasta en un 99.98% los accesos a la base de datos, siguiendo las recomendaciones para la ejecución de sentencias SQL. Nuestra propuesta se ha implementado en el servidor Web Apache (https://httpd.apache.org/) y el servidor de base de datos MySql (https://www.mysql.com/).

2. Aspectos Relacionados

La accesibilidad de una aplicación se refiere a su utilización, en forma satisfactoria, por un mayor número de personas, sin importar sus limitaciones intrínsecas o derivadas del entorno en que se encuentran (W3C, 2010). Los tiempos de respuesta de la aplicación por lo tanto impactan en gran medida en la accesibilidad.

Algunas de las razones por las que una aplicación Web es lenta, por ejemplo, es que no fueron diseñadas pensando en la velocidad de carga sino en la apariencia (diseñador), o se tienen accesos a bases de datos sin considerar un tráfico pesado (Smith, 2013).

2.1 Optimización del servidor web Apache

La elección de la configuración correcta de los parámetros para optimizar el servidor web Apache no es un trabajo sencillo. Las mejores configuraciones

dependerán del hardware, las cargas de trabajo y cualquier proceso ejecutándose simultáneamente en el sistema. (Gandhi, Tilbury, Diao, Hellerstein, & Parekh, 2002).

Algunos autores, (Gandhi et al., 2002; Jun Li & Menghan Lu, 2013) implementan una optimización de desempeño dinámica en el servidor web Apache, centrando su atención en limitar el uso del CPU y la memoria, justificando que, cualquier cambio en el archivo de configuración de Apache con el fin de optimizar el desempeño, sólo funciona cuando el servidor es reiniciado. Este trabajo por el contrario se centra en aplicar ciertas configuraciones al archivo de configuración de Apache debido a que se tiene el control total del sistema.

En el desarrollo de aplicaciones Web existen directrices (Smith, 2013) para la optimización del desempeño que impactan fuertemente en la accesibilidad de la aplicación si son tomadas en cuenta, en especial el manejo de compresión y de caché tanto del cliente como del servidor y la replicación y buenas prácticas del uso de bases de datos. La figura 1 muestra el manejo del caché.

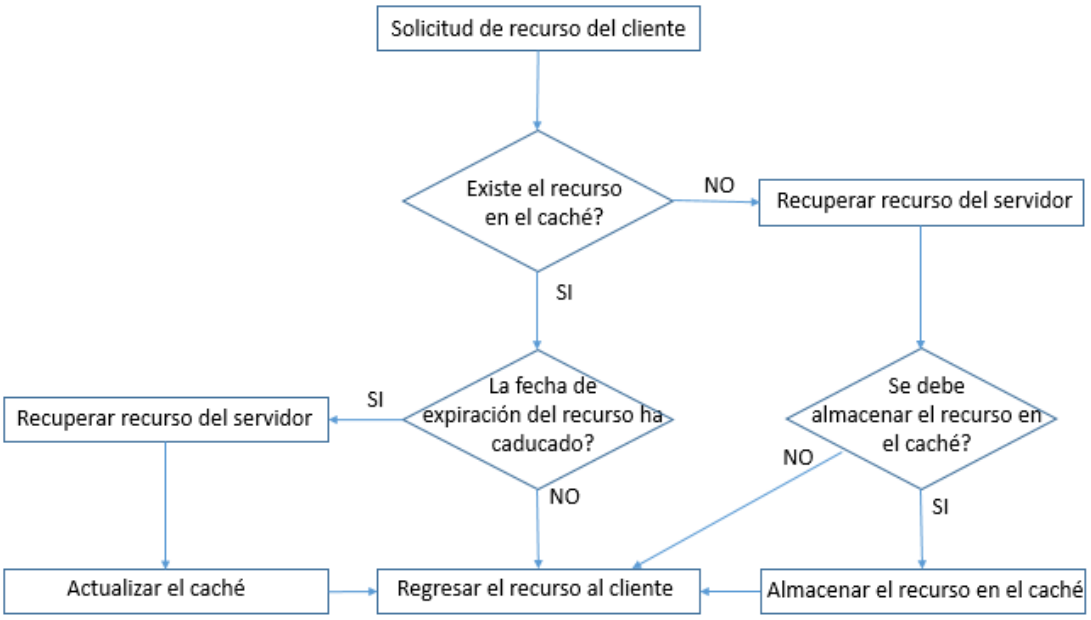


Figura 1. *Manejo del caché.*

A cada petición del usuario, el navegador determina si el recurso requerido ya ha sido solicitado recientemente para recuperarlo del caché y utilizarlo siempre y cuando no haya vencido su período de caducidad. Esto elimina el tiempo de espera por recibir el recurso del servidor y hace más rápida su presentación. Un proceso similar se puede hacer del lado del servidor para el manejo del caché.

De igual forma tanto en el cliente como en el servidor se puede emplear un esquema de compresión de recursos antes de almacenarlos en el caché y de esta manera se optimizan los tiempos de procesamiento y se agiliza la descarga de los recursos solicitados.

El esquema de compresión varía dependiendo del tipo de recurso, ya que si son imágenes o documentos PDF, la compresión está dada por el mismo formato del archivo; En cambio, si son hojas de estilo, scripts o páginas estáticas es posible comprimirlas e incluso minificar el código (Smith, 2013).

Diversos estudios han demostrado la factibilidad de aplicar éste y otro tipo de configuraciones a nivel de servidores, obteniendo reducciones significativas en volumen de transferencia y despliegue de la información, llegando a ser hasta del 40% (Ahuja, Wu, & Dixit, 2003; Menéndez, Castellanos, Aguilar, & Gómez, 2013).

2.2 Optimización de la base de datos MySql

En Ryser y Glinz (1999) se define un procedimiento para la recopilación de los requisitos y su documentación a través de escenarios, en el cual destacan 15 pasos para la creación de un escenario:

3. Patrones de Escenario

En el mundo contemporáneo el tamaño de las bases de datos crece exponencialmente y se hace inevitable la necesidad de una mejora en el desempeño de la misma (Myalapalli & Savarapu, 2014).

Cuando hablamos de sistemas de recuperación de información es muy importante seguir buenas prácticas al utilizar las sentencias SQL, debido a que el proceso de construcción del índice requiere de una gran interacción con la base de datos.

Por otro lado la replicación de base de datos (Fig. 2) es ampliamente utilizada para la tolerancia a fallos, escalabilidad y desempeño. El fallo en una base de datos con réplica no detiene el sistema. La escalabilidad puede ser alcanzada distribuyendo la carga a través de las réplicas o añadiendo nuevas réplicas si la carga aumenta. Finalmente la replicación de base de datos provee un acceso local rápido incluso si los clientes están distribuidos geográficamente, si las copias están localizadas cerca de los clientes (Kemme, Jimenez-Peris, & Patino-Martinez, 2010).

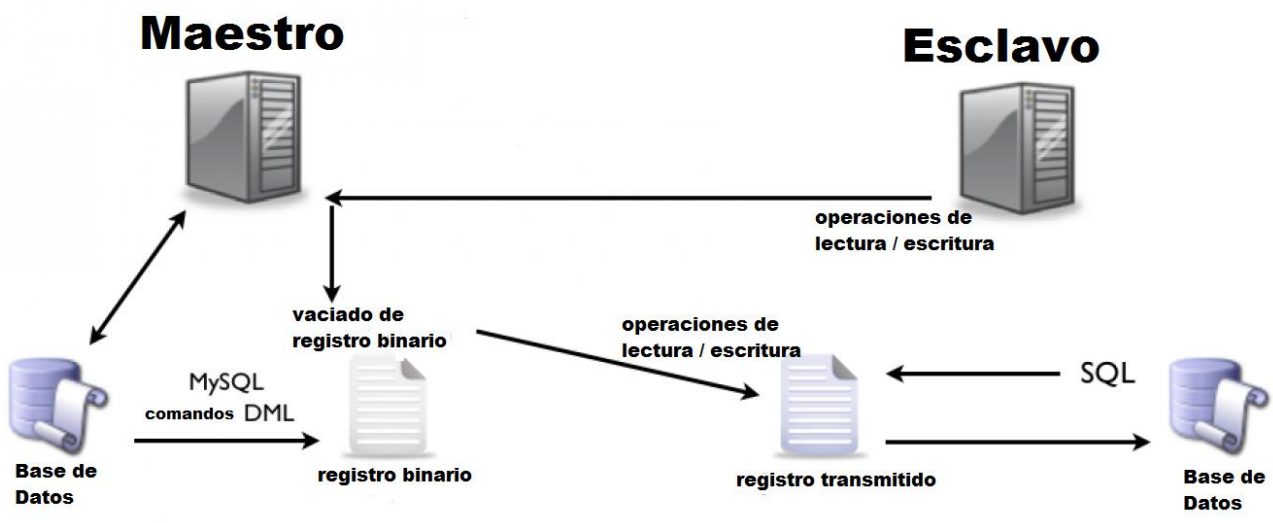


Figura 2. *Esquema de replicación maestro – esclavo de MySql.*

3. Caso de Estudio

Esta propuesta considera como caso de estudio el Sistema de Recomendación de Trabajos de Titulación de la Facultad de Ciencias Antropológicas de la Universidad Autónoma de Yucatán (Vences et al., 2015), que se desarrolló sobre el servidor Web Apache y el manejador de base de datos MySql.

Este sistema presenta una solución para la búsqueda, recuperación y recomendación de trabajos de titulación en dicha institución educativa, la cual cuenta con una matrícula de más de 700 alumnos y cerca de 100 profesores, quienes son beneficiados por el uso del sistema y por lo tanto cualquier

optimización en el desempeño del mismo repercute directamente en la percepción del usuario en cuanto a la rapidez de despliegue de la información y por tanto en su usabilidad.

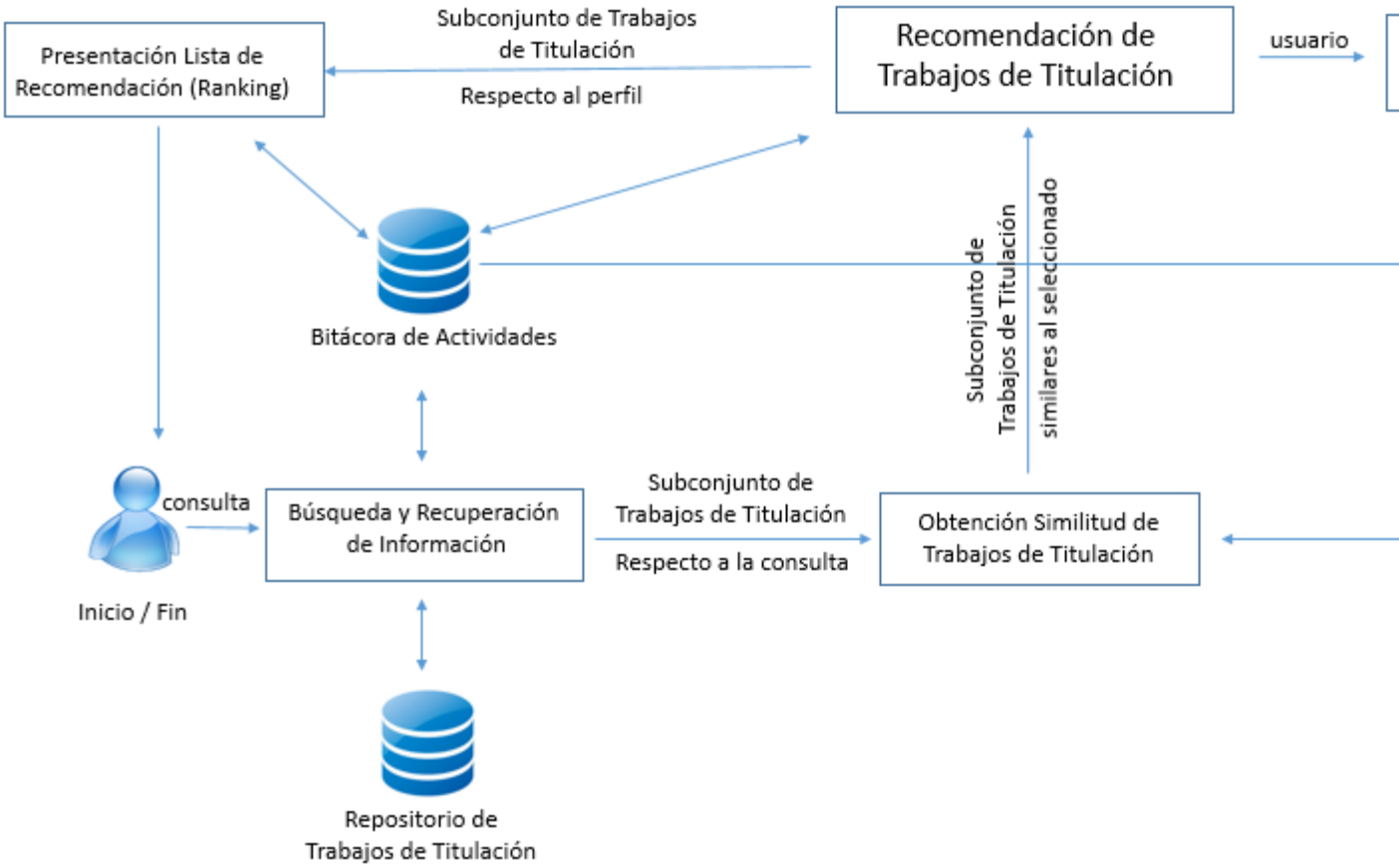


Figura 3. *Diagrama de funcionamiento del Sistema de Recomendación.*

Como puede observarse, en la Fig. 3 el usuario realiza una búsqueda al sistema, el cual le devuelve una lista de trabajos de titulación relevantes a su consulta y además le proporciona recomendaciones de trabajos similares a aquellos que ha consultado en el pasado o aquellos trabajos que han sido los más consultados.

Referente a las configuraciones de los servidores, para el caso de Apache nos centraremos en las configuraciones relativas al manejo de caché, para lo cual existen módulos que se activarán y adecuarán a nuestras necesidades. Un módulo de Apache es un componente que extiende las funcionalidades del servidor Web

(Apache HTTP Server Reference Manual – for Apache version 2.2.17, 2010). Los módulos mod_cache_disk, mod_expires, mod_deflate son las implementaciones de estas características. Para el caso de MySql nos referiremos a la replicación del servidor (“MySQL 5.7 Reference Manual,” 2015).

A continuación mencionaremos algunas recomendaciones al momento de ejecutar sentencias SQL que impactan en el tiempo de ejecución, las cuales se encuentran agrupadas en las siguientes categorías: control de caché, compresión de datos y base de datos.

3.1 Control de caché

La gestión de los encabezados, especialmente aquellos relacionados con el control de caché del cliente son realizados mediante el módulo Apache mod_expires.

Su propósito es controlar la fecha de expiración de los documentos solicitados en las peticiones HTTP, con lo que se puede indicar cuándo un documento fue modificado o su fecha de acceso por parte del cliente. Estos encabezados permiten controlar la persistencia y validez de un documento en el caché del navegador Web.

El módulo es selectivo en la inclusión de encabezados de expiración: si un documento solicitado al servidor tiene un encabezado de expiración entonces no modifica el encabezado.

Se ha establecido que todos los documentos PDF enviados al cliente tengan una fecha de expiración de un año ya que se tratan de trabajos de titulación que no van a cambiar su contenido. Cuando se trate de imágenes, hojas de estilo, scripts, etc., se estableció su caducidad a un mes, debido a que es el tiempo recomendado en la literatura, sin embargo no es una regla, ya que depende de cada propósito en particular. Para activar el control de caché del cliente, se han incorporado las siguientes directivas en el archivo de configuración del servidor Apache:

```
<ifmodule mod_expires.c>
ExpiresActive on
```

```
AddType image/x-icon .ico
<Filesmatch "\.(jpg|jpeg|png|gif|js|css|swf|ico)$">
ExpiresDefault "access plus 1 month"
</Filesmatch>
<Filesmatch "\.(pdf)$">
ExpiresDefault "access plus 1 year"
</Filesmatch>
</ifmodule>
```

Una posibilidad que se tiene con el servidor Apache es configurar su servicio de caché, de manera que todos los documentos requeridos sean almacenados para una rápida recuperación posterior. Esto si bien tiene importancia cuando se habilita un servidor Proxy, tiene una relevancia mayor cuando se utiliza la compresión de datos junto con la generación dinámica de contenidos.

Con un caché de servidor se optimiza la carga de este pues la compresión de los datos se hace una sola vez, evitando realizar la misma operación cada vez que se requiera el mismo archivo.

El servicio tiene la posibilidad de optimizar tanto la frecuencia de actualización del caché y considerar las directivas de control de este en los encabezados, así como la configuración relevante cuando tratamos con contenidos generados dinámicamente con una alta frecuencia de actualización.

Por omisión, todos los documentos almacenados en el caché del servidor tienen una fecha de expiración de una hora. Para activar su control, se han incorporado las siguientes directivas de configuración del servidor Apache que controlan la ubicación del caché y la distribución de los archivos temporales dentro de la estructura del disco:

Table 1. Resumen de configuraciones de control de caché y compresión de datos.

Parámetro	Tipo de documento	Valor propuesto	Descripción
mod_expires	PDF	Un año, ya que los documentos son trabajos de titulación y su contenido no cambia.	Controla la fecha de expiración de los documentos solicitados en las peticiones HTTP.
	JPG, JPEG, PNG, GIF, JS, CSS, ICO	Un mes, siguiendo lo recomendado por la literatura.	
mod_cache	Cualquiera	Una hora es el tiempo establecido por omisión y es el recomendado por la literatura.	Configura el servicio de caché en el servidor. Todos los documentos requeridos son almacenados para una rápida recuperación posterior.
mod_deflate	Cualquiera excepto formatos de archivo ya comprimidos, imágenes y PDF.	Habilitado en el servidor.	Comprime documentos antes de su envío al cliente.

<IfModule		mod_cache_disk.c>
CacheRoot		c:\xampp\tmp
CacheEnable	disk	/
CacheDirLevels		5

```
CacheDirLength 3
</IfModule>
</IfModule>
```

3.2 Compresión de datos

El servidor web Apache cuenta con un módulo que puede comprimir documentos antes de su envío al cliente. En cuanto al módulo Apache mod_deflate permite controlar el grado de compresión y su velocidad para no afectar el desempeño del servidor. Sin embargo, no todos los navegadores reconocen la compresión de datos en el intercambio de mensajes HTTP.

Para el caso de nuestro sistema, los documentos que se manejan son PDF y la compresión está dada por el mismo formato de archivo. Sin embargo con la siguiente configuración comprimimos todo excepto imágenes, archivos ya comprimidos y archivos PDF:

```
<IfModule mod_deflate.c>
SetOutputFilter DEFLATE
SetEnvIfNoCase Request_URI \.(?:gif|jpe?g|png)$ no-gzip dont-vary
SetEnvIfNoCase Request_URI \.(?:exe|t?gz|zip|bz2|sit|rar)$ no-gzip dont-vary
SetEnvIfNoCase Request_URI \.pdf$ no-gzip dont-vary
</IfModule>
```

Los parámetros antes mencionados de control de caché y compresión de datos se resumen en la Tabla1

3.3 Base de datos

Con respecto a la replicación de la base de datos se parte de la premisa de que se tiene dos servidores con MySql instalados. La dinámica posee la siguiente secuencia:

3.3.1 Servidor maestro

Primero en el servidor maestro se edita el archivo de configuración con las siguientes directivas y después se reinicia:

```
server-id=1
log-bin=mysql-bin
innodb_flush_log_at_trx_commit=1
sync_binlog=1
```

Posteriormente se crea un usuario en el servidor maestro que servirá para la replicación con el servidor esclavo:

```
GRANT REPLICATION SLAVE ON *.* TO 'slave_user_name'@'slave_ip'
IDENTIFIED BY 's3cret';
```

Con el objetivo de prevenir la escritura a la base de datos en el servidor maestro se ejecuta el siguiente comando:

```
FLUSH TABLES WITH READ LOCK;
```

También, se considera realizar un respaldo de la base de datos del servidor maestro, para luego restaurarlo en el servidor esclavo. Después se ejecuta el siguiente comando y se anota lo que muestre 'file' y 'position' que será usado en el servidor esclavo más tarde. Adicionalmente, en el servidor maestro se ejecuta el siguiente comando para desbloquear las tablas:

```
SHOW MASTER STATUS;
UNLOCK TABLES;
```

3.3.2 Servidor esclavo

Ahora se configura el servidor esclavo, editando el archivo de configuración:

```
server-id=2
```

Se ejecuta el siguiente comando en el servidor esclavo:

```
CHANGE MASTER TO
MASTER_HOST='<direccion_ip_master>',
MASTER_USER='<usuario_esclavo>',
MASTER_PASSWORD='<password>',
MASTER_PORT=3306,
MASTER_LOG_FILE='<file_obtenido_del_master>',
```

```
MASTER_LOG_POS=<position_obtenido_del_master>,
MASTER_CONNECT_RETRY=10;
```

En el servidor esclavo se restaura el respaldo de la base de datos original y se ejecuta el siguiente comando para iniciar el servicio y con esto finaliza la replicación de base de datos:

```
START SLAVE;
```

Por parte de las recomendaciones para la ejecución de sentencias SQL se pudo identificar que inicialmente, el sistema al momento de crear su índice de palabras generaba una gran cantidad de accesos a la base de datos debido a las sentencias INSERT utilizadas. Se obtiene una reducción significativa del tiempo si se genera la cadena de la sentencia INSERT por documento y se invoca su ejecución una sola vez.

En vez de generar una consulta INSERT por cada palabra por documento:

```
foreach ($palabrasClave as $key => $val){
mysql_query("insert into palabras (iddoc,palabra,frec) values('".$key."', '".$val."') ");
}
```

Se genera la cadena de la consulta completa por documento y ejecuta solo una vez:

```
$sqlInsert = "insert into palabras (iddoc,palabra,frec) values ";
foreach($palabrasClave as $key => $val){
$sqlInsert.= "('".$arch."','".$key."','".$val/$maxFrec."'),";
}
$sqlInsert = substr($sqlInsert,0,-1);
mysql_query($sqlInsert);
```

Cabe destacar que antes se realizaban 1,242,953 accesos a la base de datos y ahora se ejecutan 258, por lo tanto hay una reducción del 99.98% de accesos a la base de datos.

Algo similar sucede con las sentencias UPDATE. Cuando queremos optimizar el acceso a la base de datos tenemos que generar la cadena con la consulta y luego

acceder a la base de datos sólo una vez, en lugar de generar un acceso a la base de datos por cada sentencia generada:

```
foreach ($idf as $key = > $val){
mysql_query("update palabras set tfidf = '"$ndoctos/$val."' where palabra =
'"$key."'");
}
```

Se genera la cadena de la consulta y se ejecuta solo una vez:

```
$sqlUpdate = "update palabras set tfidf = CASE palabra ";
foreach ($idf as $key = > $val){
$sqlUpdate.="when '".$key."' then '".$ndoctos/$val."' ";
}
$sqlUpdate.= " end";
mysql_query($sqlUpdate);
```

4. Resultados Experimentales

Dentro del marco de los resultados experimentales se utilizó la herramienta YSlow (“Yahoo! YSlow,” 2015), la cual analiza una página Web y le otorga un grado según una colección de reglas definas por la empresa Yahoo! para mejorar su funcionamiento. Adicionalmente, YSlow otorga un grado de “A” hasta “F” a una página Web, siendo el mejor valor “A”, según el promedio del grado de cumplimiento de cada una de las reglas del perfil de validación. Estudios han demostrado que el tiempo de respuesta de las páginas web puede ser mejorado de 25% a 50% siguiendo estas reglas.

Yahoo! establece 34 reglas agrupadas en 7 categorías que constituyen una guía para agilizar la descarga y ejecución de una página Web. YSlow implementa las 22 reglas que pueden automatizarse para validar una página.

En la figura 4 (lado izquierdo) se muestra el análisis de la herramienta antes de aplicar las directivas al archivo de configuración de Apache con respecto al manejo

de caché y la compresión de datos. Como se puede observar la herramienta nos ofrece unas recomendaciones a seguir para mejorar el desempeño. En la misma figura 4 (lado derecho) se presenta el resultado del análisis de Yslow después de haber modificado el archivo de configuración de Apache con las directivas de manejo de caché y compresión de datos descritas en la secciones 3.1 y 3.2 respectivamente.

Grade A Overall performance score 94 Ruleset applied: YSlow(V2) URL: http://localhost/proyecto/

ALL (23) FILTER BY: CONTENT (6) | COOKIE (2) | CSS (6) | IMAGES (2) | JAVASCRIPT (4) | SERVER (6)

A	Make fewer HTTP requests
C	Use a Content Delivery Network (CDN)
A	Avoid empty src or href
E	Add Expires headers
B	Compress components with gzip
A	Put CSS at top
A	Put JavaScript at bottom
A	Avoid CSS expressions
n/a	Make JavaScript and CSS external
A	Reduce DNS lookups
A	Minify JavaScript and CSS
A	Avoid URL redirects
A	Remove duplicate JavaScript and CSS
A	Configure entity tags (ETags)
A	Make AJAX cacheable
A	Use GET for AJAX requests
A	Reduce the number of DOM elements
A	Avoid HTTP 404 (Not Found) error
A	Reduce cookie size
A	Use cookie-free domains

Grade A on Make fewer HTTP requests

Decreasing the number of components on a page reduce
reduce the number of components include: combine files,
and image maps.

»Read More

Copyright © 2015 Yahoo! Inc. All rights reserved.

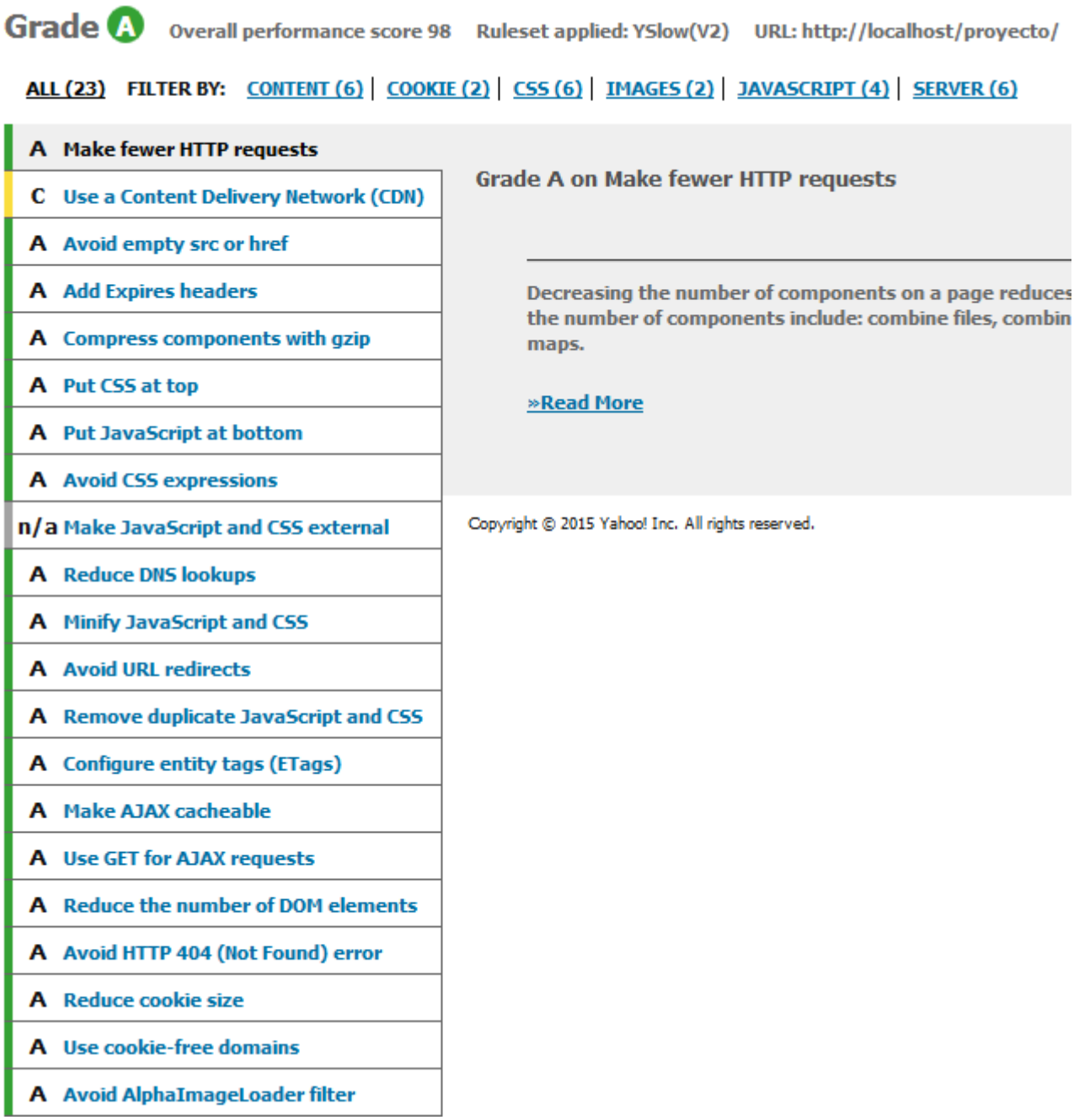


Figura 4. *Comparativa de resultados de la herramienta YSlow.*

Se puede observar en la figura 4 cómo las dos reglas que nos interesan, Add Expires headers (E) y Compress components with gzip (B), mejoraron su resultado (A) al aplicar las configuraciones mencionadas de compresión y control de caché; Al mismo tiempo el desempeño general pasó de 94 a 98.

El desempeño general es un porcentaje ponderado de las calificaciones individuales de cada regla. Las calificaciones de cada regla individual son calculadas de manera diferente dependiendo de la regla. Por ejemplo, para la regla 1, tres scripts externos son permitidos. Por cada script extra que se tenga se restan cuatro puntos de la calificación de esa regla, que tiene un peso por defecto de ocho.

La herramienta Yslow! permite además generar información con respecto al comportamiento de una página en términos de su tamaño y tiempos de trasferencia. La tabla 2 presenta una comparativa de tiempos de despliegue de información y de volumen de datos transferidos. En la primera columna (Normal) se puede observar el tiempo que tarda en cargar la página web con la configuración por defecto de Apache, mientras que en la segunda columna (Optimizado) muestra el tiempo que tarda en carga la misma página pero habiendo aplicado las directivas de control de caché y compresión de datos en el archivo de configuración de Apache.

Table 2. Resultado de las prueba de validación.

	Normal	**Optimizado**
Optimizado	173 ms	87 ms
Datos transferidos inicialmente	18.6 Kb	15.5 Kb
Datos transferidos posteriormente	4.8 Kb	1.7 Kb

A continuación se presentan gráficos de pastel que resaltan el beneficio de utilizar el caché y la compresión, en relación al volumen de datos que es transferido la primera vez que se accede a la página (18.6 Kb) sin compresión contra una visualización posterior de la misma página utilizando la configuración optimizada (Figura 5). En este último caso, el volumen de transferencia se reduce hasta ser de

1.7 Kb (reducción del 90.8%), que es una diferencia muy significativa en términos de percepción del usuario (la página se presentará más rápido).

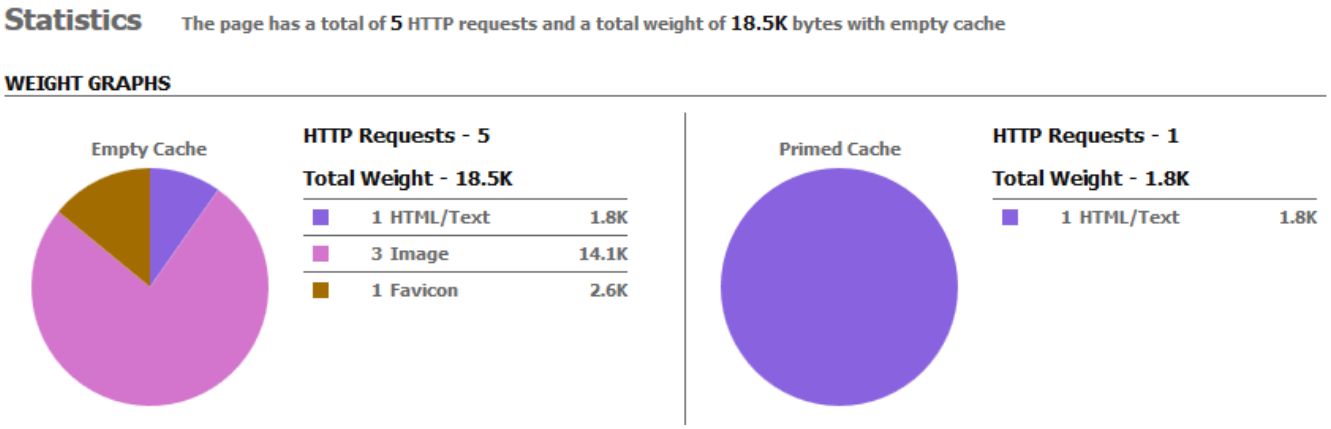


Figura 5. *Estadísticas de descarga de la página inicial aplicando las configuraciones (gráfico generado por Yslow).*

Para analizar el tiempo de descarga de la página se utilizó la herramienta FireBug que incorpora nuevas funcionalidades en un navegador Web, las cuales están orientadas al desarrollo y depuración de aplicaciones Web.

Se ha generado un reporte de duración (Figura 6) que brinda información sobre el tiempo que tarda el navegador Web en transferir y presentar los distintos recursos que conforman a la página Web. En total, el navegador ha tardado 87 ms (parte inferior) en desplegar la página, contra los 173 ms (parte superior) que si se visualizara la misma página sin considerar la compactación. Una mejora de poco más del 50%. Como se puede observar en la figura, las imágenes que ya son un formato comprimido no sufren modificación, sin embargo la página principal se reduce con la compresión de datos de 4.8 Kb a 1.8 Kb.

Cabe señalar que estos tiempos no son fijos, tienen variaciones que dependen del ancho de banda de la conexión y el grado de saturación de paquetes en la misma (no es la misma velocidad de acceso a una página en distinto horario de un mismo día). Sin embargo, la estadística generada da buena cuenta de los beneficios que aporta utilizar un esquema de optimización de aplicaciones Web.

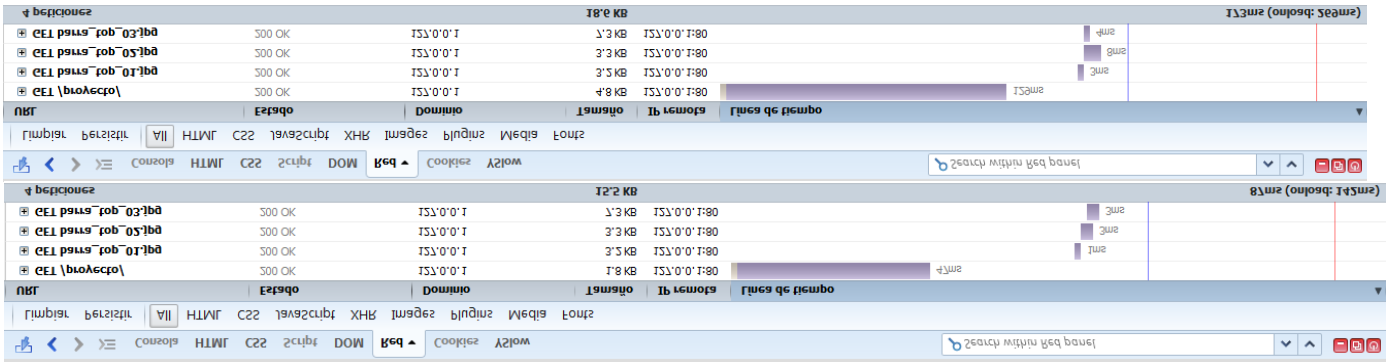


Figura 6. *Comparativa del tiempo de descarga, al utilizar compresión de datos, de la página principal analizada con Firebug.*

5. Conclusiones

En este trabajo se demuestra como combinando distintas configuraciones al servidor web Apache, como son el manejo de caché y la compresión de datos y siguiendo buenas prácticas al utilizar las sentencias SQL en nuestro manejador de base de datos, se obtiene una reducción significativa (poco más del 50%) en los tiempos de respuesta de nuestro sistema, así como también una reducción en el volumen de datos transferidos (reducción del 90.8%) entre el servidor y el cliente, lo cual tiene una repercusión inmediata en la percepción del usuario ya que la página se despliega más rápido.

Por otro lado, se demuestra que para los casos donde la interacción del sistema con la base datos es primordial y requiere un gran uso de ella, como lo es la creación de índices para un sistema de recuperación de información, y siguiendo las recomendaciones en la construcción y ejecución de sentencias INSERT, se logra una reducción significativa (reducción del 99.98%) en los accesos a la base de datos lo que repercute directamente en una reducción en el tiempo de procesamiento.

Si bien se analizaron los resultados para nuestro caso de estudio, se puede afirmar que los mismos beneficios serán obtenidos siempre y cuando la plataforma bajo la que funcione cualquier otro sistema sea la misma, es decir, Apache y MySql, ya que las directivas de control de caché y compresión de datos se pueden

configurar para otros tipos de archivos, sin embargo, lo determinante es seleccionar la configuración correcta de los parámetros del servidor web Apache.

Referencias

Ahuja, S., Wu, T. W. T., & Dixit, S. (2003). On the effects of content compression on Web cache performance. Proceedings ITCC 2003. International Conference on Information Technology: Coding and Computing. http://doi.org/10.1109/ITCC.2003.1197573

Ahuja, S., Wu, T. W. T., & Dixit, S. (2003). On the effects of content compression on Web cache performance. Proceedings ITCC 2003. International Conference on Information Technology: Coding and Computing. http://doi.org/10.1109/ITCC.2003.1197573

Apache HTTP Server Reference Manual – for Apache version 2.2.17. (2010). EUA: Network Theory Ltd.

Baeza-Yates, R., & Ribeiro-Neto, B. (1999). Modern Information Retrieval. England: Addison-Wesley Longman Publishing Co., Inc.

Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. Knowledge-Based Systems, 46, 109–132. http://doi.org/10.1016/j.knosys.2013.03.012

Gandhi, N., Tilbury, D. M., Diao, Y., Hellerstein, J., & Parekh, S. (2002). MIMO control of an apache web server: Modeling and controller design. Proceedings of the American Control Conference, 6, 4922–4927. http://doi.org/10.1109/ACC.2002.1025440

Jun Li, & Menghan Lu. (2013). The performance optimization and modeling analysis based on the Apache Web Server, 1712–1716.

Karin Verspoor, K. B. C. (2013). Natural Language Processing. In Encyclopedia of Systems Biology (pp. 1495–1498). Springer New York. http://doi.org/10.1007/978-1-4419-9863-7_158

Kemme, B., Jimenez-Peris, R., & Patino-Martinez, M. (2010). Database Replication. Morgan & Claypool. http://doi.org/10.2200/S00296ED1V01Y201008DTM007

Manning, C., Raghavan, P., & Schütze, H. (2009). An Introduction to Information Retrieval. Cambridge, England: Cambridge University Press.

Manning, C., & Schutze, H. (1999). Chapter 11. Probabilistic Context Free Grammars. Foundations of Statistical Natural Language Processing, 381–404.

Menéndez, V. H., Castellanos, E., Aguilar, R. A., & Gómez, S. (2013). Optimización del Desempeño de una Plataforma e-Learning mediante Técnicas no Invasivas. Congreso Internacional de Investigación Academia Journals, 5, 2188 –2193.

Myalapalli, V. K., & Savarapu, P. R. (2014). High Performance SQL Finesse for Lucrative Programming. Annual IEEE India Conference (INDICON).

MySQL 5.7 Reference Manual. (2015). Retrieved from http://dev.mysql.com/doc/refman/5.7/en/

Smith, P. (2013). Professional Website Performance: Optimizing the End and the Back End. Indianapolis, Indiana: John Wiley & Sons, Inc.

Souders, S. (2007). High Performance Web Sites: Essential Knowledge for Frontend Engineers. EUA: O'Reilly Media, Inc.

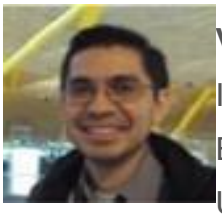
Vences, R., Menéndez, V. H., & Zapata, A. (2015). Sistema de Recomendación de Trabajos de Titulación de la Facultad de Ciencias Antropológicas de la Universidad Autónoma de Yucatán. In R. Juarez, G. César, H. Jadwiga, G. Ibargüengoitia, C. Fernández, G. Licea, & S. Vázquez (Eds.), Tendencias en Investigación y Aplicaciones Prácticas de la Ingeniería de Software (pp. 3–8). San Luis Potosí, México.

W3C. (2010). Web Accessibility Initiative (WAI). Retrieved from http://www.w3.org/WAI/

Yahoo! YSlow. (2015). Retrieved from http://developer.yahoo.com/yslow/

Notas biográficas:

Rodrigo Vences Nava es Licenciado en Ciencias de la Computación por la Universidad Autónoma de Yucatán, México. Actualmente se encuentra cursando la Maestría en Ciencias de la Computación de la Universidad Autónoma de Yucatán, México. Es Administrador de Tecnologías de Información de la Facultad de Ciencias Antropológicas de la Universidad Autónoma de Yucatán, México. Su trabajo de investigación se centra en temas relacionados con repositorios de tesis electrónicas, sistemas de recomendación y minería de datos.



Víctor Hugo Menéndez Domínguez es Doctor en Tecnologías Informáticas Avanzadas por la Universidad de Castilla-La Mancha, España. Es Profesor Titular en la Facultad de Matemáticas de la Universidad Autónoma de Yucatán, México. Su trabajo de investigación se centra en temas relacionados con repositorios de documentos digitales, la representación del conocimiento y la gestión de Objetos de Aprendizaje.

Alfredo Zapata González es Doctor en Tecnologías Informáticas Avanzadas por la Universidad de Castilla-La Mancha, España. Es Profesor Titular en la Facultad de Educación de la Universidad Autónoma de Yucatán, México. Su trabajo de investigación se centra en temas relacionados con Minería de datos y Sistemas de Recomendación para la Educación a distancia.

Esta obra está bajo una licencia de Creative Commons
Atribución-NoComercial-Compartirlgual 4.0 Internacional.