Universidad Central "Marta Abreu" de Las Villas

Facultad de Ingeniería Eléctrica

Departamento de Telecomunicaciones y Electrónica

TRABAJO DE DIPLOMA

Algoritmo eficiente para el manejo de paquetes en servidores proxy

Autor: **Chi Le Van**

Tutor: **Dr. Héctor Cruz Enríquez**

Santa Clara 2011

"Año 53 de la Revolución"

Universidad Central “Marta Abreu” de Las Villas

Facultad de Ingeniería Eléctrica

Departamento de Telecomunicaciones y Electrónica

TRABAJO DE DIPLOMA

Algoritmo eficiente para el manejo de paquetes en servidores proxy

Autor: **Chi Le Van**

E-mail: cle@uclv.edu.cu

Tutor: **Dr. Héctor Cruz Enríquez**

E-mail: hcruz@uclv.edu.cu

Santa Clara 2011

"Año 53 de la Revolución"

Hago constar que el presente trabajo de diploma fue realizado en la Universidad Central "Marta Abreu" de Las Villas como parte de la culminación de estudios de la especialidad de Ingeniería en Automática, autorizando a que el mismo sea utilizado por la Institución, para los fines que estime conveniente, tanto de forma parcial como total y que además no podrá ser presentado en eventos, ni publicados sin autorización de la Universidad.

_____

Firma del Autor

Los abajo firmantes certificamos que el presente trabajo ha sido realizado según acuerdo de la dirección de nuestro centro y el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

_____ _____

Firma del Autor

Firma del Jefe de Departamento donde se defiende el trabajo

_____

Firma del Responsable de Información Científico-Técnica

PENSAMIENTO

"El conocimiento es la fuente de toda riqueza.".

José Martí

DEDICATORIA

A mi familia y a mi novia que siempre han sido las personas más maravillosas de toda mi vida.

AGRADECIMIENTOS

A mis padres por los sabios consejos que he recibido de ustedes, por no haber dejado de luchar diariamente para encaminarme en la vida, por no descansar para verme cada día convertirme en una mejor persona y por el amor que me han brindado los quiero mucho.

A toda mi familia…

A mi tutor Dr. Héctor Cruz Enríquez por su apoyo incondicional, por su profesionalidad, que me ha sugerido las ideas para realizar esta tarea.

A mi novia Tieu por acompañarme, ayudarme tanto en los momentos bonitos y así como los difíciles.

A mis amigos Carlos de La Guardia, Henry Moreno Diaz que siempre me han ayudado en la realización de este trabajo.

A todos los profesores por haberme brindado todos los conocimientos preciosos de mi carrera.

A todos mis compañeros de grupo, especialmente a Carlos de La Guardia, Denis fueron los mejores cinco años los que pase junto a ustedes.

A todos los que de una manera u otra han hecho posible la realización de este trabajo.

Muchas gracias.

TAREA TÉCNICA

- Realizar estudio bibliográfico sobre el empleo de caché para servicios Web y sus políticas de reemplazo.
- Desarrollar un modelo de teletráfico que caracterice el acceso de Internet a través de servidores proxy.
- Analizar el rendimiento de servidores proxy con la herramienta WebTraff.
- Confeccionar el informe final.

Firma del Autor

Firma del Tutor

RESUMEN

En la actualidad los términos relacionados con la World Wide Web, su popularidad, cantidad de información disponible y el empleo de los servicios Web en sentido general, han experimentado un crecimiento exponencial. Con el fin de reducir la sobrecarga asociada a las frecuentes peticiones de documentos reiterados por los usuarios locales, se han desarrollado esquemas de almacenamiento en caché tanto del lado del cliente como del lado del servidor proxy o intermediario, algo que ha sido ampliamente utilizado en la actualidad. Sin embargo, la cantidad de memoria disponible en los servidores locales para el almacenamiento de esta información es realmente limitada; por lo tanto hay que decidir cuáles objetos de la Web se almacenan y cuáles no. Este hecho ha dado origen a varios métodos o políticas de reemplazo; algunos de ellos son abordados en el presente trabajo. La mayoría de las implementaciones de las políticas tradicionales se basan en mecanismos de "uso reciente" (LRU, del inglés *Least Recently Used*). Sin embargo, debido a la heterogeneidad de las demandas de tráfico en la Web, tanto en el tamaño de los documentos como en los retardos de la transferencia, el uso por defecto de una política de reemplazo no resulta eficiente. De acuerdo a la evaluación de las políticas de reemplazos y al empleo de diversas métricas de calidad en el servidor proxy de caché (Request Hit Rate, Bye Hit Rate y Latency Ratio) se propone un algoritmo que pretende mejorar el mecanismo de remplazo de contenidos en la caché del proxy haciendo más eficiente el manejo de paquetes.

TABLA DE CONTENIDOS

INTRODUCCIÓN

Con la creciente popularidad de la World Wide Web, el tráfico Web se ha convertido en una de las aplicaciones que más consumen recursos en Internet. La búsqueda de soluciones para mejorar la utilización del ancho de banda y el tiempo de respuesta del tráfico de Internet es actualmente un problema importante. Diferentes métodos están siendo investigados en este ámbito, y una de las alternativas más utilizadas para reducir de forma directa el tráfico en Internet es el almacenamiento en la caché del proxy. El método de caché en el proxy utiliza un único host para atender a varios usuarios (de uno a varios cientos de clientes). Cada cliente envía las solicitudes a través del proxy quien mantiene copias de los documentos en su caché. Una cuestión crucial en el almacenamiento en caché es decidir qué documentos se deben mantener en la memoria de caché. En otras palabras, cuando un nuevo documento se presenta en la caché y esta se encuentra llena (o muy cargada), ¿Cuál documento debe ser removido de la caché? La estrategia para tomar tales decisiones se conoce como política de reemplazo en caché.

Problema:

¿Cómo cambia el tiempo de respuesta con diferentes niveles de tráfico?

De manera que otra de las interrogantes sería:

¿Cómo se evalúa el tiempo de respuesta con la implementación de una política de reemplazo?

¿Qué tipo de distribución sigue el tiempo de respuesta?

¿El almacenamiento en la caché del proxy mejorá el tiempo de respuesta?

¿Cuál será el efecto del tamaño de la caché y las políticas de reemplazo en el tiempo de respuesta?

Objeto de estudio:

Proponer la implementación de un algoritmo eficiente para la optimización del proceso de remplazo en la caché del proxy.

Objetivo general:

Buscar una política de reemplazo de caché más eficiente para mejorar el tiempo de respuesta y el uso del ancho de banda con el empleo de servidores proxy.

Objetivos específicos:

- Incrementar los conocimientos teóricos sobre los algoritmos de reemplazo en caché, tamaño de caché y las métricas que repercuten en el tiempo de respuesta.
- Desarrollar el modelo o esquema de simulación así como la implementación de herramientas para la evaluación del tiempo de respuesta en servidores proxy.
- Formular propuestas de medidas de calidad para alcanzar un funcionamiento más eficiente en servidores proxy.

Organización del informe

En el primer capítulo se hace un análisis bibliográfico de conceptos y teorías imprescindibles para el cumplimiento de los objetivos de este trabajo; sobre todo haciendo énfasis en los algoritmos para el reemplazo en caché, los modelos de teletráficos que caracterizan el acceso a internet a través de servidores proxy y las métricas que repercuten en el tiempo de respuesta.

En el segundo capítulo se abordan analíticamente algunos desarrollos que conducen a la propuesta de un algoritmo para el remplazo de contenidos en la cache de los servidores proxy.

En el tercer capítulo se expresan los resultados experimentales que ilustran la efectividad del algoritmo propuesto..

Finalmente se brindan conclusiones y recomendaciones para el trabajo futuro.

CAPÍTULO 1. ESTADO DEL ARTE SOBRE EL USO DE CACHÉ PARA SERVICIOS WEB

En este capítulo se aborda lo relacionado a los servidores proxy, el empleo de caché para servicios Web y las distintas políticas de reemplazo que posteriormente serán evaluadas en este estudio. También se realiza una introducción sobre los factores de rendimiento en un servidor proxy como el tiempo de respuesta, la razón de aciertos (HR – Hit Rate) y la razón de aciertos de bytes (BHR – Byte Hit Rate).

1.1 Introducción al concepto de World Wide Web

Hasta la década de los 90 los principales usuarios de Internet eran académicos, empleados gubernamentales, investigadores e industriales (Tanenbaum, 1996); y las principales aplicaciones utilizadas eran el correo electrónico, noticias, acceso remoto y transferencia de archivos.

Un nuevo servicio o aplicación conocida como WWW (del inglés *World Wide Web*), cambió el concepto de la Internet original, permitiendo a millones de usuarios no académicos navegar por Internet. Concebido originalmente por el físico Tim Berners-Lee perteneciente al CERN (del inglés *European Centre for Nuclear Research*), la WWW ha sido ilustrada como un espacio de hipertexto global (Berners-Lee, 1990) en la que cualquier información accesible desde la red puede ser referida con solo localizador (del inglés *Universal Document Identifier*) más tarde URI (del inglés *Universal Resource Identifier*) (Fielding et al., 1998). El diseño de la WWW se basa en algunos criterios: un sistema de información debe ser capaz de registrar las asociaciones entre objetos arbitrarios; los usuarios no deben ser obligados a utilizar un lenguaje en particular o sistema operativo; la

información debe estar disponible en todas las plataformas, incluyendo las futuras; la información dentro de una organización debe estar representada con precisión y la entrada o la corrección debe ser trivial. En términos más generales, la información debe ser accesible de manera transparente (Mahanti, 1999).

La arquitectura de la Web se basa en un modelo cliente-servidor (Arlitt y Williamson, 1997b) medianteel acceso del cliente a la información localizada en un servidor remoto haciendo uso de un software o navegador Web. Algunos de estos navegadores se encuentran disponibles y abarcan desde solo texto hasta interfaces gráficas de usuario GUI (del inglés *Graphical User Interface*). Software como Mozilla Firefox, Netscape Communicator y Microsoft Internet Explorer constituyen un ejemplo.

Los navegadores Web se comunican con los servidores Web utilizando el protocolo HTTP (del inglés *Hyper Text Transfer Protocol*) (Tanenbaum, 1996). El protocolo HTTP se basa en el protocolo TCP /IP. TCP es un protocolo fiable orientado a la conexión que permite la entrega de un flujo de bytes de un equipo a otro dentro de una Intranet o en Internet (Tanenbaum, 1996).

Existen dos formas diferentes de solicitar una página Web (por ejemplo: archivos de texto, audio, vídeo, imágenes, etc) de un servidor Web: conexión directa desde el cliente al servidor o conexión indirecta a través de un servidor proxy para servicios Web (en lo adelante servidor proxy Web).

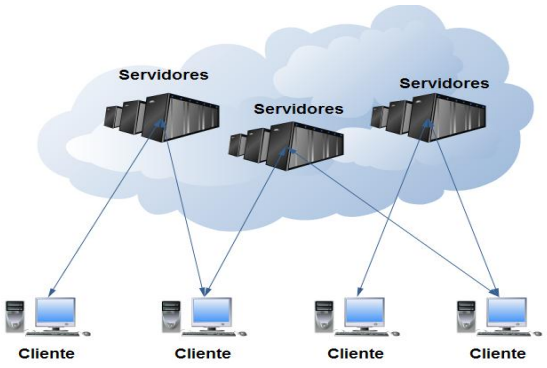


Figura 1.1: Modelo de conexión directa.

Los dos casos se ilustran en la figura 1.1 y figura 1.2. En el primer caso (véase figura 1.1), un cliente establece una conexión Web (HTTP) directamente con el servidor. El servidor responde con la página solicitada, que es visualizada por el navegador.

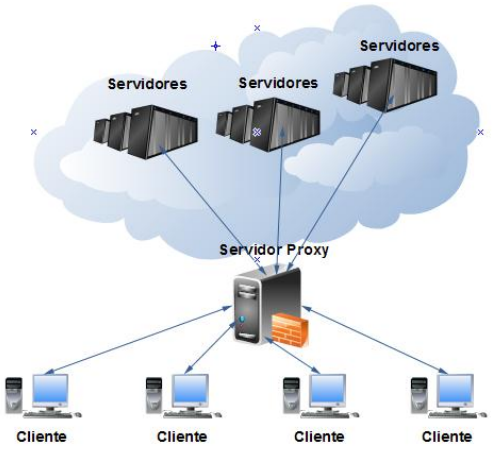


Figura 1.2: Modelo de conexión indirecta a través de un servidor proxy.

En otro caso (véase figura 1.2), la solicitud del cliente se envía a un servidor proxy, que remite la solicitud al servidor (si es necesario), y cuando la página solicitada es recibida, se envía al usuario que la visualizará mediante su navegador Web. El servidor proxy también puede ser configurado para actuar como un proxy caché de modo que los documentos solicitados se almacenan en una caché local para satisfacer las demandas futuras.

1.2 Concepto de Cliente de Web

La WWW es un repositorio de documentos de diferentes tipos que por lo general se refiere a páginas y sitios Web. Cada página puede contener enlaces a otras páginas en cualquier parte del mundo (relacionados o no). Estas páginas que apuntan a otras páginas utilizan el modo de hipertexto.

Los navegadores Web son programas que permiten visualizar las páginas deseadas. Para mostrar una página Web, un navegador establece una conexión con un servidor Web y solicita una página Web. Cuando el navegador recibe la página, este interpreta los comandos de texto y el formato que figura en esta y lo muestra en el formato correcto de la pantalla. Con un navegador basado en GUI, los enlaces a otras páginas dentro de una página son subrayados y/o aparecen en un color diferente. Estos enlaces se conocen como hipervínculos (del inglés *Hyperlinks*).

Una página Web también puede apuntar a otras páginas Web a través de un icono, fotografía, entre otras. Algunas páginas pueden contener elementos no visibles, como clips

de vídeo, clips de audio, etc. El término hipermedia se utiliza para referirse a las páginas cuyo contenido es una mezcla de texto, imágenes, audio, video, y cualquier otro tipo de medios de comunicación. Algunas páginas Web pueden contener elementos como imágenes que son muy grandes, por lo que se necesita tiempo para que el navegador complete toda la descarga.

Algunos navegadores utilizan una caché de memoria en el disco del cliente para almacenar las páginas recientemente solicitadas en caso de que el usuario decida visitarlas nuevamente.

1.3 Concepto de Servidor Web

Un servidor Web es un proceso de software que se ejecuta en la máquina servidor; el proceso utiliza el puerto TCP 80, comúnmente utilizado para las solicitudes HTTP entrantes desde un cliente. Un navegador normalmente realiza una solicitud de una página Web mediante el establecimiento de una conexión a un host servidor a través del puerto 80. Luego de establecida la conexión; el navegador envía una solicitud GET para conseguir una página. El proceso del servidor responde con un mensaje de respuesta HTTP que tiene el formato siguiente (Mahanti, 1999; Fielding et al., 1998; Mahanti y Williamson, 1999):

Response = *Status-Line*

[Header]

CRLF

[message-body]

Status-Line: Consiste en la versión del protocolo HTTP seguido de un código de estado numérico y su descripción asociada:

Status-Line = HTTP-version Status-Code Reason-Phrase

Status-Code: Es uno de los factores más importantes a la hora de determinar si un documento ha de ser guardado en Caché o no. Es el código de respuesta o estado del servidor HTTP. El código de estado, de tres dígitos, indica si la petición tuvo éxito o si por el contrario, ocurrió algún tipo de error. Los códigos de estado se dividen en 5 grupos:

Código	Descripción del Grupo
1xx	Un estado intermedio, de información. Indica que la transacción está siendo procesada.
2xx	La petición fue recibida y procesada con éxito.
3xx	El servidor está redireccionando al usuario a una ubicación diferente.
4xx	Existe un error o problema con la petición del usuario, como por ejemplo que se requiere autenticación o que el documento pedido no existe.
5xx	Se produjo un error en el servidor para una petición válida.

1.4 Concepto de Servidor Proxy

Un servidor proxy WWW proporciona acceso a la Web para usuarios en subredes privadas sólo pueden acceder a Internet a través de un determinado servidor o cortafuegos (como se muestra en la figura 1.3). Incluso usuarios que no dispongan de servidores de DNS (del inglés *Domain Name Service*) pueden acceder a Internet a través del proxy conociendo su dirección IP (del inglés *Internet Protocol*) (Reddy et al., 2011).

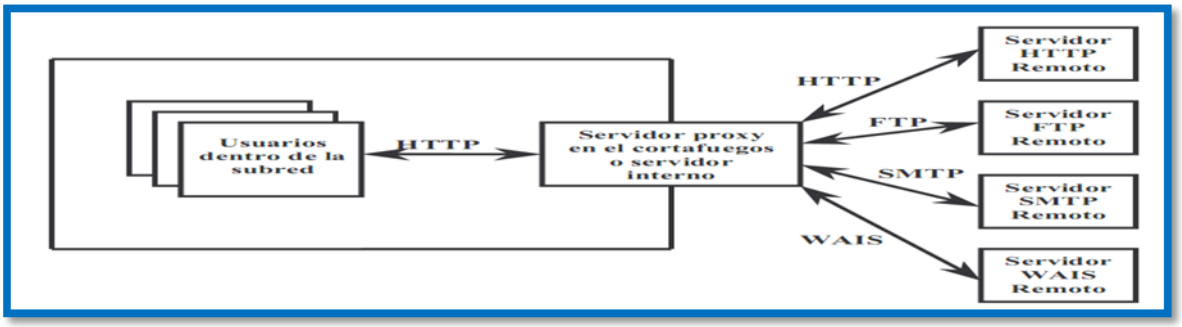


Figura 1.3: Ejemplo de acceso a Internet a través de servidor proxy.

El servidor proxy espera el arribo de una petición procedente de un usuario desde la red local. Una vez que se produce, si el servidor proxy no dispone del documento solicitado, o si del que dispone ha expirado, la dirige la solicitud a un servidor remoto fuera de la Intranet. Cuando este recibe la respuesta del exterior, la manda de vuelta al usuario y la

almacena en su memoria local. Actúa por tanto, como intermediario entre el usuario y otro servidor de información al que se accede y puede ser usado por todos los usuarios de una Intranet. Los usuarios no pierden funcionalidad haciendo uso del proxy excepto si es necesario hacer algún procesamiento especial.

El servidor proxy permite el almacenamiento en caché, de forma local que las páginas consultadas con mayor frecuencias tengan un acceso local y así, ante una petición de un usuario, si el documento solicitado está presente en caché y no ha alcanzado aún su tiempo de expiración (es decir, aún se considera como *´fresco´*), se puede descargar directamente de la caché. La diferencia entre ambos casos se encuentra en que si no ha expirado, se considera que el documento sigue siendo válido, mientras que si lo ha hecho, pero el servidor ha confirmado que aún no ha sido modificado también se puede seguir usando el documento y no es necesario volver a descargarlo desde el servidor remoto original.

De este modo se obtiene un notable incremento en la velocidad de transferencia de la información con el consiguiente uso eficiente del ancho de banda y un menor tiempo de respuesta, además de permitir un ahorro en espacio de disco puesto que una sola copia es almacenada y usada por múltiples usuarios (Ali et al., 2011; Patil y Pawar, 2011). Incluso puede permitir mostrar páginas Web aún cuando la red exterior no esté disponible, proporcionándolas de su propia caché. El servidor proxy puede actuar como filtro de contenidos, como traductor de formato de archivos, adaptador de protocolos (sin pérdida de funcionalidad) o para verificar la seguridad (virus, accesos permitidos, etc.) incrementando la seguridad de la red interna y pudiendo ser tan efectivo como un cortafuego.

A la hora de hacer transacciones con el servidor proxy, el cliente siempre usa HTTP aún cuando el recurso pedido se encuentre en un servidor remoto que use un protocolo distinto, como por ejemplo FTP (del inglés *File Transfer Protocol*), como se muestra en la figura 1.4. A la hora de hacer la petición al servidor proxy, se especificará la URL (del inglés *Uniform Resource Locator*) completa y no sólo el nombre de la ruta u otras claves de búsqueda adicionales como ocurre en HTTP.

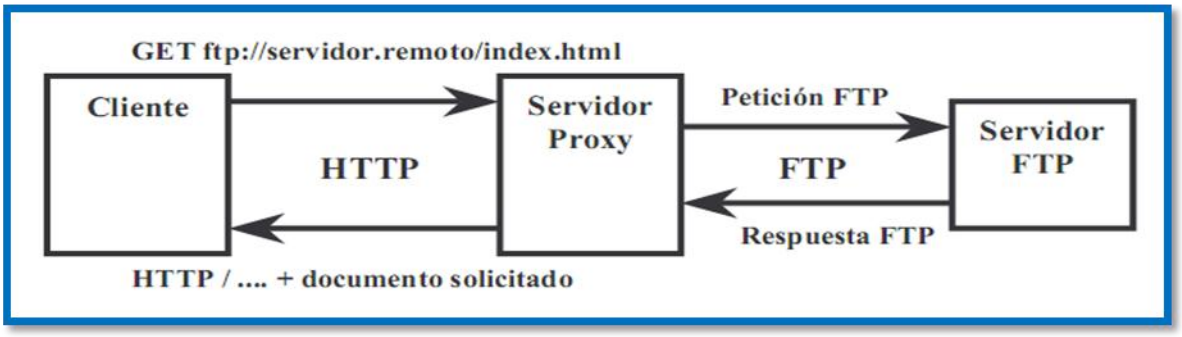


Figura 1.4: Ejemplo de transacción a través de proxy.

Por tanto, el servidor proxy debe realizar funciones tanto de servidor como de cliente. Actúa como servidor cuando acepta las peticiones HTTP de los usuarios conectados a él y como cliente cuando cursa esas peticiones hacia los servidores remotos para poder obtener de ellos los documentos solicitados por los usuarios.

1.5 Concepto de Caché

La caché es otro aspecto importante y sobre el que ya se ha hecho alguna reseña anteriormente. Se encarga del almacenamiento de los documentos más frecuentemente consultados por los usuarios, como se muestra en la Figura 1.5. El uso de una caché sólo es beneficioso cuando el coste de almacenamiento de un documento sea menor al que supondría traer dicho documento directamente de un servidor remoto.

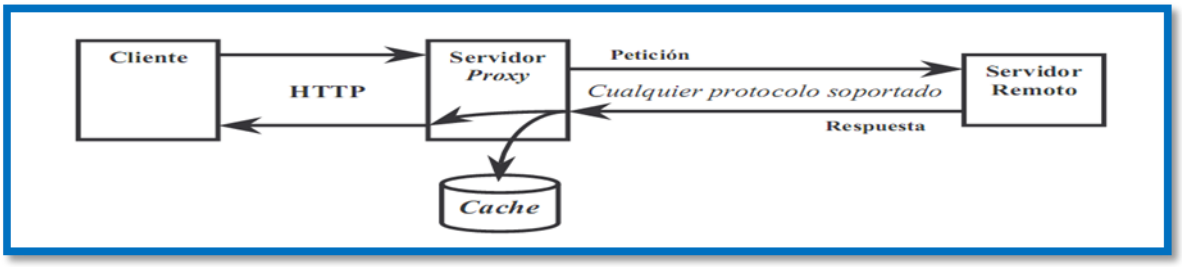


Figura 1.5: Ejemplo de uso de caché en el proxy.

El concepto de caché es aplicable a casi todos los aspectos de la computación y los sistemas de Red. Los procesadores de las computadoras tienen caché tanto de datos como de instrucciones mientras que los sistemas operativos tienen caché para los manejadores de discos y archivos de sistemas.

El buen funcionamiento de la caché radica en el principio de localidad que siguen las referencias. Existen dos tipos de localidad, temporal y espacial. La localidad temporal hace referencia a la popularidad de los documentos (en un período de tiempo existe más probabilidad de que se soliciten ciertos documentos que otros), mientras que la localidad espacial tiene que ver con el hecho de que ciertos documentos tienen la tendencia de ser pedidos conjuntamente con otros, como por ejemplo las imágenes que pueda haber en una determinada página Web. Así, al solicitarse una página Web, también existe una elevada probabilidad de que se soliciten las imágenes que contiene la misma.

Las cachés usan una localidad de referencia para predecir accesos futuros basados en otros previos y en el caso de que la predicción sea correcta se producirá un incremento notable del rendimiento. Cuando un documento solicitado se encuentra en caché, se conoce a esto como un acierto en caché mientras que si no está se producirá un fallo en caché. La mayoría de las tareas de procesamiento de datos exhiben localidad de referencia y por lo tanto se benefician del almacenamiento en caché.

El uso de las cachés se justifica en base a tres beneficios principales como son la reducción en la latencia, el uso más eficiente del ancho de banda y la disminución de la carga de tráfico soportada por los servidores remotos, donde se encuentran originalmente los documentos a los que se desea acceder.

La latencia, básicamente, está referida a los retardos que sufre la transmisión de la información de un punto a otro. La transmisión de información sobre circuitos eléctricos y ópticos está limitada por la velocidad de la luz. De hecho, los pulsos eléctricos y ópticos viajan aproximadamente a dos tercios de la velocidad de la luz en cables y fibras, lo que genera un retardo que se incrementa con la distancia y que en conexiones transoceánicas es bastante considerable.

Por otro lado, si en las comunicaciones se emplean enlaces que están próximos a su límite máximo de uso, los documentos sufren largos retardos en las colas de los enrutadores y conmutadores. Esto puede ocurrir en un elevado número de puntos a lo largo del camino seguido por la información, lo que puede ocasionar grandes retardos e incluso la pérdida de información en el caso de que sea descartada de la cola de un encaminador si dicha cola se llena.

Una caché situada cerca de sus usuarios reduce la latencia para los aciertos en caché. Los retardos de transmisión que se producen son mucho menores porque los sistemas se encuentran próximos. Además, los retardos originados por retransmisión o por la introducción de los paquetes en colas son menos probables pues en la transmisión se ven involucrados menos enrutadores y enlaces.

Por otro lado, un fallo en caché no debe sufrir un retardo mucho mayor del que supondría una transferencia directa entre el usuario y el servidor original. Por lo tanto, los aciertos en caché reducen la latencia media de todas las peticiones.

Otro de los beneficios principales del uso de cachés es la reducción en el ancho de banda utilizado, puesto que cada petición que supone un acierto en caché permite un ahorro de este. Si una conexión de Internet está congestionada, instalando una caché se mejora el rendimiento de otras aplicaciones de red, ya que todas compiten por el mismo ancho de banda. Así, con la caché se consigue reducir el ancho de banda consumido por tráfico HTTP dejándolo libre para otras aplicaciones que también lo requieran.

El tercer beneficio principal del uso de caché es la reducción en la carga de tráfico soportado por los servidores origen de los documentos. El tiempo de respuesta de un servidor aumenta conforme al incremento de la tasa de peticiones sobre el mismo, mientras que por el contrario, un servidor libre de peticiones es más rápido respondiendo a la llegada de alguna solicitud. De esta forma, si haciendo uso de la caché se logra una disminución de la carga de trabajo, se está favoreciendo a una mayor efectividad en las conexiones con los servidores remotos.

Por todo ello, se elige la alternativa del uso de caché para conseguir que los usuarios perciban un menor retardo en sus peticiones de documentos, los gestores de las redes un menor tráfico y los servidores Web una menor tasa de peticiones y todo ello gracias a las copias de los documentos más populares en la caché del servidor proxy.

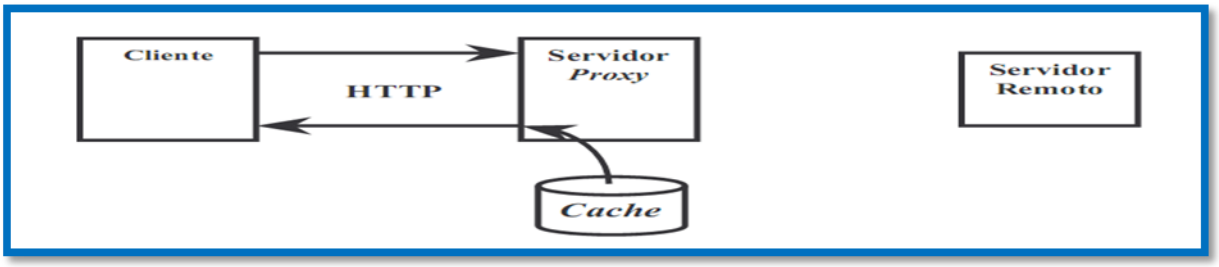


Figura 1.6: Ejemplo de acierto en caché.

La caché del proxy suele actuar como una caché de segundo nivel ya que recibe sólo los fallos en caché procedentes de los usuarios Web que ya usan una caché a nivel de usuario. De los aciertos en caché de usuario tan sólo se reciben peticiones para comprobar la frescura del documento existente en la caché del usuario y en caso de comprobarse la validez del mismo, permite un ahorro en ancho de banda para ambas cachés simultáneamente.

1.6 Cargas de Trabajo

Un factor importante para entender el comportamiento del tráfico en la Web es el análisis de las cargas de trabajo que reciben o generan una cierta cantidad de clientes, proxy con caché y servidores Web.

1.6.1 Cargas de trabajo en clientes

Uno de los primeros estudios hechos para entender los patrones de acceso de los clientes en la Web fue realizado en (Carlos et al., 1995). Su análisis estadístico detallado mostró que varias características en el uso de los clientes en la Web pueden ser modeladas usando distribuciones que siguen una ley potencial (ejemplo: el aspecto de la distribución es una hipérbola, Zipf, Pareto, Richter). Su función de densidad de probabilidad correspondiente es:

$$p(x) = \alpha k^{\alpha} x^{-\alpha-1}$$

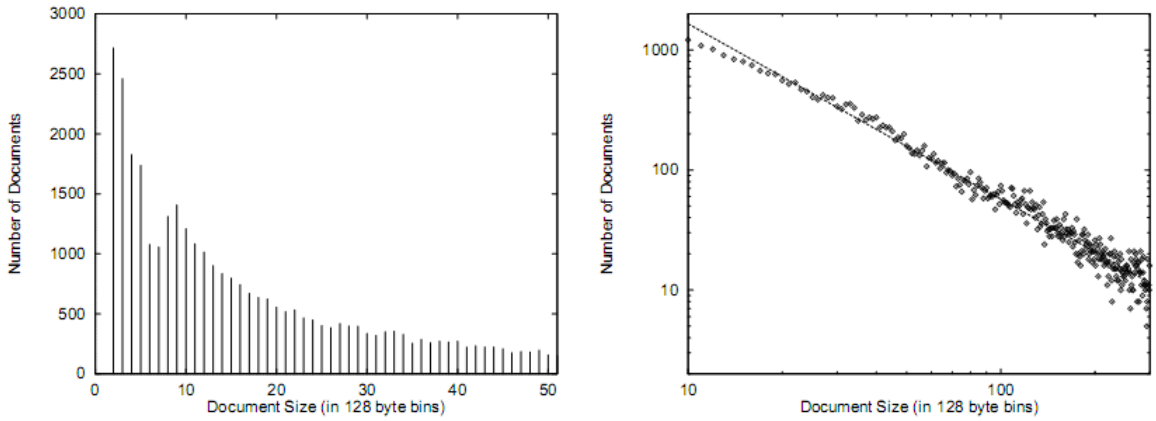


Figura 1.7 Distribución del tamaño de los documentos

La función de distribución de probabilidad correspondiente es entonces:

$$F_{par}[X \leq x] = 1 - (\frac{k}{x})^{\alpha}, \quad \alpha, k \geq 0, x \geq k.$$

En (Carlos et al., 1995) se muestra que elementos como la distribución de tamaños de documentos, la popularidad de documentos como una función del tamaño, y la frecuencia de referencias a documentos como una función de su clasificación (posición en una lista), siguen una distribución de ley potencial. En otro artículo (Barford et al., 1999) se comparan algunas trazas más recientes de clientes en la Web con otros estudios realizados anteriormente con el fin de comprender cómo las cargas de trabajo cambian con el tiempo. El análisis estadístico de los nuevos datos muestra que la mayoría de las características de comportamiento señaladas en los trabajos anteriores todavía pueden ser modeladas utilizados leyes de potencialidad.

1.6.2 Cargas de Trabajo en Servidores

Varios estudios para caracterizar las cargas de trabajo en servidores Web se han elaborado en los últimos años (Braun y Claffy, 1995; Mogul, 1995; Arlitt y Williamson, 1997a). En un estudio exhaustivo en (Arlitt y Williamson, 1997a) se identifican diez características comunes para seis diferentes cargas de trabajo cuyas duraciones comprendían entre una semana y un año. Las características identificadas son:

(a) Aproximadamente el 90% de todas las peticiones resultaron transferencias de documentos exitosas.

(b) Los archivos HTML y de imágenes representan más del 90% de las peticiones totales (información encontrada también en (Braun y Claffy, 1995) para el servidor de la NCSA, y en (Carlos et al., 1995) para sus trazas en los clientes).

(c) Las solicitudes diferentes representan menos del 3% del total de peticiones vistas por el servidor.

(d) El tamaño de transferencia promedio es menor que 21KB (kilo bytes).

(e) Aproximadamente una tercera parte de los documentos son accedidos sólo una vez.

(f) La distribución del tamaño de los archivos es una cola pesada (del inglés heavy-tailed), lo que implica que una pequeña fracción de los documentos transferidos representan una gran fracción de bytes transferidos.

(g) Los tiempos de interreferencias entre documentos son independientes y distribuidos exponencialmente.

(h) El 10% de los archivos accedidos representan el 90% de las peticiones recibidas y bytes transferidos por el servidor.

(i) Los sitios remotos representan al menos el 70% de las referencias a los servidores.

(j) Los servidores Web son accedidos por miles de dominios, el 10% de los dominios cuentan con más del 75% del uso del servidor.

1.6.3 Cargas de Trabajo en la Caché del servidor Proxy

El estudio de los servidores proxy para los servicios Web, desarrollado por Abdulla en (Abdulla et al., 1998), considera las siguientes características:

(a) El tamaño de archivo promedio de los documentos solicitados es menor de 27 KB con una media aproximada de 2 KB.

(b) Entre el 90 y el 98% de los archivos transferidos son de tipo HTML y gráficos.

(c) Los rchivos gráficos representan la mayor parte de los bytes transferidos.

(d) El 25% de los servidores Web son responsables del 90% de las solicitudes y los bytes transferidos.

(f) La auto-similitud (del inglés *self- similarity)* está en los registros de acceso del proxy.

Características	Proxy-CESCA	Proxy-BU
Fecha	Dic. 2000	Ene. 1999
Número de Peticiones	3,089,592	2,123,432
Número de Clientes	11,765	7,845
Peticiones por Segundo	21	19
Bytes Transferidos	30GB	24GB
Tamaño medio de los objetos	2.5KB	2.2KB
Mediana del tamaño de los objetos	29KB	26KB
Ficheros HTML e Imágenes	94%	96%
Ficheros de Imágenes	64% de bytes transferidos	60% bytes transferidos
Servidores Referenciados una vez	4% de los servidores	5%
Accesos a Servidores Únicos	10% de los accesos	12%
Servidores responsables de más del 85% de referencias	25% de los servidores	24% de los servidores
Servidores responsables de más del 90% de los bytes accedidos	25% de los servidores	27% de los servidores
Trasnferencias Exitosas	98%	97%

Figura 1.8 Características similares de algunos proxy de caché.

1.7 Políticas de Reemplazo

Uno de los objetivos del estudio sobre la caracterización de la carga de trabajo es identificar las características que son inherentes al tráfico de la red visto por los clientes Web, servidores Web y servidores proxy para el servicio Web, y en base a estas características proponer una alternativa que permita obtener un mejor rendimiento en el acceso a la información. Estos estudios sobre los servidores Web y servidores proxy para los servicios Web han permitido identificar alternativas para esta realización, en particular para el almacenamiento en caché.

Según lo planteado en la caracterización de la carga de trabajo de servidor Web en (Arlitt y Williamson, 1997a; Arlitt, 1996; Bestavros et al., 1995; Williams et al., 1996) y las caracterizaciones de carga de trabajo de servidores proxy en (Abdulla et al., 1998; Mahanti, 1999; Mahanti y Williamson, 1999; Arlitt et al., 1999; Rizzo y Vicisano, 1998), se evidencia claramente como algunos documentos son mucho más solicitdos que otros. En el lado del servidor proxy, esto adquiere una mayor importancia debido a que se mueven documentos desde el disco a la memoria para reducir la latencia resultante de I /O en disco.

Una cuestión que debe tenerse en cuenta es lo relacionado con las políticas de reemplazo. Cuando el espacio de caché está totalmente ocupado y un nuevo documento arriba para ser almacenado, surge una interrogante sobre cuál es el documento (o los documentos) actualmente en la caché que deben ser removidos para hacer espacio para el próximo. Es importante que el servidor proxy tome buenas decisiones de reemplazo. Este problema es el de mayor preocupación para el rendimiento del servidor proxy, cuanto mayor sea el número de peticiones o el volumen de datos que se puedan satisfacer desde la caché, será mejor la política de reemplazo. Esto significa menos accesos a servidores remotos y mejor tiempo de respuesta. En un servidor proxy, esto se traduce en reducción de la carga, menos acceso a la red externa y una reducción en la latencia para los clientes.

Los algoritmos de reemplazo son comúnmente evaluados en base a dos parámetros: razón de aciertos en caché (HR) y la razón de aciertos de byte en caché (BHR) (Bestavros et al., 1995; Arlitt, 1996; Mahanti, 1999; Rizzo y Vicisano, 1998; Mahanti y Williamson, 1999; Arlitt y Williamson, 1997b; Shanmugavadivu y Rajaram, 2010; Arlitt y Williamson, 1997a). La razón de aciertos en caché es el número de solicitudes servidas directamente por la caché, dividido por el número total de peticiones de los usuarios, mientras que la proporción de aciertos de byte da la relación entre el total de bytes servidos por la caché a los usuarios, dividido entre el total de bytes solicitados por el proxy a los servidores de origen en Internet (transferidos al proxy desde Internet). Ambos parámetros son necesarios ya que los documentos Web tienen tamaños muy variables.

Varios investigadores han propuesto diferentes algoritmos en la literatura. Uno de los algoritmos más conocidos popularmente es el algoritmo “Menos Usado Recientemente” (LRU, del inglés *Least Recently Used* -) (Abrams et al., 1995; Yu y MacNair, 1998; Arlitt et al., 1998). En LRU, una lista de documentos se mantiene en la memoria caché y se ordena sobre la base de la disminución de experiencia de acceso reciente por el usuario. En otras palabras, el documento al que se ha accedido más recientemente es el primero de la lista, mientras que el documento menos accedido recientemente se encuentra en la parte inferior. Cuando se tiene acceso a un nuevo documento, uno o más documentos en la lista pueden ser eliminados de la caché con el fin de hacer espacio para el documento recién visitado (Yu y MacNair, 1998; Arlitt y Williamson, 1997a). El acceso frecuente a un documento que ya está en la caché coloca el documento en la parte superior de la lista.

LRU se utiliza en el software de los servidores proxy de dominio público (Squid) (Maltzahn et al., 1997; Rousskov y Soloviev, 1998; Wessels y Claffy, 1998).

La política "Menos Usado Frecuentemente" (LFU, del inglés *Least Frequently Used*) (Arlitt y Williamson, 1997a; Nishikawa et al., 1998; Arlitt et al., 1998; Arlitt, 1996; Williams et al., 1996) es otro algoritmo muy popular, que toma las decisiones de sustitución basada en la frecuencia de acceso a cada documento. En LFU cada documento tiene asociado un recuento del número de veces que ha sido visitada. El acceso a un nuevo documento que no está en el caché puede provocar una decisión de reemplazo, en cuyo caso se elimina el documento con el número más bajo de referencias. Si dos documentos tienen el mismo número de referencias, entre estos el que se haya accedido menos recientemente es eliminado. En otras palabras, la política LRU se utiliza para romper el empate.

LRU y LFU no toman en cuenta el tamaño del documento. Un algoritmo basado en el tamaño es estudiado en (Arlitt y Williamson, 1997a; Williams et al., 1996). Con la política SIZE, el archivo más grande se retira cuando hay una necesidad de eliminar un documento de la caché. A menudo, esto significa que una gran cantidad de pequeños o más archivos pueden ser alojados en la memoria caché.

El algoritmo FIFO (del inglés *First In First Out*) significa, por sus siglas en inglés, que el primer objeto en entrar en la caché es el primer archivo en salir de la caché. Esta política toma como parámetro para eliminar un objeto la edad del objeto en la caché, mientras más tiempo tenga en la caché, mayor es su probabilidad de ser eliminado. El reemplazo en esta política es un consecutivo lineal de los objetos Web.

El algoritmo RANDOM realiza la función de reemplazo por medio de la selección aleatoria de un objeto, sin tomar en cuenta ninguna clase de parámetro ni propiedad del objeto que será eliminado en la caché. Esta política de reemplazo es estudiada por motivos académicos, pero en realidad su desempeño es bajo en comparación con otras políticas.

1.8 Métricas y factores de rendimiento

Varios indicadores se utilizan comúnmente en la evaluación de rendimiento de la caché para servicios Web. La razón de aciertos (HR), como se ha mencionado anteriormente, es el

porcentaje de solicitudes que fueron atendidas gracias a que se encontraba en caché una copia del objeto solicitado.

$$HR = \frac{Número\ de\ aciertos}{Número\ total\ de\ solicitud}$$

Un alto valor en la tasa HR refleja una política eficaz. Si los documentos son homogéneos en tamaño esta métrica será de gran eficacia. Si los resultados son de distintos tamaños, la razón de aciertos de bytes (BHR) resultará en una medida de mejor rendimiento. El BHR se define como el cociente entre el número de bytes cargados de la caché y el número total de bytes recibidos por el proxy desde Internet.

$$BHR = \frac{Números\ bytes\ servidos\ por\ Caché}{Números\ bytes\ totales}$$

La utilización de ancho de banda es otra medida cuyo objetivo evidente es la de reducir la cantidad de ancho de banda consumido.

Una cuarta medida es el tiempo de respuesta percibido por el usuario, es decir, el tiempo que un usuario espera por el sistema para obtener un documento solicitado.

Otras medidas de la eficiencia en el acceso a la información pueden incluir la caché de la CPU del servidor y la utilización del sistema I/O, la fracción del total de ciclos de CPU, velocidad y tamaño de caché en los disco y la latencia en la recuperación de un objeto (o el tiempo descarga de la página), que son de especial interés para los usuarios finales. La latencia es inversamente proporcional a la razón de aciertos (Hit Rate), debido a que un HIT de caché se puede cargar con mayor rapidez que una solicitud que debe pasar a través de la caché a un servidor de origen o remoto. El aumento de la razón de aciertos no implica necesariamente una disminución en la latencia del acceso. El almacenamiento en caché de algunos documentos con una latencia de descarga alta en realidad podría reducir la latencia media.

Varios factores afectan el rendimiento de la caché en los servicios Web. El comportamiento de la población de usuarios que solicitan documentos de la caché se caracteriza por el patrón de acceso de los usuarios. Si un usuario accede a un pequeño número de documentos la mayor parte del tiempo, estos documentos serán candidatos obvios para el

almacenamiento en caché. Los patrones de acceso no son estáticos y por lo general esto implica que una política de caché eficaz no debe ser estática.

1.9 Tamaño de la caché

El tamaño de caché es otro factor que influye en el rendimiento de la caché. Cuanto mayor sea el tamaño de la caché mayor es el número de documentos que pueden mantener y mayor es la proporción de aciertos de la caché. Una pregunta muy común es, "¿Qué tan grande debe ser el volumen de mi caché?" Es posible que simplemente al aumentar el espacio de disco se aumente la tasa de éxito, y por lo tanto el rendimiento pueda mejorar. Esto no es necesariamente cierto, hay un punto donde la tasa de aciertos de caché no incrementa significativamente a pesar de que se aumenta el espacio disponible en disco. Objetos en caché tiene un TTL (Time To Live) que especifica el tiempo que el objeto se puede mantener. Cada página solicitada se compara con el TTL, y si el objeto en caché es demasiado viejo, una nueva copia se solicita a través de Internet y el TTL se restablece.

Esto significa que los volúmenes de una gran caché podrían llenarse de una gran cantidad de datos antiguos que no serán reutilizados. El tamaño de caché de disco por lo general debe estar entre 10 y 18GB (Flickenger, 2006). Pocos administradores de caché han trabajando con grandes discos para caché con capacidades de 140 GB o más, pero en la práctica sólo utilizan un tamaño de caché de alrededor de 18GB. Si se especifica un volumen caché muy grande, el rendimiento en realidad puede decaer. El proxy tiene que manejar muchas solicitudes simultáneas a la caché, el rendimiento puede disminuir a medida que aumenta la cantidad de solicitudes por segundo con una caché muy grande. El bus de acceso al disco podría convertirse en un cuello de botella.

1.10 Introducción al modelo de tiempo de respuesta

La mayoría de los estudios de simulación contemplan el modelado de colas. Por esta razón se considera conveniente adquirir conocimientos básicos sobre la teoría de colas para construir los modelos y los esquemas de simulación con facilidad y exactitud. Para modelos simples la teoría de colas puede ayudar a analizar el comportamiento del modelo. Para

modelos complejos los conceptos ligados a la teoría de colas pueden facilitar el análisis de los resultados de la simulación.

1.10.1 Notación de Kendall

A principios de la década de los cincuenta Kendall define una clasificación de los sistemas que se caracterizan por tener una sola cola y el arribo aleatorio de los usuarios con la nomenclatura A/B/k

- A : tipo de distribución de probabilidad de tiempos entre arribos consecutivos
- B : tipo de distribución de probabilidad del tiempo de servicio
- k: número de servidores.

Los tipos de distribución más estudiadas son:

- M : distribución exponencial o "sin memoria" (la ´M´ viene del inglés *memory-less*
- G : distribución general con media 1/μ y varianza O_s^2
- D : deterministas o constantes

1.10.2 Sistemas M/M/1

En un sistema M/M/1 la razón de arribo (λ) de las solicitudes sigue una distribución de tipo exponencial, al igual que el tiempo de servicio con una razón de terminación μ y un tiempo promedio de servicio para cada solicitud igual a $\frac{1}{\mu}$.

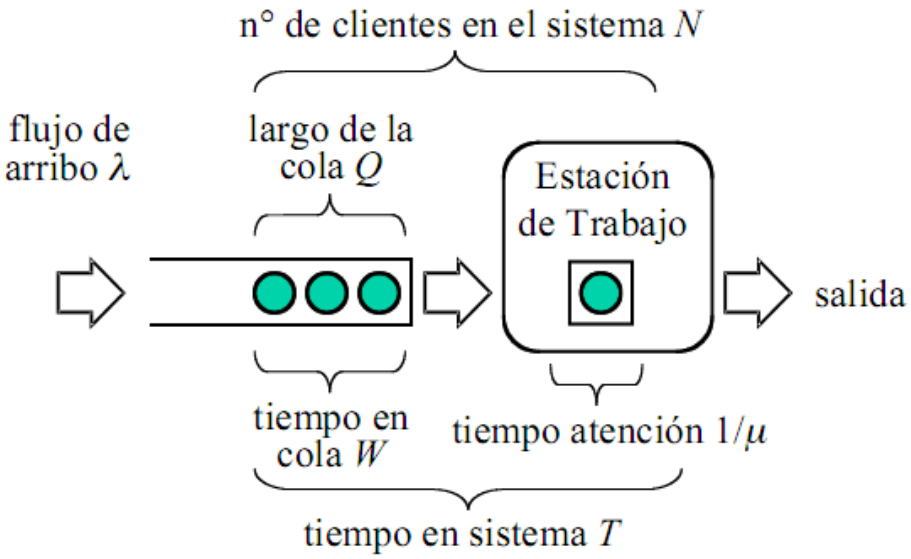


Figura 1.9: Sistema de espera M/M/1.

El número previsto de usuarios en el sistema puede ser descrito por la siguiente expresión:

$$Q = \sum_{n=0}^{\infty} nP_n \quad (P_n: \text{probabilidad de que haya n clientes en el sistema})$$

Según la ley de Little:

$$Q = \frac{\lambda}{\mu - \lambda}$$

Entonces el tiempo previsto de permanencia en el sistema es:

$$T = \frac{Q}{\lambda} = \frac{\lambda}{\mu - \lambda} \cdot \frac{1}{\lambda} = \frac{1}{\mu - \lambda}$$

1.10.3 Teorema de Jackson

En 1957 se publica el teorema de Jackson (Jackson, 2004). En dicho teorema se demuestra que una red de puesta en fila de espera con *n* nodos (*M/M/n*) tiene forma de producto. Sus conclusiones fueron inspiradas por el resultado obtenido por Burke el año anterior.

Teorema de Jackson: Considérese una red de puesta en fila de espera abierta con *K* nodos que satisfacen las siguientes condiciones:

a) Cada nodo es un sistema de puesta en fila *M/M/n*. El nodo *k* tiene n_k servidores y el promedio del tiempo de servicio es $1/\mu_k$.

b) Los clientes llegan desde fuera del sistema al nodo *k* conforme a un proceso de Poisson con intensidad λ_k. Pueden llegar también clientes de otros nodos al nodo *k*.

c) Un cliente, que acaba de finalizar su servicio en el nodo *j*, se transfiere inmediatamente al nodo *k* con probabilidad p_{jk} o sale de la red con probabilidad.

$$1 - \sum_{k=1}^{K} p_{jk}$$

Un cliente puede visitar varias veces el mismo nodo si $p_{kk} > 0$.

El promedio de la intensidad de arribo Λ_k en el nodo k se obtiene empleando las ecuaciones de equilibrio de flujo:

$$\Lambda_k = \lambda_k + \sum_{j=1}^{K} \Lambda$$

Sea $p_{(i1, i2, \ldots, iK)}$ la representación de las probabilidades de espacio de estado conforme a la hipótesis de equilibrio estadístico, es decir la probabilidad que haya i_k clientes en el nodo *k*. Así mismo, se supone que:

$$\frac{\Lambda_k}{\mu_k} = A_k < n_k$$

Las probabilidades de espacio de estado vienen dadas entonces en forma de producto:

$$p(i_1, i_2, \ldots, i_K) = \prod_{k=1}^{n} p_k(i_k)$$

Aquí para el nodo *k*, $p_k(i_k)$ es la probabilidad de estado de un sistema de puesta en fila *M/M/n* con intensidad de llegadas Λ_k y razón de servicio μ_k. El tráfico ofrecido $\Lambda c/\mu k$ al nodo *k* debe ser menor que la capacidad n_k del nodo para entrar en equilibrio estadístico.

El punto fundamental del teorema de Jackson es que cada nodo puede ser considerado independientemente de los otros y que las probabilidades de estado vienen dadas por las fórmulas C de Erlang. Esto simplifica considerablemente el cálculo de las probabilidades de estado. La prueba del teorema fue obtenida por Jackson en 1957 demostrando que la solución satisface las ecuaciones de equilibrio para el equilibrio estadístico.

En el último modelo de Jackson la razón de arribos que proveniente del exterior puede depender del número actual de usuarios en la red, lo cual se refleja en la ecuación siguiente:

$$\lambda = \sum_{j=1}^{K} \gamma$$

Asimismo, μ_k puede depender del número de clientes en el nodo *k*. De esta manera, se pueden modelar redes de puesta en fila que sean cerradas, abiertas o mixtas. En los tres casos, las probabilidades de estado tienen forma de producto.

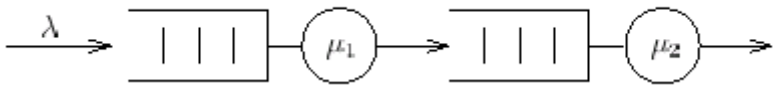


Figura 1.10: Diagrama de transición de estado de una red de puesta en fila abierta constituida por dos sistemas *M/M/1* en serie.

1.11 Conclusiones del capítulo 1.

La temática asociada con el empleo de caché en servidores proxy para el servicio Web ha sido ampliamente abordada en la literatura científica desde hace algunos años. El estudio y análisis del estado del arte en esta temática ha permitido determinar una serie de factores que han sido abordados de manera independiente y que todos repercuten en la calidad de servicio del usuario final. El análisis integral de todos ellos como un sistema único podría conducir a una mejora significativa en la calidad de servicios percibida por los usuarios finales que acceden a Internet con el empleo de un servidor proxy y limitado ancho de banda en el acceso.

CAPÍTULO 2. MECANISMO DE FUNCIONAMIENTO DE LAS POLÍTICAS DE REEMPLAZO EN CACHÉ Y MODELO ANALÍTICO PARA LA EVALUACIÓN DEL TIEMPO DE RESPUESTA

Las ventajas en el empleo de un servidor proxy con caché dependen de varios factores. Los factores más importantes a tener en cuenta son la probabilidad de archivos deseados ya almacenados en la caché (HR, BHR), las políticas de reemplazo de caché, la velocidad de la caché del servidor proxy, el ancho de banda de conexión a Internet, la velocidad del servidor Web remoto, y el ancho de banda disponible por la red donde se encuentra alojado el sitio Web remoto .

Para discernir el efecto de los factores mencionados en el rendimiento del servidor proxy con caché se utiliza una red de colas para modelar la dinámica con que se manejan las solicitudes de los usuarios en un servidor proxy caché.

2.1 Modelo analítico para la evaluación del tiempo de respuesta.

Como se observa en la figura. 2.1 (Bose y Cheng, 2000), las solicitudes de páginas Web o archivos que llegan al servidor proxy caché es una distribución de Poisson de λ por unidad de tiempo, donde F denota el tamaño medio de los archivos solicitados. La probabilidad de que el servidor proxy caché puede cumplir con las solicitudes es p. Por lo tanto, p es la probabilidad de que los archivos deseados ya residen en el servidor proxy caché. Con (1 - p) se denota la probabilidad de que el servidor proxy caché tiene que buscar los archivos solicitados en el sitio Web remoto. Definimos λ_1 y λ_2 de tal manera que:

$$\lambda = \lambda_1 + \lambda_2$$

Donde $\lambda_1 = p\lambda$

Y $\lambda_2 = (1 - p)\,\lambda$.

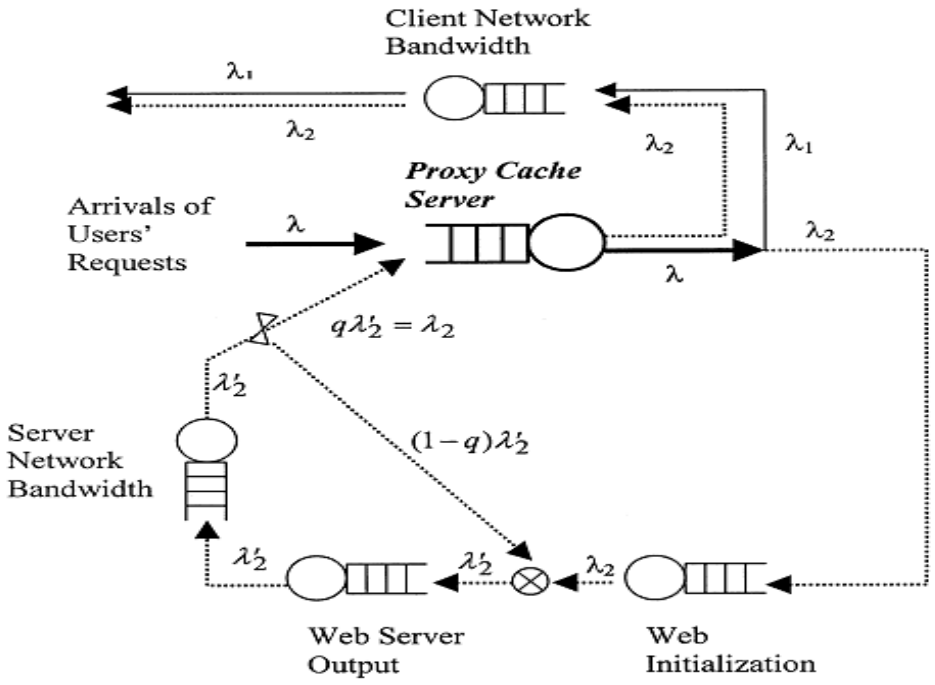


Figura 2.1: Esquema del modelo analítico.

El tráfico λ_1, representado por una línea continua en la figura 2.1, entrega el contenido disponible del servidor proxy caché al usuario solicitante.

La línea discontinua en la figura 2.1 representa el tráfico de λ_2 donde se obtienen los archivos solicitados del servidor Web remoto y son insertados (el flujo de trabajo regresa a este punto) en la caché del servidor proxy PCS (del inglés *Proxy Cache Server*).

El tráfico λ_2 pasa primeramente a través de un estado de inicialización (proceso de establecimiento de sesión que se realiza solo una vez). Este proceso de inicialización consumeel tiempo utilizado por todos los procesos de "*hand shakings*" necesarios para

establecer un Protocolo de Control de Transmisión (TCP, del inglés *Transmission Control Protocol*) entre el servidor proxy caché de la empresa y el sitio Web remoto.

Se utiliza el parámetro I_S para caracterizar esta inicialización . A continuación el servidor Web remoto recupera los archivos solicitados y los entrega al servidor proxy caché de la empresa. Una copia de los archivos esdescargada en el usuario que la solicita.

El rendimiento de los servidores Web remotos se caracteriza por:

- El tamaño de su búfer de salida (en bytes), B_s.
- El tiempo estático del servidor (en segundos), I_s.
- La razón dinámica del servidor (en byte por segundos), R_s.

Del mismo modo, servidor proxy caché de la empresa se caracteriza por B_{xc}, I_{xc}, y R_{xc} donde el subíndice "*xc*" significa" caché del proxy.

No es inusual para el tamaño del archivo solicitado, F, exceder el tamaño de búfer de salida del servidor Web remoto, B_s. En este caso, puede tomar varias iteraciones la recuperación y entrega de archivos más pequeños para completar la solicitud de servidor proxy caché. Este fenómeno de bucle es inherente a la tecnología Hyper Text Transfer Protocol (HTTP), donde la recuperación de la página principal es seguida por la recuperación de imágenes incrustadas en línea.

Para modelar este bucle, se denota como q a la probabilidad de ramificación (fragmentación del paquete de respuesta debido a la capacidad del buffer de salida del servidor Web) de que una solicitud del servidor proxy caché pueda ser servida en el primer intento;

$$q = \min \{1,(B_s/F)\}$$

En consecuencia, la proporción $(1 - q)$ de las solicitudes se repetirá de nuevo al servidor Web remoto para su posterior procesamiento. En el equilibrio, el tráfico que sale del

servidor Web remoto hacia el Servidor proxy caché después de la ramificación debe ser igual al tráfico entrante original λ_2. Por lo tanto $q\lambda'_2$ es igual a λ_2 (figura 2.1), donde λ'_2 es el tráfico que sale del canal de acceso (ancho de banda de salida, server network bandwidth) desde el servidor remoto debido a la ramificación.

La red de colas de la figura 2.1 puede ser considerada como una red de Jackson. En esencia, esta red se comporta como si todos los nodos fueran independientes M/M/1 colas (Gross y Harris, 1985). Siguiendo esta idea se obtiene que el tiempo de respuesta global para completar las solicitudes de los usuarios en la presencia de un Servidor proxy caché está dada por la ecuación 2:

$$T_{xc} = \frac{1}{\frac{1}{I_{xc}} - \lambda} + p.\left\{\frac{1}{\frac{1}{\frac{F}{B_{xc}}\left[Y_{xc}+\frac{B_{xc}}{R_{xc}}\right]} - \lambda_1} + \frac{F}{N_c}\right\} +$$

$$+(1-p).\left\{\frac{1}{\frac{1}{I_s} - \lambda_2} + \frac{1}{\frac{1}{\frac{F}{B_s}\left[Y_s+\frac{B_s}{R_s}\right]} - \frac{\lambda_2}{q}} + \frac{F}{N_s} + \frac{1}{\frac{1}{\frac{F}{B_{xc}}\left[Y_{xc}+\frac{B_{xc}}{R_{xc}}\right]} - \lambda_2} + \frac{F}{N_c}\right\} \quad (2)$$

- El primer término $\frac{1}{\frac{1}{I_{xc}} - \lambda}$ es el tiempo esperado de la búsqueda necesaria para ver si los archivos que se desean ya están disponibles en el Servidor proxy caché. Con probabilidad p, el contenido ya está almacenado.

- El segundo término de la ecuación $p.\left\{\frac{1}{\frac{1}{\frac{F}{B_{xc}}\left[Y_{xc}+\frac{B_{xc}}{R_{xc}}\right]} - \lambda_1} + \frac{F}{N_c}\right\}$ describe el tiempo esperado para el contenido que se entrega al usuario solicitante.

- El tercer término en la ecuación identifica el tiempo esperado necesario el momento en que el Servidor proxy caché inicia la recuperación los archivos deseados y el

servidor Web remoto transfiere los archivos al servidor proxy caché, a la vez que el Servidor proxy caché entrega una copia al usuario solicitante.

$$(1-p).\left\{\frac{1}{\frac{1}{I_S}-\lambda_2}+\frac{1}{\frac{1}{\frac{F}{B_S}\left[Y_S+\frac{B_S}{R_S}\right]}-\frac{\lambda_2}{q}}+\frac{F}{N_S}+\frac{1}{\frac{1}{\frac{F}{B_{XC}}\left[Y_{XC}+\frac{B_{XC}}{R_{XC}}\right]}-\lambda_2}+\frac{F}{N_C}\right\}$$

De la figura 2.1 y la ecuación 2, la razón de arribos en el servidor Web remoto es λ'_2, que es igual a λ_2 / q, debido a la naturaleza del bucle de procesamiento en el servidor Web remoto. Sin un servidor proxy caché, nuestro modelo se reduce a un caso especial reportado en la literatura (Slothouber, 1996) donde se describe el tiempo de respuesta global en la ecuación 3:

$$T=\frac{1}{\frac{1}{I_S}-\lambda}+\left\{\frac{1}{\frac{1}{\frac{F}{B_S}\left[Y_S+\frac{B_S}{R_S}\right]}-\frac{\lambda}{q}}+\frac{F}{N_S}+\frac{F}{N_C}\right\} \quad (3)$$

Una rápida inspección de las ecuaciones 2 y 3 muestra que ambas contienen el término F / N_c. Los límites máximos de las tasas de recepción de las solicitudes del usuario, λ_{max}, y el tamaño medio de los archivos F_{max}, se puede obtener, sobre la base de no permitir términos negativos, la ecuación 2 de la siguiente manera:

$$\lambda_{max}=min\left\{\frac{1}{I_S},\frac{1}{I_{XC}},\frac{qB_SR_S}{F(R_SY_S+R_S)},\frac{qB_{XC}R_{XC}}{F(R_{XC}Y_{XC}+R_{XC})}\right\} \quad (4)$$

$$F_{max}=min\left\{\frac{qB_SR_S}{\lambda(R_SY_S+R_S)},\frac{qB_{XC}R_{XC}}{\lambda(R_{XC}Y_{XC}+R_{XC})}\right\} \quad (5)$$

La implicación de las ecuaciones 4 y 5 es que cuando las razones de arribo o el tamaño de los archivos superan los límites, el resultado es un tiempo de respuesta intolerablemente largo de los usuarios.

2.2 Exploración numérica de los parámetros utilizados en el modelo.

Varios experimentos numéricos se han llevado a cabo para examinar el comportamiento de la ejecución del Servidor proxy caché utilizando valores de parámetros diferentes. Los valores de los parámetros de los experimentos se resumen en la Tabla 1. Estos experimentos tratan de determinar las condiciones que hacen a un servidor proxy caché beneficioso. La razón de arribo de las solicitudes de los usuarios λ, se varía desde 10 hasta 90 peticiones/s. Algunas publicaciones, por ejemplo en (Rodriguez y Biersack, 1998), experimentan con sólo un máximo de 10 peticiones/s. Sin embargo esta gama más amplia de la razón de arribo es considerada más acertada siempre y cuando se encuentre dentro del rango especificado en la ecuación 4. El tamaño medio de los archivos solicitados F, varía mucho de un sitio Web a otro. Slothouber (Slothouber, 1996) visitó 1000 páginas Web de forma aleatoria usando una función de enlace al azar con varios motores de búsqueda y obtuvo un valor de 5275 bytes para el tamaño medio de los archivo. Por lo tanto, se experimenta con valores de F por debajo y por encima de 5275 bytes, los cuales van desde 1250 a 8750. Este rango de valores es consistente con lo planteado en (Almeida et al., 1996b), donde el 94 % de los archivos solicitados fueron reportados por debajo de los 50 Kbytes. El tamaño del búfer de salida del servidor Web B, es igual a 2000 bytes como en la referencia (Slothouber, 1996). El tiempo de inicialización de una sola vez del servidor Web remoto Is, es igual a 0,004 s. El tiempo estático del servidor Web remoto Ys, es de 0.000016 s y la razón dinámica del servidor R, se ha fijado en 1,25 Mbyte/s. Estos valores han sido elegidos para cumplir con las características de rendimiento de los servidores Web reportados en la literatura (Menasce y Almeida, 1998).

Tabla 2: Valores típicos de los parámetros experimentales.

Parámetros	Valor experimentado
λ	10-90 solicitud por segundo
F	1250 – 8750 bytes
B_s	2000 bytes

I_s	0.0004 segundo
Y_s	0.000016 segundo
R_s	1.25 Mbyte por segundo
$\alpha = B_{xc} / B_s$	0.1 – 1.0
$\beta = R_{xc} / R_s = Y_{xc} / Y_s$	0.1 – 1.0
$\gamma = I_{xc} / I_s$	0.1 – 1.0
p	0.1 – 0.9

Tabla 3: Ancho de banda N_c y N_s

Tipo de conexión	Ancho de banda (Kbps)
ISDN 1	64
ISDN 2	128
T1	1,544
T3	45,000
Ethernet	10,000
Fast Ethernet	100,000
OC – 3	154,400
OC -12	622,000

Tabla 4: Los parámetros del modelo analítico

λ	Razón de arribo de las solicitudes (en números de solicitud por segundo)
F	Tamaño medio del fichero solicitado
B_s	Tamaño del búfer de salida en el servidor Web
I_s	Tiempo total necesario para el proceso de inicialización en el servidor Web remoto
Y_s	Tiempo estático del servidor Web
R_s	Razón dinámica del servidor de Web
B_{xc}	Tamaño del búfer de salida en el servidor proxy caché
I_{xc}	Tiempo de búsqueda en el servidor proxy caché
Y_{xc}	Tiempo estático del servidor proxy caché
R_{xc}	Razón dinámica del servidor proxy caché

Con el fin de entender el efecto de la velocidad del proxy se hace que sus valores dependan de los parámetros del servidor Web remoto de la siguiente manera. En los primeros experimentos, α, β y γ se fijan en 1.0, lo que implica que el servidor Web remoto y Servidor proxy caché tienen idénticas características de rendimiento. En la última serie de experimentos, se varían α, β y γ desde 0,1 a 1,0 para investigar el efecto de la velocidad del Servidor proxy caché en el tiempo de respuesta global sobre las peticiones de los usuarios. La probabilidad de que los archivos solicitados estén disponibles en el Servidor proxy caché se varió desde 0,1 hasta 0,9. Las estadísticas empíricas de las gamas de *p* van desde 21,3 hasta 56,6% según (Baentsch et al., 1997) y entre 30 y 60% en (Luotonen, 1998).

Las opciones disponibles de valores para el ancho de banda de red del cliente N_c, y el ancho de banda del servidor de red N_s, responden a las publicadas en (Menasce y Almeida, 1998) (véase la tabla 3).

En los experimentos se ha asumido de manera objetiva que la red del lado del cliente es más lenta que la red del lado del servidor y se han configurado los parámetros de ancho de banda N_c = 128 Kbps y N_s = 1,544 Kbps.

2.3 Mecanismos de funcionamiento de las políticas de reemplazo en caché.

2.3.1 LRU

LRU es una política incluida en el marco de las políticas basadas en "el uso reciente". Éstas políticas se basan en analizar el uso reciente o no de los objetos almacenados como el factor principal, de tal forma que la mayor parte de los algoritmos que forman parte de este grupo son en mayor o menor medida una extensión de la estrategia LRU.

Estas estrategias se basan en el principio de localización para justificar su comportamiento. Existen dos formas de diferenciar la localización: localización espacial y localización. Ambas son descritas a continuación:

- **Localización Espacial:** Hace referencia al principio que realiza la siguiente suposición: Las referencias a determinados elementos de la caché implican necesariamente que exista una mayor probabilidad de realizar una llamada a un determinado grupo de elementos en la caché. De tal manera que la llamada a ciertos objetos podría servir como un predictor de futuras referencias.
- **Localidad Temporal:** Se basa en el hecho de que recientes aciertos de elementos de la caché son mucho más probables de ser referenciados nuevamente que el resto.

El algoritmo LRU se basa el principio de localidad temporal, es decir, pretende mantener siempre en memoria los objetos que más recientemente se han referenciado, o dicho de otra manera, los objetos que siguiendo el principio de localidad temporal tienen más probabilidad de volverse a referenciar.

Este algoritmo forma parte de aquellos basados en el"uso reciente", pero desde luego no es el único. Algunos de los algoritmos que también forman parte de este grupo son LRU-Treshold, LRU-Min, Size, Value Aging EXP1, PSS, LRU-LSC, Partitioned Caching, Pitkow/Reckers strategy o HLRU (Podlipnig y Böszörmenyi, 1998).

Las ventajas de las estrategias basadas en el *"uso reciente"* son las siguientes:

- Ellas consideran la localidad temporal como un factor principal. Dado que las peticiones Web muestran cierto orden por su localización temporal, puede resultar un sistema ventajoso.
- Son estrategias fáciles de implementar y rápidas. Muchas de estas estrategias emplean una lista LRU. Nuevos objetos son insertados en la cabeza de la caché. Un acierto provoca que el objeto en cuestión sea eliminado de su posición actual y se reinserte en la cabeza de la caché. Los objetos más susceptibles de ser reemplazados se aglutinarán en torno al final de la caché. De tal manera, la inserción y eliminado se convierten en operaciones de una baja complejidad. Igualmente, la búsqueda puede ser soportada por técnicas de búsqueda *hashing*.

A continuación se exponen algunas desventajas de acuerdo a las características de este grupo de políticas de reemplazo:

- No consideran información alguna sobre la frecuencia con la que se referencian los elementos de la caché. Lo cual será una seria desventaja a tener en cuenta, ya que éste es un importante indicador en escenarios de tráfico Web.
- Otra de las desventajas que afecta en gran medida las estrategias derivadas de la LRU es que no combinan el almacenamiento de objetos recientemente llamados con su tamaño de una manera balanceada. Los documentos Web suelen ser de diferente tamaño, por lo tanto el tamaño es un factor a tener en cuenta en cada reemplazo.

A continuación se explicará el funcionamiento de este algoritmo. Como su propio nombre lo indica, la estrategia LRU reemplaza el documento que menos reciente haya sido referenciado y éste es el principio fundamental de su funcionamiento. Pero, es conveniente desarrollar más ampliamente éste método, para ello se describe cómo actúa esta política ante diversas circunstancias (inicialmente la caché está vacía):

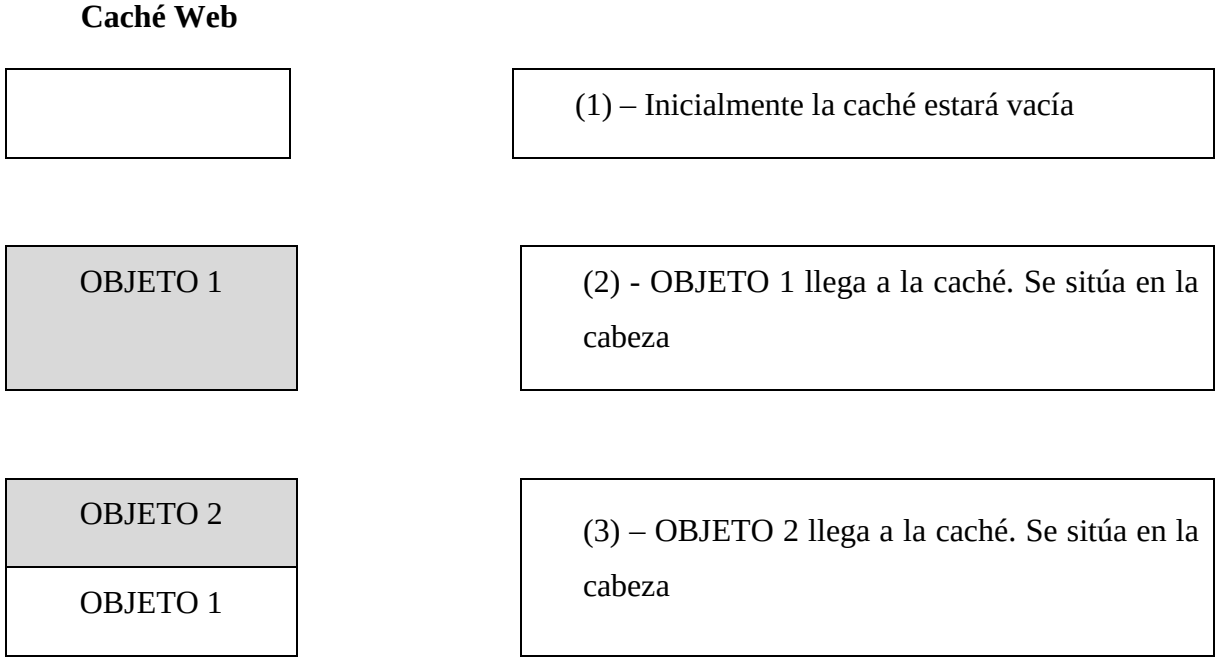
1) **Llega un elemento nuevo** → Se analiza si el nuevo elemento cabe en la caché. De ser así se almacenará en la cabeza de la caché (es decir, en la posición superior de la misma). El resto de los elementos almacenados serán desplazados en una posición.

2) **Llega un elemento nuevo que no cabe** → Se eliminan objetos de la caché comenzando desde el último hasta que el espacio liberado sea lo suficiente como para insertar el nuevo elemento (en la cabeza de la caché).
3) **Se produce un acierto en la caché** → O lo que es lo mismo, el documento solicitado está en la caché y por lo tanto será servido por la misma. En tal caso se realizará una búsqueda de este objeto almacenado y una vez localizado (se consigue su índice) se reubica de nuevo en la cabeza del sistema (se inserta como si se tratase de un elemento nuevo).

Por supuesto que en el caso de llegar un elemento cuyo tamaño exceda el propio tamaño de la caché, este será directamente descartado (esto será común para todas las políticas de reemplazo).

Se puede comprobar cómo efectivamente los objetos más recientes, ya sea porque son nuevos o porque son elementos acertados, se ubican siempre en la zona superior de la caché en torno a su cabeza. Por el contrario, los elementos referenciados menos reciente permanecerán en el final del sistema, siendo candidatos óptimos para su eliminación.

A continuación se realiza una pequeña demostración:



OBJETO 3
OBJETO 2
OBJETO 1

(4) – OBJETO 3 llega a la caché. Se sitúa en la cabeza

OBJETO 4
OBJETO 3
OBJETO 2

(5) – OBJETO 4 llega a la caché. No cabe. Se elimina el último objeto de la misma (OBJETO1). Ya cabe y se inserta en la cabeza.

OBJETO 3
OBJETO 4
OBJETO 2

(6) – OBJETO 3 llega a la caché. Éste elemento ya está almacenado, es un acierto. Se reubica en la cabeza del sistema.

En resumen esta política tradicional es en la práctica la más frecuentemente usada y ofrece un gran rendimiento para caché de la CPU y sistemas de memoria virtual. Su rendimiento no es tan conveniente para sistemas de caché Web debido a que la localización temporal del tráfico Web a menudo presenta patrones muy diversos.

2.3.2 LFU

LFU es una política de reemplazo agrupada dentro de las políticas ”basadas en la frecuencia”. Estas estrategias usan la frecuencia como característica principal, es decir, el número de veces que se ha realizado una llamada a un documento determinado, siendo en mayor o menor medida una extensión de la propia LFU, tal y como ocurría en las estrategias basadas en el uso reciente con la política LRU. Se basan en el hecho de que los

diversos documentos Web poseen una distinta popularidad la cual se traduce en un mayor o menor indicador de frecuencia. Estas políticas emplean dicho indicador para tomar futuras decisiones.

Este grupo de políticas pueden implementarse de dos formas distintas:

- Perfect LFU: Se crea un contador que se incrementa con cada llamada a un objeto, este factor permanecerá inalterable aún cuando el elemento haya sido eliminado de la caché, de esta manera, este factor mantiene una cuenta de todas las referencias a dicho objeto, presentes y pasadas.
- In-Caché LFU: En este caso, dicho contador se reseteará cada vez que el elemento en cuestión sea eliminado de la caché, de tal manera que el contador mantendrá la cuenta de las veces que se han referenciado únicamente los elementos que se encuentren en caché.

LFU forma parte de una serie de algoritmos basados en la frecuencia, pero como sucedió con LRU no es el único. Algunos de los cuáles son LFU Aging, LFU-DA, α-Aging o sw-LFU.

De igual manera, el empleo de políticas basadas en la frecuencia genera una serie de ventajas e inconvenientes.

Como ventaja se tiene la siguiente:

- Tienen en cuenta la frecuencia de acceso a los elementos almacenados, es decir, objetos más veces referenciados tendrán una mayor prioridad sobre el resto a la hora de evitar ser eliminados.

Algunas de las desventajas son:

- Estas son políticas de una mayor complejidad, ya que requieren un tratamiento más elaborado de la caché. En estas estrategias suele ser extendido el uso de colas de prioridad.
- Muchos de los objetos almacenados pueden llegar a tener el mismo valor de frecuencia de llamadas, en tal caso sería necesario algún factor que reordenase los elementos en la lista.

- Estas estrategias son invariantes a los cambios de tráfico, es decir, es posible que documentos muy referenciados en el pasado pero totalmente marginados actualmente, permanezcan en la zona superior de la caché, y por lo tanto no serán eliminados. Existen técnicas de envejecimiento o Aging, que buscan solucionar tal circunstancia, a costa de complicar aún más el algoritmo.

A continuación se ilustra el funcionamiento de este algoritmo. La estrategia LFU reemplaza el documento que menor número de veces haya sido referenciado, siendo esto el principio fundamental de su funcionamiento. Seguidamente se desarrolla más ampliamente éste método, para ello se describe cómo actúa esta política ante diversas circunstancias:

1)**Llega un elemento nuevo** → Se analiza si el nuevo elemento cabe en la caché. De ser así, se almacena en la misma de manera ordenada en función de su contador de frecuencias, lo que a partir de ahora se denomina como contador de referencia. Así elementos con un mayor contador de referencia se situarán siempre en la zona superior de la caché, mientras que elementos que han sido pocas veces referenciados estarán en la zona inferior de la misma y serán futuros candidatos para ser eliminados.

2)**Llega un elemento nuevo que no cabe** → Se eliminan objetos de la caché, comenzando desde el último, hasta que el espacio liberado sea el suficiente como para insertar el nuevo elemento (en la posición que le corresponda en función de su contador de referencia).

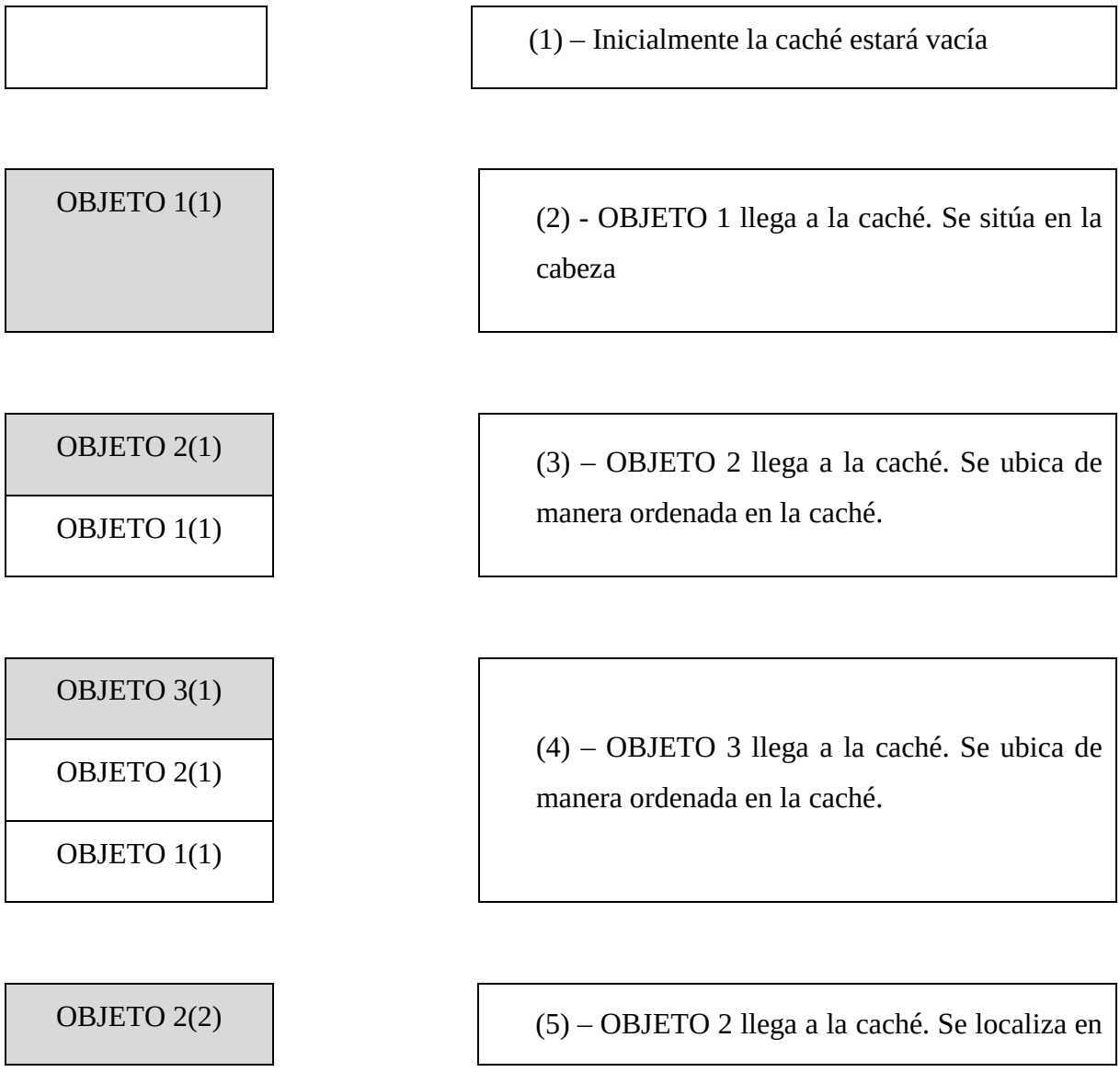
3)**Se produce un acierto en la caché** → O lo que es lo mismo, el documento solicitado está en la caché, y por tanto será servido por la misma. En tal caso se realizará una búsqueda de este objeto almacenado y una vez localizado (se consigue su índice) se incrementa su contador de referencia (ya que ha sido llamado una vez más) y se reubica de nuevo en el sistema en función de su nuevo contador de referencia.

Con todo esto, se obtiene una caché ordenada en función del valor del contador de referencia de cada uno de los elementos que la componen, de tal manera que los objetos más *"populares"* (más veces referenciados) permanecen en la zona superior del sistema;

mientras que los elementos menos *"populares"* quedarán en la zona inferior, listos para ser eliminados.

Se ha supuesto que, en caso de existir elementos con un mismo contador de referencia, éstos se ordenarán en función de cuan recientemente fueron llamados, es decir, el objeto que más recientemente se referenció estará por encima del resto.

En el siguiente diagrama se realiza una pequeña demostración (entre paréntesis se muestra el valor del contador de referencia del elemento en cuestión):



OBJETO 3(1) OBJETO 1(1)	el sistema. Se incrementa su contador y se reubica en la zona superior de la caché.
OBJETO 2(2) OBJETO 4(1) OBJETO 3(1)	(6) – OBJETO 4 llega a la caché. No cabe. Se elimina el último objeto de la misma (OBJETO1). Ya cabe. Se inserta de manera ordenada en la caché en función del contador.
OBJETO 3(2) OBJETO 2(2) OBJETO 4(1)	(7) – OBJETO 3 llega a la caché. Se localiza en el sistema. Se incrementa su contador y se reubica en la zona superior de la caché.

En resumen, el algoritmo LFU reemplaza el documento que ha sido referenciado un menor número de veces. Esta estrategia intenta mantener los documentos más populares y reemplazar los menos referenciados. Sin embargo algunos documentos pueden alcanzar un alto contador de referencia y nunca ser referenciados de nuevo, es lo que se suele llamar polución de caché, ante la cual la estrategia LFU no provee ningún mecanismo de defensa.

2.3.3 GDS

Acrónimo inglés de Greedy Dual Size. Esta estrategia, así como (GDSF y GD*) forman parte de lo que se denomina políticas *"basadas en función"*, es decir, estrategias que emplean alguna función con la que establecer la posición de cada objeto en la caché.

Tal y como ocurría en las estrategias LFU y LFU-DA, los elementos dentro de la caché estarán ordenados, la diferencia estriba que el factor de ordenamiento. En este caso no será

el contador de referencia el elemento en cuestión, sino el valor arrojado por la función para cada objeto.

Algunas otras políticas (a partir de las ya citadas) que forman parte de este grupo son Server-assisted caché replacement, TSP (Taylor Series Prediction), Bolot/Hoschka's strategy, MIX, M-Metric, HYBRID, LRV, LUV, LR (Logistic Regressión-Model) (Podlipnig y Böszörmenyi, 1998).

Como sucede en los apartados anteriores, este grupo de políticas presenta una serie de ventajas y desventajas. Las ventajas más relevantes son:

- Consideran un número de factores para manejar diversas situaciones de carga de trabajo.
- No asumen una combinación fija de factores o un uso fijo de estructuras de datos. Con la opción apropiada de parámetros uno puede intentar optimizar cualquier comportamiento, es decir, con la función adecuada se puede optimizar su funcionamiento ante cualquier escenario de tráfico Web.

A continuación se mencionan las desventajas más notables:

- Elegir la función apropiada para cada caso es tarea difícil, esta se suele deducir de los estudios de las muestras Web, pero alcanzar la función idónea para cada carga de tráfico es realmente complejo; además el tráfico Web es muy variable con el tiempo, por lo tanto sería necesario una configuración adaptativa de los parámetros. Adaptación que por otro lado, sin llegar a solucionar por completo los problemas de tráfico variable, añaden una gran complejidad al proceso de reemplazo.
- El empleo de información de latencia para el cálculo de parámetros introduce grandes problemas. Cálculos relativos a la frecuencia de llamadas o cuan reciente se han referenciado ciertos objetos pueden ser calculados fáciles partiendo de las peticiones hechas a la caché. La latencia es calculada en la caché pero influenciada por muchos factores como son el camino entre el servidor y la caché. De tal manera que fluctuaciones en el cálculo de éste parámetro pueden complicar cualquier decisión tomada.

Seguidamente se presenta este algoritmo.

La estrategia GDS se emplea en cachés que mantienen sus objetos ordenados en función de un cierto factor que se denomina clave. Elementos con un campo clave más alto se ubican en la zona superior del sistema y aquellos con menor valor del factor clave se sitúan en la zona inferior, siendo los objetos a eliminar en caso de necesitar liberar espacio.

Pero, ¿cómo se calculará esta clave?, para ello se emplea la función mostrada en la ecuación:

$$\text{clave} = \left(\frac{\text{coste}}{\text{tamaño\_objeto}}\right)$$

La GDS que asigna una función de costo igual a 1 (se entiende por costo, al valor asociado que produzca un fallo de caché, es decir, que haya que traer el documento directamente del servidor), quedando el factor clave de la siguiente manera:

$$\text{clave} = \left(\frac{1}{\text{tamaño\_objeto}}\right)$$

Así, elementos de mayor tamaño generan factores claves menores y se insertan en la zona inferior de la caché, o lo que es lo mismo, esta es una política que beneficia a los objetos pequeños sobre los de mayor tamaño.

Debido a esto, cada vez que sea necesario liberar espacio, se eliminan los objetos de mayor tamaño, es decir, el resultado es una caché con un mayor número de objetos; aunque de menor tamaño. La conclusión principal que se desprende es que una función de costo igual a 1 podría conseguir el mejor HR, pero por el contrario, un BHR más bajo. En general lo que GDS busca es minimizar los fallos de caché.

A continuación se describe este funcionamiento básico a la hora de realizar una inserción, una eliminación, o al producirse un acierto:

- **Llega un elemento nuevo** → Se analiza si el nuevo elemento cabe en la caché. De ser así, se almacenará en la misma, de manera ordenada en función de su campo clave. Así elementos con una mayor clave (menor tamaño) se situarán siempre en la zona superior de la caché, mientras que elementos de menor clave (mayor tamaño)

se ubican en la zona inferior de la misma y son futuros candidatos para ser eliminados.

- **Llega un elemento nuevo que no cabe** → Se eliminan objetos de la caché, comenzando desde el último, hasta que el espacio liberado sea el suficiente como para insertar el nuevo elemento. Se toma la clave del último elemento eliminado, valor que será restado a las claves de todos los objetos almacenados en el sistema. Una vez recalculadas todas las claves en la caché, el nuevo elemento se introduce en la misma (en la posición que le corresponda en función de su clave).
- **Se produce un acierto en la caché** → O lo que es lo mismo, el documento solicitado está en la caché, y por lo tanto será servido por la misma. En tal caso se realizará una búsqueda de este objeto almacenado y una vez localizado (se consigue su índice) se restaura el valor de su clave y se reubica en función de ese nuevo valor.

Como resultado de este algoritmo la caché será ordenada en función del valor de las claves de cada uno de los elementos del sistema. El hecho de que tras una eliminación, se recalcule las claves de todos los elementos no es más que un mecanismo de envejecimiento de documentos.

A continuación se realizan dos pequeñas ilustraciones con las que se pretende alcanzar una total comprensión de cómo funciona el algoritmo.

	(1) – Inicialmente la caché estará vacía
OBJETO1 (0.1,10)	(2) - OBJETO 1 llega a la caché. Se sitúa en la cabeza
OBJETO1 (0.1,10)	(3) – OBJETO 2 llega a la caché. Se ubica de

OBJETO2 (0.05,20)	manera ordenada en la caché
OBJETO1(0.1,10) OBJETO3(0.08,12.5) OBJETO2(0.05,20)	(4) – OBJETO 3 llega a la caché. Se ubica de manera ordenada en la caché
OBJETO4 (0.2,5) OBJETO1(0.05,10) OBJETO3(0.03,12.5)	(5) – OBJETO 4 llega a la caché. No cabe. Se elimina el último objeto de la misma (OBJETO2). Se actualiza el valor de la clave del resto de elementos almacenados. Ya cabe. Se ubica de manera ordenada en la caché.
OBJETO4(0.2,5) OBJETO3(0.08,12.5) OBJETO1(0.05,10)	(6) – OBJETO 3 llega a la caché. Éste elemento ya está almacenado, es un acierto. Se restaura su clave al valor original (0.08). Se reubica de manera ordenada.

Como se observa, el algoritmo GDS gestiona la caché manteniendo los objetos ordenados en función del valor de su clave. Elementos de gran tamaño (clave pequeña) son más propensos a ser eliminados, es decir, se ubican en la zona inferior de la memoria, mientras que elementos de menor tamaño permanecen en la zona superior y su eliminación es mucho menos probable.

2.3.4 RAND

Esta política está ubicada en el marco de las *"políticas aleatorias"*, es decir, aquellas políticas que liberan espacio en la caché seleccionando documentos a eliminar de manera aleatoria.

Este conjunto de estrategias intentan reducir la complejidad de los procesos de reemplazo, intentando no sacrificar en demasía el rendimiento de los mismos. Como sucede en anteriores apartados, en este grupo se engloban otras muchas estrategias, cada una de las cuales se detallan con mayor atención en apartados posteriores.

Es evidente intuir que este grupo de políticas presentan una serie de ventajas e inconvenientes. Las ventajas más relevantes son las siguientes:

- Las estrategias aleatorias no necesitan estructuras de datos especiales para insertar o borrar documentos.
- Son estrategias muy simples de implementar.

La principal desventaja sería la siguiente:

- Son estrategias mucho más difíciles de evaluar ya que diversas simulaciones con una misma muestra de entrada arrojarían resultados muy diversos.

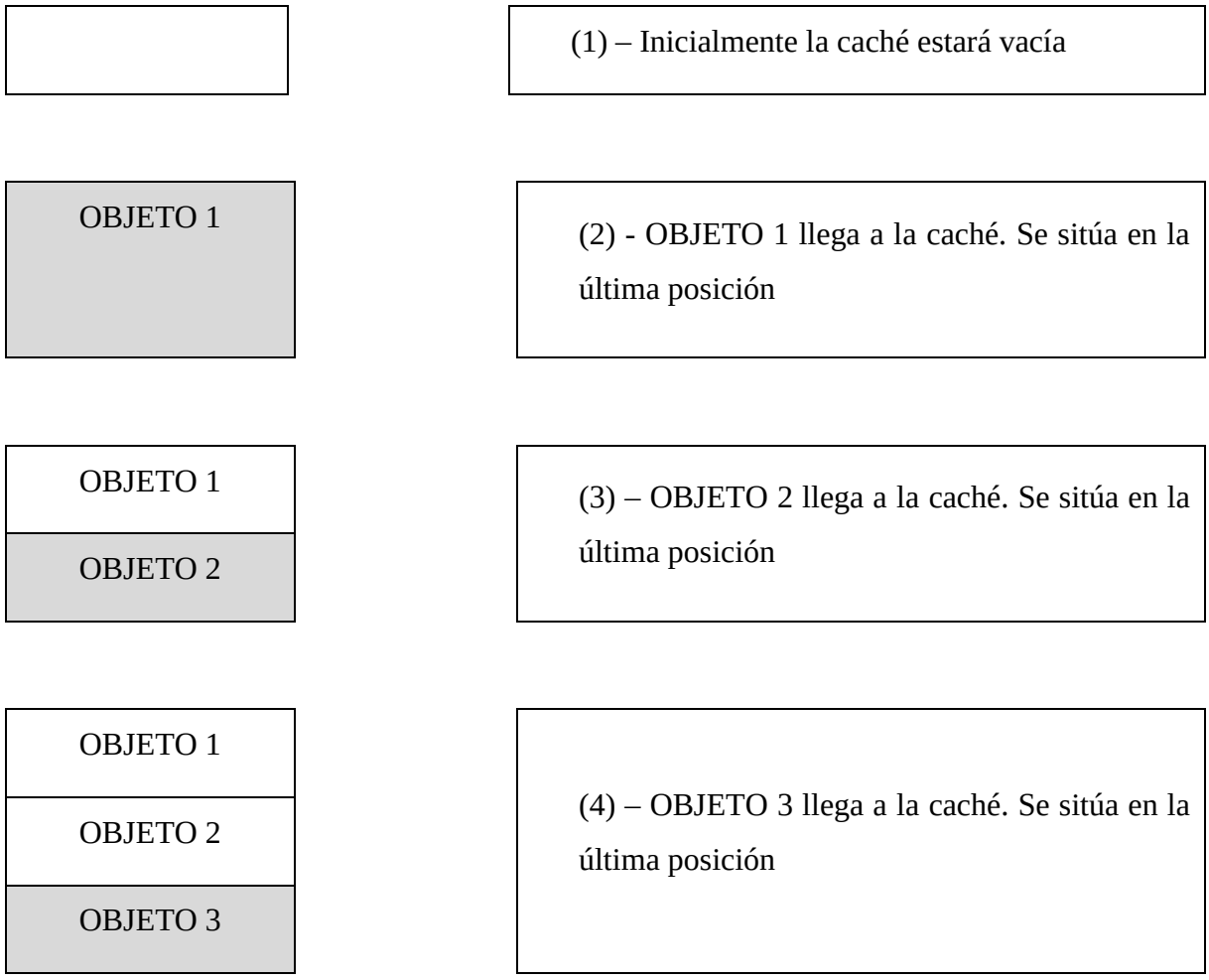
El funcionamiento de este algoritmo es tremendamente simple, en caso de ser necesario eliminar documentos, estos se seleccionarán de forma aleatoria. Dado que el algoritmo elimina objetos aleatoriamente, la forma en que éstos serán insertados no es relevante ya que los objetos tendrán la misma probabilidad de ser eliminados sea cual sea la posición en la que se encuentren.

De esta manera se obtiene una caché que no está ordenada en ningún modo. Claro que en este caso el campo clave no tiene sentido y no es calculado. Por comodidad y para establecer un criterio común a todas las políticas aleatorias se considera que la inserción se realiza siempre al final de la caché (en este caso los elementos acertados no se reubican).

Seguidamente se procede a la presentación del algoritmo en cuestión:

- **Llega un elemento nuevo** → Se analiza si el nuevo elemento cabe en la caché. De ser así, se almacenará en la cabeza de la caché (es decir, en la posición superior de la misma). El resto de los elementos almacenados serán desplazados una posición.
- **Llega un elemento nuevo que no cabe** → Se eliminan objetos de la caché hasta que el espacio liberado sea suficiente como para insertar el nuevo elemento. Los objetos a eliminar se elegirán de manera aleatoria.
- **Se produce un acierto en la caché** → O lo que es lo mismo, el documento solicitado está en la caché y por tanto será servido por la misma. En tal caso se deja en la posición en que estaba (ya que no tiene ningún sentido reubicarlo).

A continuación se realiza una pequeña demostración (la caché originariamente estará vacía):



OBJETO 1
OBJETO 2
OBJETO 4

(5) – OBJETO 4 llega a la caché. No cabe. Se elimina un documento aleatoriamente (OBJETO3). Ya cabe. Se inserta en la última posición.

OBJETO 2
OBJETO 4
OBJETO 3

(6) – OBJETO 3 llega a la caché. No cabe. Se elimina un documento aleatoriamente (OBJETO1). Ya cabe. Se sitúa en la cabeza

En resumen, esta es la política aleatoria de mayor sencillez, yconsiste básicamente (siempre que sea necesario) en eliminar objetos de la caché seleccionándolos de manera aleatoria.

2.3.5 FIFO

Esta política toma como parámetro para eliminar un objeto la edad del objeto en la caché, mientras más tiempo tenga en la caché mayor es su probabilidad de ser eliminado.

Las ventajas más relevantes son las siguientes:

- es fácil de implementar.
- permite predecir el orden de ejecución de los procesos.

Por el contrario tiene como desventajas las siguientes:

- bajos rendimientos, no es útil en sistemas interactivos puesto que no garantiza buenos tiempos de respuesta.

Las principales características de este algoritmo son:

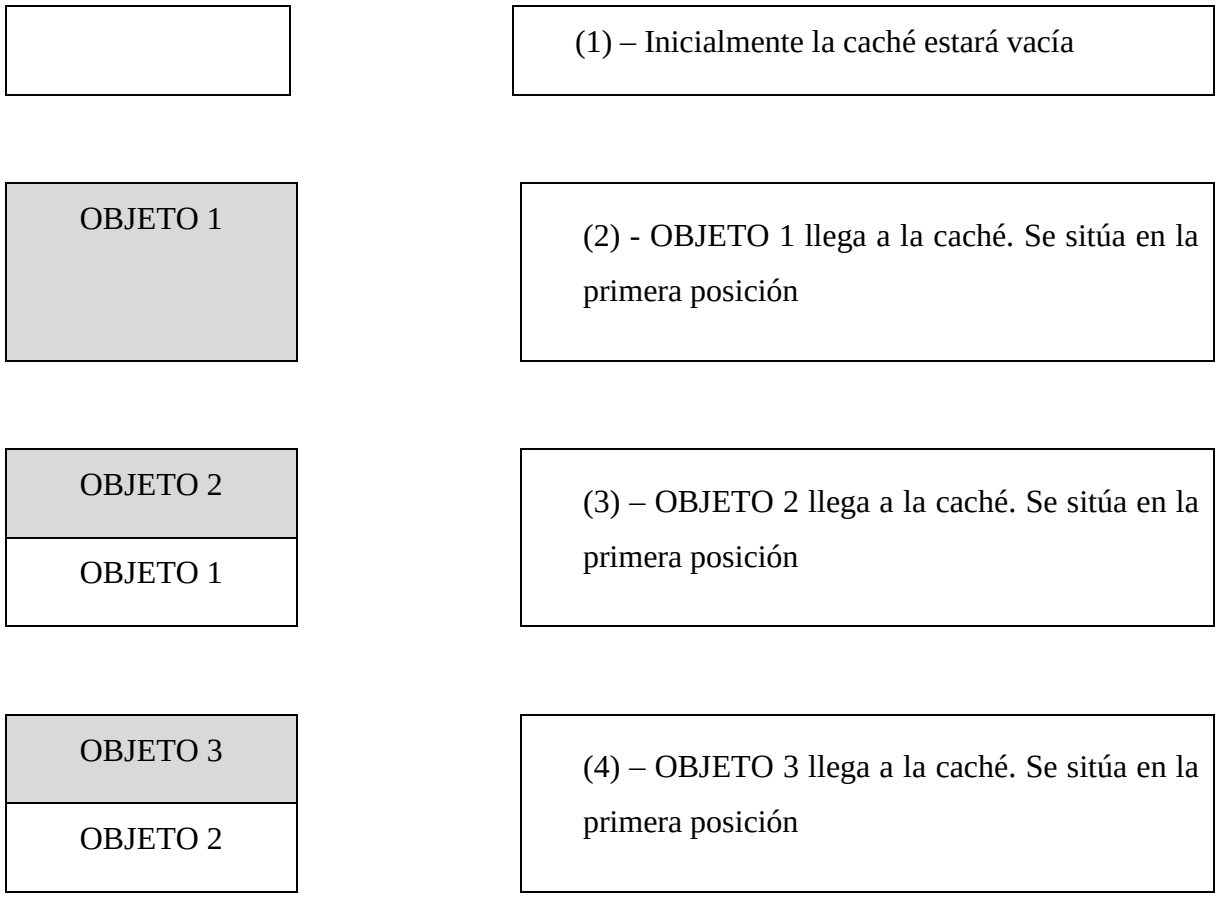
1)**Llega un elemento nuevo** → Se analiza si el nuevo elemento cabe en la caché. De ser así se almacenará en la cabeza de la caché (es decir, en la posición superior de la misma). El resto de elementos almacenados serán desplazados una posición.

2)**Llega un elemento nuevo que no cabe** → Se eliminan objetos de la caché hasta que el espacio liberado sea el suficiente como para insertar el nuevo elemento. Los objetos a eliminar son los que tienen más tiempo en la caché.

3)**Se produce un acierto en la caché** → O lo que es lo mismo, el documento solicitado está en la caché y por tanto será servido por la misma. Se reubica de nuevo en la cabeza del sistema (se inserta como si se tratase de un elemento nuevo).

A continuación se realiza una pequeña demostración (la caché originariamente estará vacía):

Caché del proxy



OBJETO 1

OBJETO 4
OBJETO 3
OBJETO 2

(5) – OBJETO 4 llega a la caché. No cabe. Se elimina (OBJETO1). Ya cabe. Se inserta en la primera posición.

OBJETO 3
OBJETO 4
OBJETO 2

(6) – OBJETO 3 llega a la caché. Este elemento ya está almacenado, es un acierto. Se reubica en la cabeza

2.4 Conclusiones del capítulo 2

El estudio realizado en este capítulo sobre el funcionamiento de las políticas de reemplazo de caché en los servidores proxy ha permitirá estimar y evaluar la influencia de este proceso en el tiempo de respuesta total percibido por el usuario en el acceso Web. Servirá además para seleccionar la política de remplazo más adecuadas de acuerdo a las características del tráfico Web asociado a una entidad particular. Se ha desarrollado un modelo de teletráfico como un sistema integral que caracteriza el acceso a Internet a través de servidor de proxy teniendo en cuenta todos los nodos que participan en el acceso y que introducen demoras en el servicio.

CAPÍTULO 3. EVALUACIÓN DEL RENDIMIENTO DE LAS POLÍTICAS DE REEMPLAZO EN CACHÉ

El presente capítulo realiza una serie de simulaciones y evaluaciones del tiempo de respuesta basada en el modelo analítico desarrollado en el capítulo 2. También se realiza una comparación detallada del rendimiento de las diversas políticas de remplazo implementadas. Para ello, se ha utilizado la misma muestra de tráfico de entrada para todas las políticas y se han comparado los resultados conseguidos ante diversos tamaños de caché. De acuerdo a los resultados obtenidos en cada política, se arriba a una serie de conclusiones, es decir, se realiza una estimación del escenario más conveniente para cada una de las estrategias.

3.1 La herramienta WebTraff para la evaluación del rendimiento de las políticas de reemplazo en caché.

La figura 3.1 muestra una captura de la pantalla de la interfaz gráfica de usuario de WebTraff. La interfaz tiene tres bloques principales, correspondientes a la generación de la carga de trabajo (parte superior), el análisis de la carga de trabajo (parte central) y la simulación de la caché del proxy Web (parta inferior). Las siguientes secciones proporcionan más detalles sobre la funcionalidad proporcionada en cada uno de estos tres bloques.

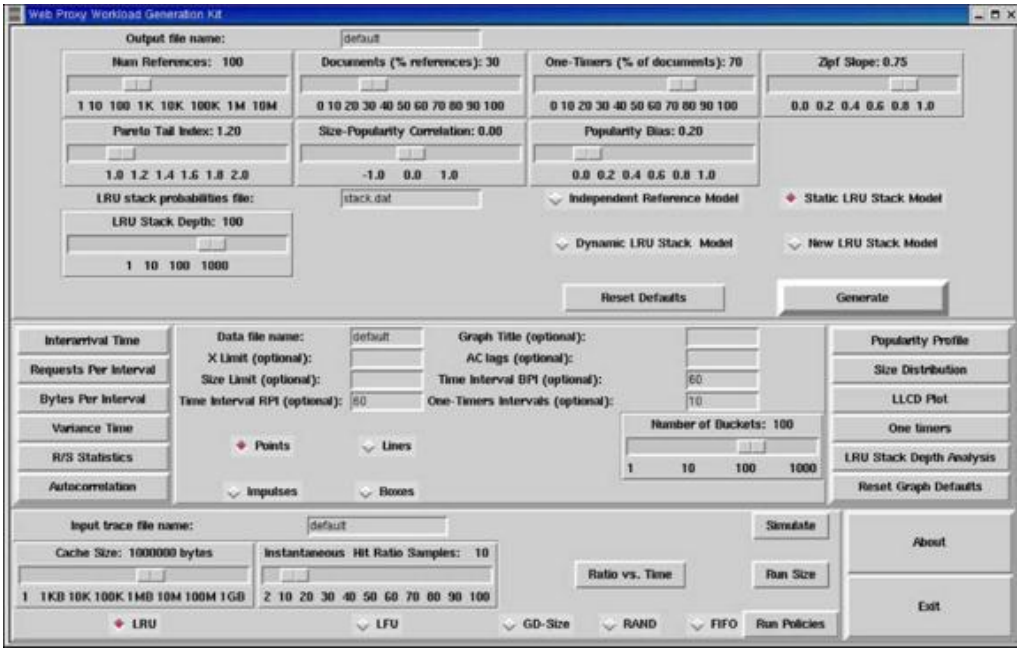


Figura 3.1: Interfaz gráfica de WebTraff.

3.1.1 Formato de la carga de trabajo para el servicio Web

TIMESTAMP	DOC_ID	SIZE
0.03245	0	1958
2.73954	9	366
3.47710	4	2536
3.57192	0	1958
4.61692	26	82906
4.68677	3	3413
5.11075	21	3257
8.97064	4	2536
9.63967	5	2914
11.23907	23	149
11.30097	2	38735
12.89020	0	1958
13.86087	9	366

Figura 3.2: Formato de carga de trabajo.

3.1.2 Carga de trabajo generada para el servicio Web

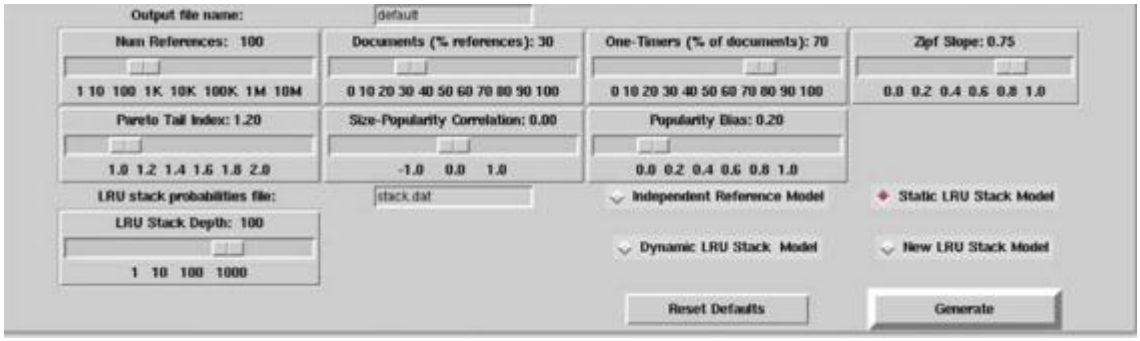


Figura 3.3: Bloque de generación de carga de trabajo Web.

El bloque superior de la interfaz gráfica de usuario del WebTraff es para la generación de las trazas de la carga de trabajo de proxy Web. Esta parte de la herramienta, simplemente proporciona una interfaz gráfica ProWGen (Busari y Williamson, 2002), que es una herramienta de generación de carga de trabajo de proxy Web desarrollada en trabajos anteriores. ProWGen es un generador de carga de trabajo sintética Web para proxy (Crovella y Bestavros, 1997; Leland et al., 1994) para la evaluación de las técnicas de almacenamiento en caché del proxy. Hay varias herramientas de generación de carga de trabajo desarrolladas para el estudio de los servidores Web y servidores proxy. ProWGen es el único con capacidad para generar cargas de trabajo de proxy que difieren en una característica cada vez. ProWGen incorpora cinco características de carga seleccionadas: referencia de una vez, la popularidad de archivo, distribución del tamaño de archivo, la correlación entre el tamaño del archivo y la popularidad; y la localidad temporal. A continuación se describe cada una de estas características y la forma en que se modelan en ProWGen.

Referencia de una vez: Estudios de servidor Web y cargas de trabajo de los proxys Web han demostrado que muchos de los documentos solicitados desde un servidor proxy se piden una sola vez (Abdulla et al., 1998; Arlitt y Williamson, 1997a; Mahanti et al., 2000). Estos documentos se refieren como una sola vez (del inglés one-timers) en la literatura. En

los logs de acceso del proxy, los one-timers pueden dar cuenta de un 50-70% de los documentos. En ProWGen el usuario especifica el porcentaje de una sola vez que desee en la carga de trabajo como porcentaje de archivos diferentes. Los resultados de la simulación descritos en (Busari y Williamson, 2001) muestran que el porcentaje de una sola vez tiene poco impacto en los resultados de rendimiento del almacenamiento en caché.

Popularidad de archivo: Las cargas de trabajo sintético producidas por el ProWGen modelan las distribuciones Zipf para la popularidad de archivo (Li, 2000). La ley que sigue la distribución Zipf expresa una relación de potencias real entre la popularidad P de un elemento, es decir, la frecuencia de su ocurrencia, y su rango r, es decir, el rango relativo entre los elementos de referencia basado en la frecuencia de ocurrencia. Esta relación es de la forma $P = C/r^{\beta}$, donde C es una constante, y β es usualmente cercano a la unidad. Los investigadores han encontrado un comportamiento similar de referencia en la Web (Roadknight et al., 2000; Mahanti et al., 2000; Arlitt et al., 1999). Algunos investigadores encontraron que el valor de β es cercano a 1 (Almeida et al., 1996a; Carlos et al., 1995) precisamente después de la ley de Zipf. Otros (Almeida et al., 1998; Breslau et al., 1999; Mahanti et al., 2000) encontraron que el valor de β es menor que 1 y que la distribución puede ser descrita sólo como "Zipf-like" con el valor de β variando traza en traza. En ProWGen la cuenta de referencia de todos los archivos, a excepción de los *one-timers*, se determina a partir de una distribución de Zipf-like. El valor de β suministrado al generador de carga de trabajo de referencia generado seguirá la distribución de Zipf perfecta ($\beta = 1$) o no ($0 < \beta < 1$). En los experimentos se utilizó $\beta = 0{,}75$.

Distribución del tamaño de archivo: Estudios de caracterización de la carga de trabajo han demostrado que la distribución de tamaño de archivo para las transferencias Web es una cola pesada (Abdulla et al., 1998; Arlitt y Jin, 2000; Arlitt y Williamson, 1997a; Duska et al., 1997). Una distribución de cola pesada implica que de forma relativa pocos archivos de gran tamaño representan un porcentaje significativo del volumen de datos transferidos a los clientes Web. En ProWGen los archivos se dividen en dos grupos, aquellos que se consideran en el cuerpo de la distribución y los de la cola. El cuerpo de la distribución de

tamaño de archivo se modela como una distribución logarítmica normal y la cola sigue el modelo con una distribución de Pareto. La distribución de Pareto doblemente exponencial es un ejemplo de una distribución de cola pesada que se ha utilizado para modelar la cola de la distribución de tamaño de archivo en (Barford y Crovella, 1998; Crovella y Bestavros, 1997; Crovella y Taqqu, 1999).

Correlación entre el tamaño y la popularidad del archivo: Algunos estudios han demostrado que existe poca o ninguna correlación entre la frecuencia de acceso a un archivo y su tamaño (Breslau et al., 1999; Mahanti et al., 2000). Para un modelado flexible, ProWGen permite la generación de cargas de trabajo que presentan una correlación positiva, negativa o cero entre el tamaño del archivo y la popularidad del mismo. La correlación positiva significa que los archivos grandes tienen más probabilidades de ser solicitados, y la correlación negativa significa que los archivos pequeños son más propensos a ser solicitados. En estos experimentos se utilizó la correlación cero.

Localidad temporal: La localidad temporal se refiere a la tendencia de los documentos Web referenciados en el pasado reciente a ser referenciados en el futuro próximo. Esta tendencia afecta el orden relativo en que las solicitudes de archivo aparecen en el flujo de referencias resultante. El enfoque utilizado en ProWGen para modelar la localidad temporal se basa el modelo de pila LRU de tamaño finito (Almeida et al., 1996a; Busari, 2000). Una pila LRU es un ordenamiento de todos los archivos por lo reciente de su referencia, de manera que el archivo más recientemente referenciado se encuentra en la parte superior de la pila y el archivo referenciado menos recientemente se encuentra en la parte inferior de la pila. La pila se actualiza dinámicamente a medida que cada referencia se procesa. En algunos casos, esta actualización incluye la adición de un nuevo elemento a la parte superior de la pila mientras empuja hacia abajo los demás, en otros casos, se trata de extraer un elemento existente en el medio de la pila y moverla a la parte superior de nuevo, empujando a otros hacia abajo según sea necesario.

La parte superior de la interfaz gráfica de usuario permite establecer parámetros para controlar las características de la carga de trabajo antes de pulsar el botón "Generar". Un

cuadro de entrada en la parte superior de la interfaz gráfica de usuario especifica el nombre del archivo de traza a generar. Controles de escala deslizables se utilizan para especificar el número de referencias (solicitudes) que se desea en el volumen de trabajo generado, así como el número de objetos Web distintos (expresado como porcentaje de las referencias totales) y el número de una sola vez (expresados como un porcentaje del total de documentos). Reguladores independientes se utilizan para especificar la pendiente de la Zipf-like, la distribución de popularidad de este documento, la pendiente de la cola de Pareto de la distribución de tamaño del documento (el cuerpo de la distribución de tamaño del documento es log-normal, con una mediana de 4 KB y una media de 10 KB) y el grado de correlación estadística (si lo hay) entre el tamaño y la popularidad de los objetos Web. La correlación positiva significa que objetos de mayor tamaño tienen más probabilidades de ser referenciados y la correlación negativa que los objetos más pequeños tienen más probabilidades de ser referenciados. La configuración por defecto de la correlación cero significa que el tamaño del documento y su popularidad son independientes.

3.1.3 Análisis de la carga de trabajo Web.

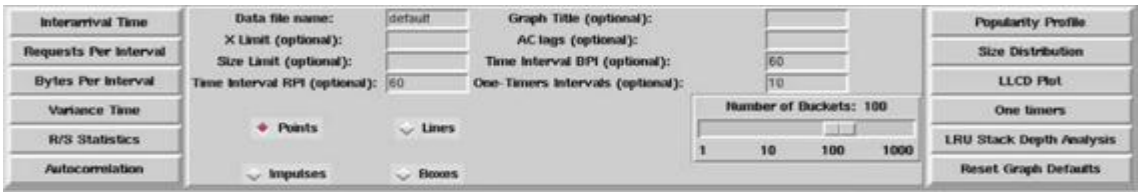


Figura 3.4: Bloque de análisis de carga de trabajo Web.

El bloque central de la interfaz gráfica de usuario del WebTraff es para el análisis de la carga de trabajo Web. Las funciones de análisis se dividen en dos categorías principales: análisis de series temporales (en el extremo izquierdo del bloque del centro), y el análisis de carga de trabajo Web (en el borde más a la derecha del bloque central). El usuario debe especificar el nombre del archivo de carga de trabajo a analizar. El nombre de archivo predeterminado es el mismo que el del último archivo de carga de trabajo generado.

Las herramientas de análisis de las series de tiempo trabajan sobre todo con la primera columna (timestamp) del formato de la carga de trabajo Web, es decir, los tiempos denominados time-stamp (ver Figura 3.2). Los botones aquí producen gráficos de la distribución del arribo de las solicitudes, el número de solicitudes por intervalo de tiempo

(especificado por el usuario), y el número de bytes por intervalo de tiempo (especificado por el usuario). Los botones adicionales proporcionan pruebas de dependencia a largo plazo (LRD) en el proceso de arribos de solicitud (es decir, la función de auto correlación, de varianza-tiempo y R/S) (Busari y Williamson, 2001; Cherkasova, 1998). Estos análisis son similares a los incluidos en el SynTraffGUI desarrollado previamente para el modelado y análisis de tráfico LRD (Arlitt y Jin, 2000).

Los botones de análisis de carga de trabajo Web funcionan sobre todo con la segunda y tercera columna de datos, en lo relacionado con la identificación del documento y el tamaño. Aquí se analiza la distribución del tamaño del documento, la distribución del tamaño de la transferencia y la cola de las distribuciones de tamaño (con un trazado de distribución log-log complementaria (LLCD)). El último análisis es útil para probar una distribución de cola pesada para el tamaño de los documentos (Busari, 2000). Dos botones adicionales producen análisis del perfil de la popularidad del documento (para comprobar si hay un perfil de popularidad Zipf-like) y el comportamiento de la URL en la pila haciendo referencia a profundidad (para caracterizar las propiedades de localidad temporales).

3.1.4 Simulación del servidor Web proxy con caché.

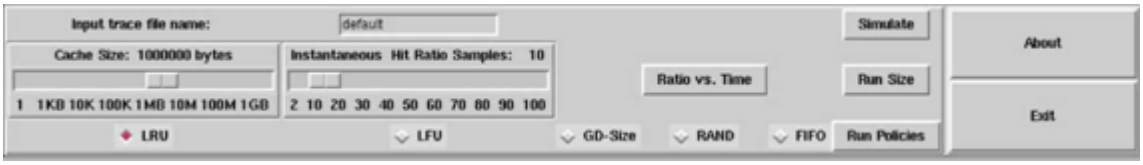


Figura 3.5: Bloque de simulación de Web proxy caché.

El bloque inferior de la interfaz gráfica de usuario en WebTraff es para el estudio de los resultados de la simulación, evaluando el rendimiento del proxy caché. El modelo utilizado para simular la caché Web del proxy es un modelo simple a nivel de aplicación para el almacenamiento en caché. Dos parámetros son necesarios: el tamaño de la caché y la política de reemplazo de caché que se utilizará para eliminar los documentos cuando se requiere más espacio para guardar un documento de entrada (por ejemplo, la política RANDOM, la política FIFO, LRU, LFU) (Almeida et al., 1996a). El usuario especifica estos dos parámetros, así como el nombre del archivo de proxy Web de carga de trabajo que se utiliza como entrada para la simulación.

Los botones restantes se asocian a diferentes tipos de experimentos de simulación. El botón "simular" ejecuta una sola simulación con el tamaño de caché especificado por el usuario y la política de sustitución. Como efecto secundario, produce un archivo con la secuencia de fallos de caché durante la simulación. El botón “tamaño de ejecución” ejecuta un conjunto de simulaciones para la política de sustitución en caché seleccionada utilizando una gama de tamaños de caché especificada por el usuario. Como resultado se genera una gráfica que muestra la proporción de aciertos de documentos (HR) y la proporción de aciertos de byte (BHR) en función del tamaño de la caché. El botón "Ejecutar políticas" ejecuta un conjunto de simulaciones variando las políticas de reemplazo de la caché así como el tamaño de la misma. Dos gráficos se crean para comparar el rendimiento de la política de sustitución: uno para la HR y el otro para el BHR. Por último, el botón de “proporción de aciertos vs tiempo” ofrece un aspecto visual en la estacionalidad (o no-estacionalidad) de la proporción de aciertos y la proporción de aciertos de byte en función del tiempo de simulación dentro de la traza. Este gráfico es útil para identificar los efectos de pre-calentamiento y las anomalías en la carga de trabajo configurada. Este botón utiliza la política de reemplazo LRU solamente, y una gama de tamaños de caché hasta el máximo tamaño de caché especificado por el usuario.

3.2 Resultados de las evaluaciones del rendimiento de las políticas de reemplazo en caché y los efectos de las métricas en el tiempo de respuesta.

3.2.1 Comparación de las políticas de reemplazo en caché

Se realizaron diferentes series de experimentos para poner a prueba el rendimiento de las diferentes políticas de reemplazo que han sido estudiadas en este trabajo. Primeramente se diseño un modelo de evaluación de pruebas donde se incluyen cada una de las políticas y se especifican los datos de cada una.

El modelo de evaluación utiliza los siguientes valores:

- Número total de petición: 500000.
- Número total de documentos solicitados: 30% de las peticiones totales = 150000.
- Número total de archivos de una sola vez: 75% de los archivos solicitados = 125000.
- Índice de cola Pareto: 1.20.

- Zipf slope: 0.75.
- Correlación entre tamaño y popularidad: 0.
- Tamaño de la caché: 1024MB

Experimento 1:

El primer experimento realizó una corrida de simulación para todas las peticiones, repitiéndose el experimento para cada una de las políticas de reemplazo. La figura 3.6 muestra los resultados obtenidos. Para cada una de las políticas de reemplazo se midió el valor de la razón de aciertos y la razón de aciertos de bytes.

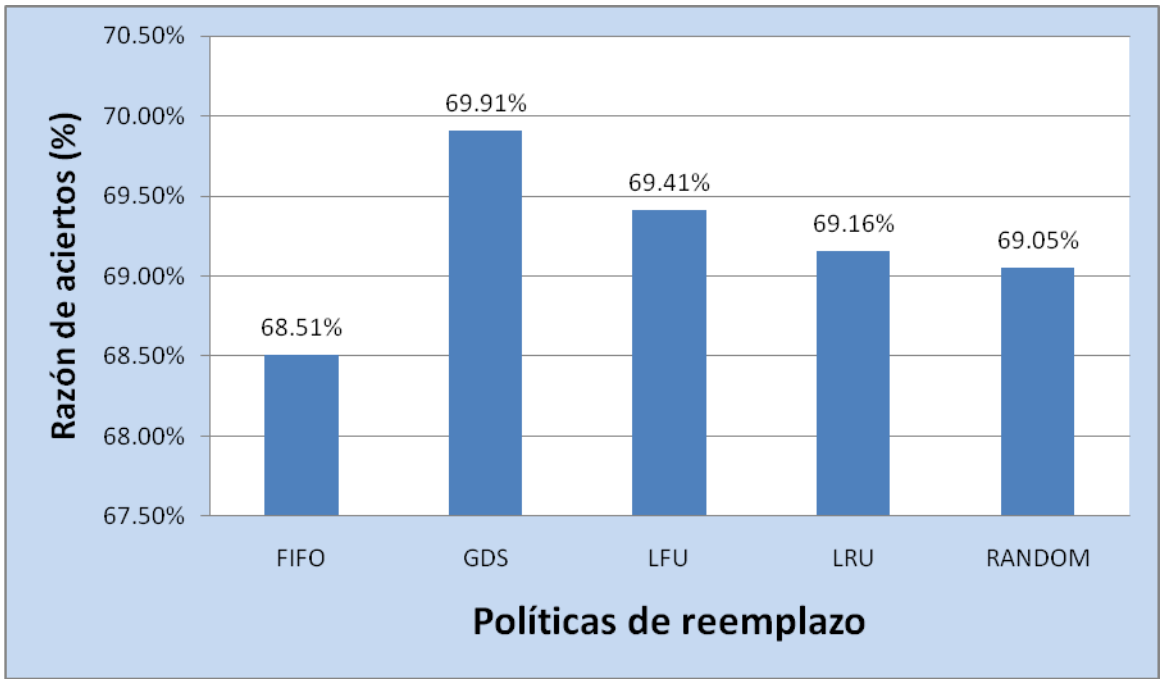


Figura 3.6 Razón de aciertos

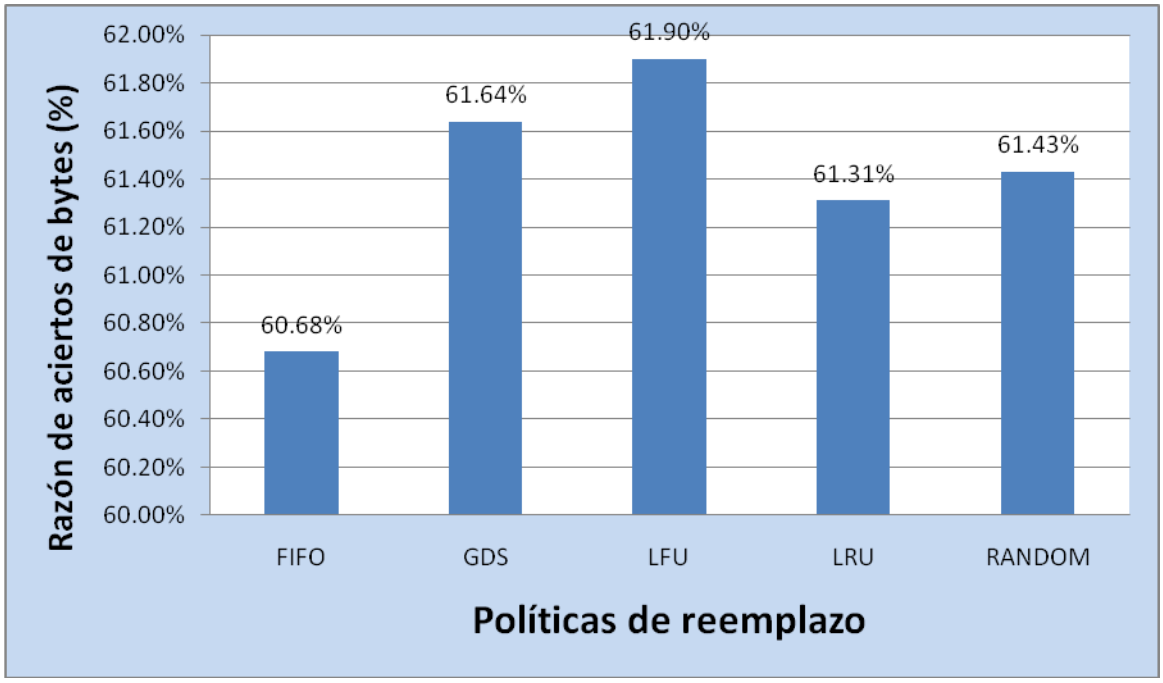


Figura 3.7: Razón de aciertos de bytes

Como se puede observar en las figuras 3.6 y 3.7, las políticas consiguen resultados muy similares para un tamaño de la caché igual a 1 GB. Las políticas RAND y FIFO obtienen siempre los peores resultados debido a que seleccionan los documentos a eliminar de manera aleatoria y lineal respectivamente, sin seguir ningún otro criterio de selección que favorezca la obtención de mejores resultados.

Se puede observar con la política de reemplazo de caché GDS, tal y como se comenta en el capítulo 2, que el hecho de emplear una función coste constante e igual a 1 parece conseguir mejores tasas de HR (los elementos más propensos a ser eliminados serán los de mayor tamaño, por lo que se almacenarán más elementos aunque de menor tamaño) a costa de perjudicar el BHR. Si se revisa detenidamente la figura 3.6 se puede observar como es la política GDS la que consigue un mayor HR, lo cual se puede explicar por el hecho de que esta política elimina con mayor probabilidad los objetos de mayor tamaño, que no favorecen la obtención de un mayor BHR. Al tratar el BHR se produce el efecto contrario, la política GDS al eliminar los objetos de mayor tamaño, será la política que consiga un menor BHR. Pero los resultados del BHR en GDS y LFU son muy parecidos. Por lo tanto se considera que GDS es la mejor estrategia de mejora en cuanto al tiempo de respuesta.

Experimento 2:

El segundo experimento consistió en simular todas las peticiones utilizando solamente la política de reemplazo GDS ya que es considerada como una de las mejores, sino la mejor, políticas de reemplazo en cuanto al uso de caché. Se realizaron pruebas para diferentes tamaños de caché: 4MB, 16MB, 64MB, 256MB y 1024Mb; esto se hizo para poder mostrar el comportamiento de una política de reemplazo con diferentes tamaños de caché que es uno de los parámetros más importantes al implementar un proxy caché.

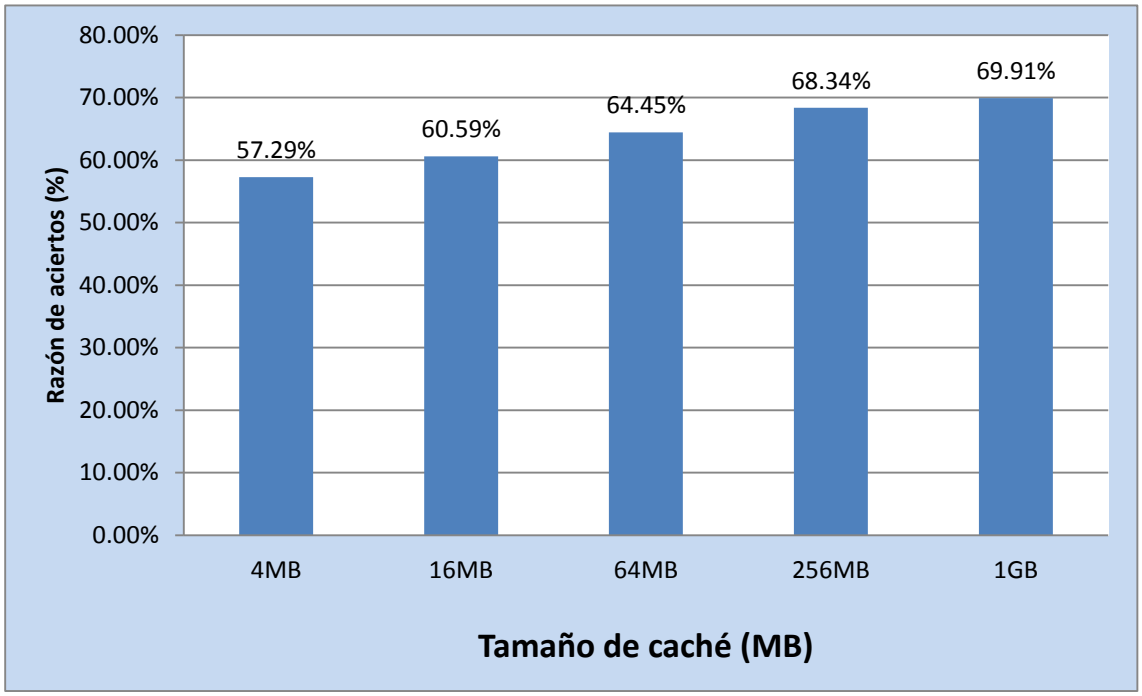


Figura 3.8: GDS – Razón de aciertos con diferentes tamaño de caché.

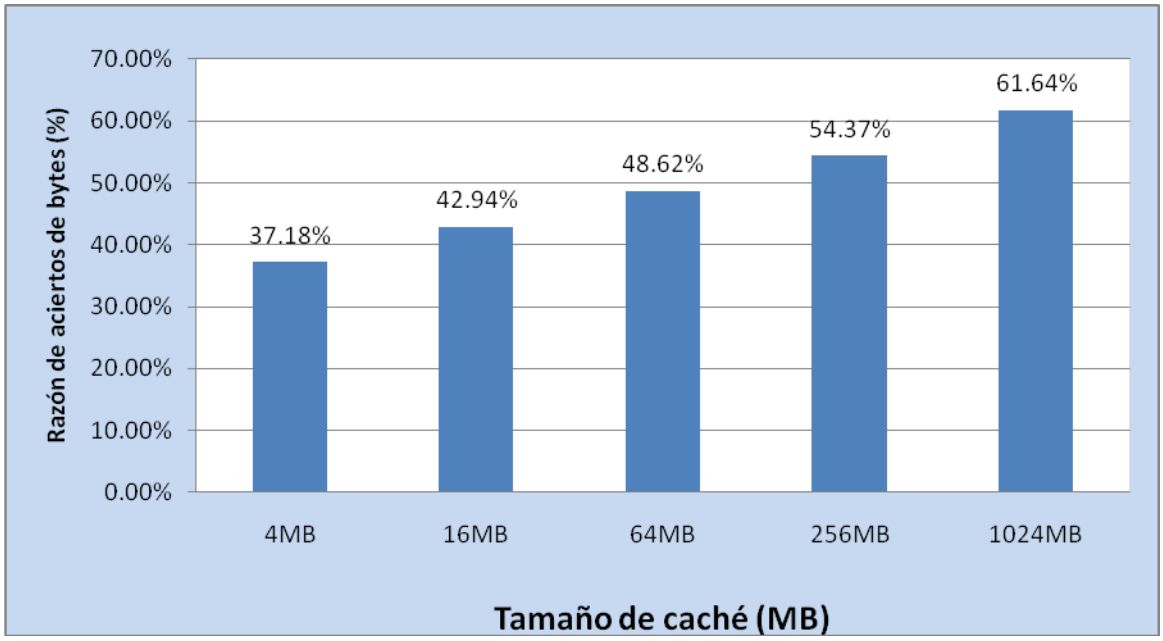


Figura 3.9: GDS – Razón de aciertos de bytes con diferentes tamaño de caché.

3.2.2 Efecto de las métricas en el tiempo de respuesta

3.2.2.1 Efecto de la probabilidad de acierto en caché "Hit Rate", p

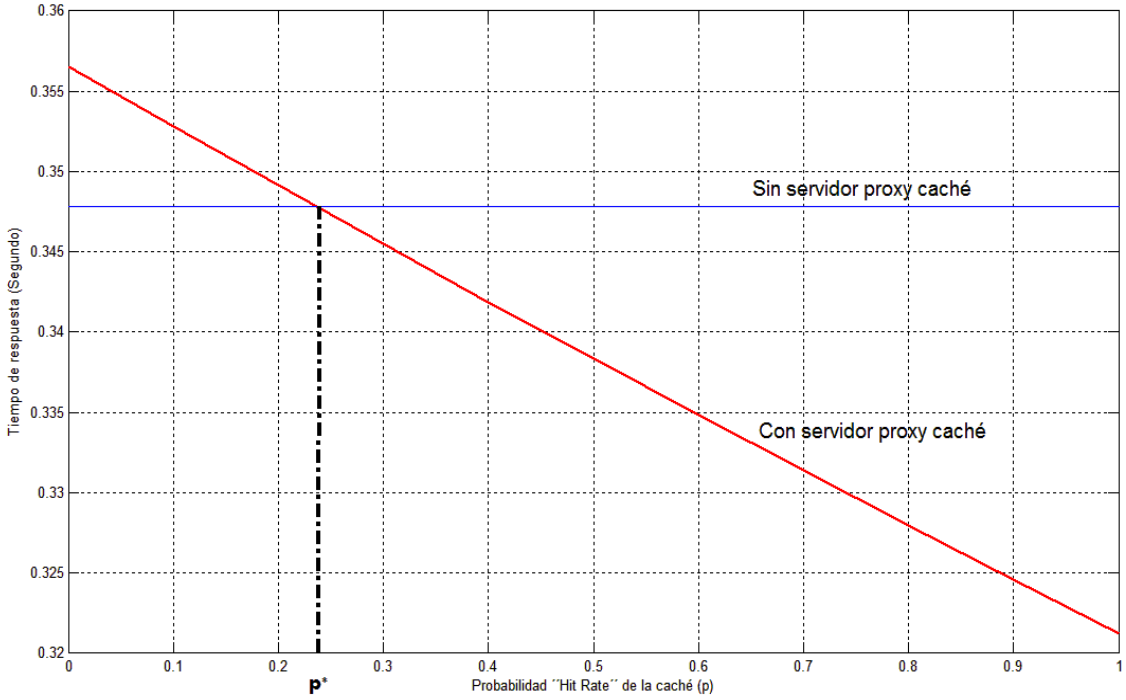


Figura 3.10: Efecto de la probabilidad p.

Varios experimentos numéricos se llevaron a cabo para examinar el efecto de la probabilidad "Hit Rate – razón de aciertos" de caché sobre el tiempo de respuesta global, donde la tasa de arribos λ se establece en 20 peticiones/s y el tamaño medio de los ficheros F se establece en 5000 bytes. La probabilidad p se varió desde 0.0 hasta 1.0 en incrementos de 0,1. El tiempo de respuesta global sin un servidor de caché es independiente de p, como se observa de la ecuación (3). Por lo tanto, T = 0.3478 s para todos los valores de p. El tiempo de respuesta general con un servidor proxy caché disminuye a medida que p aumenta como se ve en la figura. 3.1.

Hay un cruce entre los gráficos que representan T y T_{xc}. Este punto de cruce da la probabilidad correspondiente "*probabilidad de cruce*" que se define como p*. Esta probabilidad de cruce es significativa en la decisión si desea instalar un PCS. Para cualquier razón de aciertos de caché mayor que la probabilidad de cruce, p > p*, se obtienen beneficios de un servidor proxy caché ya que el tiempo total de respuesta a las peticiones de los usuarios Web se reduce.

3.2.2.2 Efecto de la razón de arribos λ

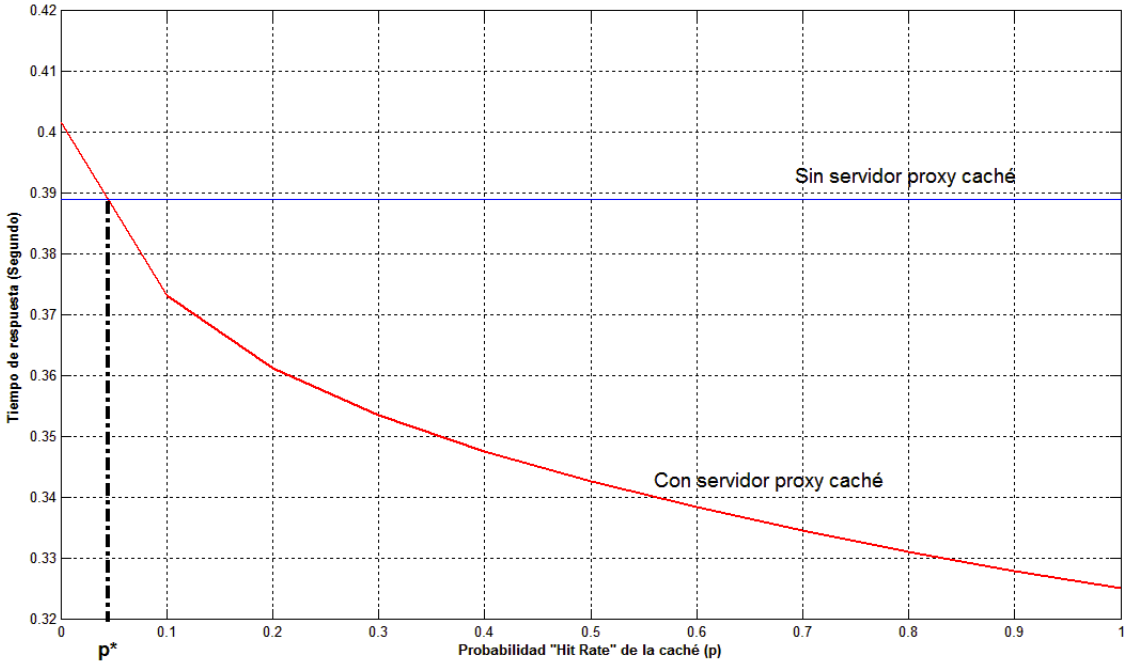


Figura 3.11: Efecto de la razón de arribos **λ.**

El efecto de una alta razón de arribos de las solicitudes Web en el tiempo de respuesta global se muestra en la figura. 3.2, donde la la razón de arribos se incrementó hasta 90 solicitudes/s manteniendo todos los demás parámetros fijos. La Figura. 3.11 muestra que con una alta razón de arribos, los tiempos de respuesta sufre por los dos casos, con y sin servidor proxy caché. La probabilidad de cruce p*, sin embargo, disminuye a medida que la tasa de llegada, A, aumenta. Las figuras. 3.1 y 3.2 muestran que p* = 0.238, cuando la tasa de llegada es de 20 peticiones/s, y p* se reduce a 0.030 cuando la tasa de llegada es 90 peticiones/s. El servidor proxy caché es más beneficioso en el caso de tráfico pesado.

3.2.2.3 Efecto del tamaño promedio de los archivos solicitados, F

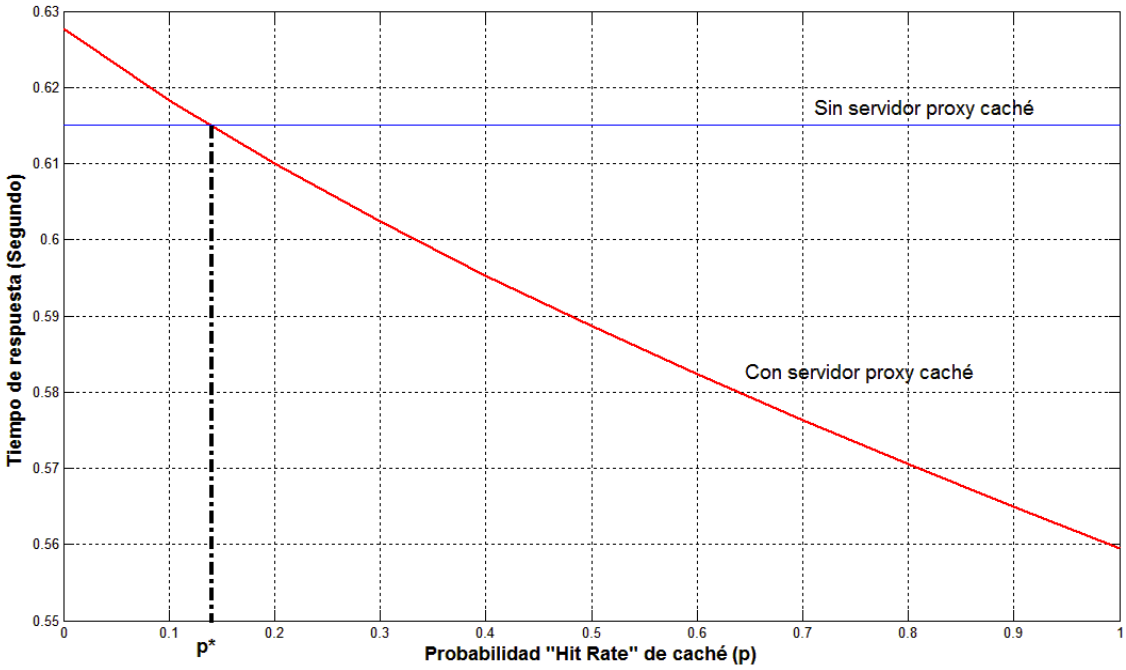


Figura 3.12: Efecto del tamaño promedio de los archivos solicitados F.

En la figura. 3.12, se grafica el tiempo de respuesta global, T_{xc} y T, con respecto a la probabilidad de razón de aciertos de caché cuando el tamaño promedio de archivo es grande. La razón de arribos es de 20 peticiones/s y el tamaño promedio del archivo

solicitado es igual a 8750 bytes. La probabilidad de cruce, p*, se encuentra en 0.148, lo que implica que el tiempo de respuesta global con un servidor proxy caché es menor cuando la probabilidad de que los archivos solicitados se almacenan en el servidor proxy caché supera los 0.148.

3.2.2.4 Efecto de los parámetros del servidor

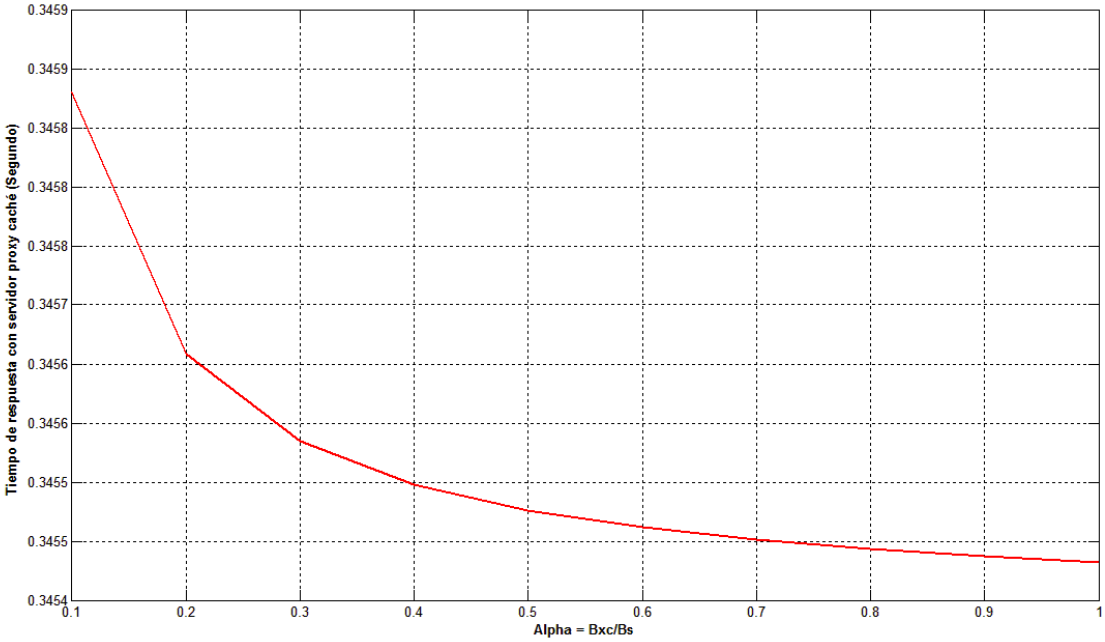


Figura 3.13: Efecto del parámetro α.

En los experimentos anteriores, donde se ha asumido de forma estratégica que el PCS es idéntico al servidor Web remoto, quedando los parámetros Bx =Bs, Rx =Rs, Yx =Ys and I_x =I_s. Se han asumido servidores idénticos con el objetivo aislar y estudiar el efecto de la razón de arribo de solicitudes, el tamaño promedio de archivos y la probabilidad de la razón de aciertos en caché sobre el tiempo de respuesta global en la presencia y ausencia de un servidor proxy caché. Sin embargo, no es necesariamente cierto que el servidor proxy caché sea idéntico al servidor Web remoto. En esta sección se estudia el efecto de los parámetros del servidor proxy caché en el comportamiento del tiempo de respuesta global. Se supone que $\alpha=B_{xc}/B_s$, $\beta = R_{xc}/R_s = Y_{xc}/Y_s$ y $\gamma= I_{xc}/I_s$. En la figura 3.13 se muestra el comportamiento de Txc con respecto a α manteniendo β = γ = 1. La razón de arribos se fija en λ = 20 peticiones/s y el tamaño de archivos se mantiene en F = 5000 bytes. A medida

que se aumenta la relación de búfer de salida, el tiempo medio de respuesta disminuye. Por lo tanto, al aumentar el tamaño del búfer de salida del servidor proxy caché se reduce el tiempo de entrega de los archivos solicitados.

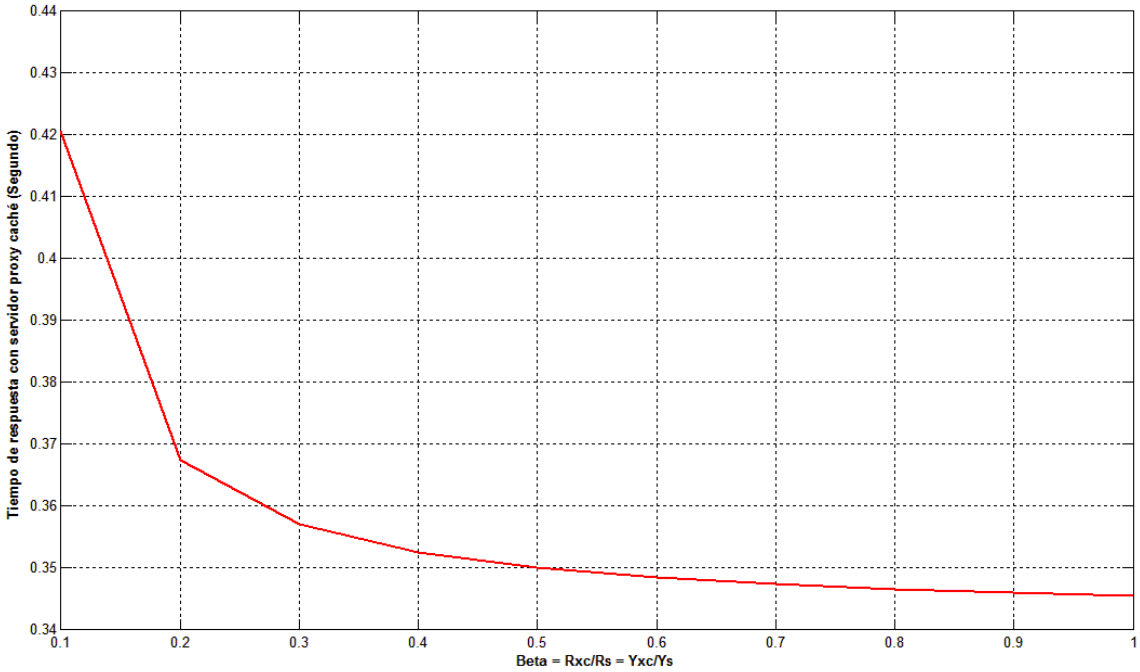


Figura 3.14: Efecto del parámetro β.

En la figura 3.14 se representa el comportamiento de T_{xc} con respecto a β. Al igual que en la figura 3.13 la razón de arribos y el tamaño promedio de los archivos siguen siendo los mismos. Se observa el mismo patrón de comportamiento que en la figura 3.13. Como la velocidad del servidor proxy caché se incrementa, el tiempo de servicio disminuye. Esto conduce a una reducción del tiempo de respuesta global.

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

Como resultado del presente trabajo se arribaron a las siguientes conclusiones:

- El uso de la política LRU como configuración por defecto en los servidores proxy cache no es la alternativa más eficiente.
- Se desarrolló un esquema de teletráfico que caracteriza el acceso a Internet para el servicio Web con el empleo de servidores proxy cache. Se demostró que el tiempo de respuesta depende de varios factores como son la razón de aciertos en cache, la razón de arribos, el tamaño promedio de los documentos y los parámetros de velocidad de los servidores Web remoto y proxy, así como la capacidad de su búfer de salida y la razón dinámica.
- El aumento de la razón de aciertos, el tamaño promedio de los documentos y la velocidad de procesamiento en búfer resultan en una reducción del tiempo de respuesta global.
- Las políticas de reemplazo de caché pueden contribuir al mejoramiento de los tiempo de respuesta y a una mejor utilización del ancho de banda. Se ha demostrado que la mejor estrategia para mejorar la razón de aciertos y por consiguiente el tiempo de respuesta es la política GDS, mientras que la política LFU resultó ser la más eficiente en términos de utilización del ancho de banda.

Recomendaciones

- Se recomienda continuar trabajando en el estudio y desarrollo de nuevas políticas de remplazo con el empleo de métodos adaptativos y técnicas de ordenamiento

estadístico con el objetivo de mejorar la eficiencia del algoritmo de remplazo y aplicarlo a casos concretos como la red UCLV.

- Buscar soluciones mediante la reprogramación o modificación del código de fuente de la herramienta WebTraff para poder extender el tamaño máximo de la caché en las simulaciones del rendimiento de las políticas a los valores típicos utilizados en la actualidad que se encuentran en el rango de 10 GB a 18 GB.
- Realizar una inferencia estadística sobre el tráfico Web presente en la UCLV con el objetivo de simular el modelo de teletráfico propuesto en este trabajo para obtener una estimación optima de los parámetros de configuración del proxy (tamaño de cache y política de remplazo) que permitan mejorar la calidad de servicio de sus usuarios o la equivalente disminución del tiempo de respuesta global.

REFERENCIAS BIBLIOGRÁFICAS

ABDULLA, G., FOX, E., ABRAMS, M. y WILLIAMS, S. (1998). "WWW Proxy Traffic Characterization with Application to Caching". *Virginia Polytechnic Institute & State University* [en línea]. Disponible en: http://csgrad.cs.vt.edu/~abdulla/proxy/proxy-char.ps.

ABRAMS, M., STANDRIDGE, C., ABDULLA, G., WILLIAMS, S. y FOX, E. (1995). "Caching Proxies: Limitations and Potentials". Electron. Proceedings 4th World-Wide Web Conference 95, 1995 Boston, MA. 119-133 [en línea]. Disponible en: http://www.linofee.org/~jel/da/papers/WWW4.ps.gz.

ALI, W., SHAMSUDDIN, S. M. y ISMAIL, A. S. (2011). "Web Proxy Cache Content Classification based on Support Vector Machine". *Journal of Artificial Intelligence,* 4 [en línea]. Disponible en: http://docsdrive.com/pdfs/ansinet/jai/2011/100-109.pdf

ALMEIDA, V., BESTAVROS, A., CROVELLA, M. y OLIVEIRA, A. (1996). "Characterizing Reference Locality in the WWW". Proceedings of the 1996 International Conference on Parallel and Distributed Information Systems, 1996. 92-103 [en línea]. Disponible en: http:// ww2.cs.fsu.edu/~jones/cop5611/ref_locality.pdf.

ALMEIDA, V., CESARIO, M., FONSECA, R., MEIRA, W. y MURTA, C. (1998). "Analysing the behavior of a proxy server in light of regional and cultural issues". Proceedings of the Third International WWW Caching Workshop, 1998 Manchester, England [en línea]. Disponible en: http://www.iwcw.org/1998/21/.

ALMEIDA, V., MURTA, C. y ALMEIDA, J. (1996). "Performance analysis of a WWW server". Proceedings of the 22nd International Conference for the Resource Management and Performance Evaluation of Enterprise Computing Systems, 1996 San Diego, USA [en línea]. Disponible en: http://www.cs.bu.edu/techreports/1996-018-www-performance-analysis.ps.Z.

ARLITT, M. (1996)."*A Performance Study of Internet Web Servers"*. University of Saskatchewan [en línea]. Disponible en: http://www.cs.usask.ca/ftp/pub/discus/thesis_arlitt_co.ps.Z.

ARLITT, M., CHERKASOVA, L., DILLEY, J., FRIEDRICH, R. y JIN, T. (1999). "Evaluating Content Management Techniques for Web Proxy Caches". 2nd Workshop on Internet Server Performance, 1999 Atlanta, Georgia [en línea]. Disponible en: http://www.hpl.hp.com/techreports/98/HPL-98-173.pdf.

ARLITT, M., FRIEDRICH, R. y JIN, T. (1998). "Performance Evaluation of Web Proxy Cache Replacement Policies". *Technical Report HPL-98-97*. Hewlett-Packard Laboratories [en línea]. Disponible en: http://www.hpl.hp.com/techreports/98/HPL-98-97R1.pdf.

ARLITT, M. y JIN, T. (2000). "A workload characterization study of the 1998 world cup Web site". *IEEE Network,* 14**,** 30-37 [en línea]. Disponible en: http://www.hpl.hp.com/techreports/1999/HPL-1999-35R1.

ARLITT, M. y WILLIAMSON, C. (1997)."Internet Web Servers: Workload Characterization and Performance Implications". *IEEE/ACM Trans, on Networking*, 5**,** 631-645 [en línea]. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.109.7690&rep=rep1&type=pdf.

ARLITT, M. y WILLIAMSON, C. (1997)."Trace-Driven Simulation of Document Caching Strategies for Internet Web Servers". *Simulation J*, vol. 68**,** 23-33 [en línea]. Disponible en: http://www.cs.gmu.edu/~setia/inft803/server-caching-SIMULATION97.ps.

BAENTSCH, M., BAUM, L., MOLTER, G., ROTHKUGEL, S. y STURM, P. (1997). "Enhancing the Web's infrastructure: from caching to replication". *IEEE Internet Computing***,** 18-27 [en línea]. Disponible en: http://www.syssoft.uni-trier.de/systemsoftware/Download/Publications/cgr.pdf.

BARFORD, P., BESTAVROS, A., BRADLEY, A. y CROVELLA, M. (1999). "Changes in Web Client Access Patterns - Characteristics and Caching Implications". *World Wide Web,* 2**,** 15—28 [en línea]. Disponible en: http://www.cs.bu.edu/techreports/1998-023-web-client-trace-changes-and-implications.ps.Z.

BARFORD, P. y CROVELLA, M. (1998). "Generating representative Web workloads for network and server performance evaluation". Proceedings of ACM SIGMETRICS Conference, 1998 [en línea]. Disponible en: http://www.cs.bu.edu/faculty/crovella/paper-archive/sigm98-surge.pdf.

BERNERS-LEE, T. 1990. "*World Wide Web, the First Web Client"* [en línea]. Disponible en: http://www.w3.org/People/Berners-Lee/WorldWideWeb.html.

BESTAVROS, A., CARTER, R., CROVELLA, M., CUNHA, C., HEDDAYA, A. y MIRDAD, S. (1995). "Application-Level Document Caching in the Internet". Proceedings of the 2nd International Workshop on Services in Distributed and Networked Environments (SDNE 95), 1995 Whistler, BC [en línea]. Disponible en: http://www.cs.bu.edu/techreports/pdf/1995-002-web-client-caching.pdf.

BOSE, I. y CHENG, H. K. (2000). "Performance models of a firm's proxy cache server". *Decision Support Systems,* 47-57 [en línea]. Disponible en: http://www.sciencedirect.com/science/article/pii/S0167923600000622.

BRAUN, H. y CLAFFY, K. (1995). "Web traffic characterization: an assessment of the impact of caching documents from the NCSA's Web server ". *Computer Network*

and ISDN systems, 28**,** 37-51 [en línea]. Disponible en: http://www.caida.org/publications/papers/1994/wtc/2iwwwc.cache.pdf.

BRESLAU, L., CAO, P., FAN, L., PHILLIPS, G. y SHENKER, S. (1999). "Web caching and Zipf-like distributions: Evidence and implications". Proceedings of IEEE INFOCOM 1999, 1999 New York, NY [en línea]. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.34.8742&rep=rep1&type=pdf.

BUSARI, M. (2000). "*Simulation evaluation of Web caching hierarchies"*. University of Saskatchewan [en línea]. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.37.5413&rep=rep1&type=ps.

BUSARI, M. y WILLIAMSON, C. (2001). "On the sensitivity of Web proxy cache performance to workload characteristics". Proceedings of IEEE INFOCOM 2001, 2001 Anchorage, Alaska USA. 1225-1234 [en línea]. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.8.9652&rep=rep1&type=pdf.

BUSARI, M. y WILLIAMSON, C. (2002). "ProWGen: A Synthetic Workload Generation Tool for Simulation Evaluation of Web Proxy Caches". *Computer Networks,* 38**,** 779-794 [en línea]. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.85.9530&rep=rep1&type=pdf.

CARLOS, R., BESTAVROS , A. y CROVELLA, M. (1995). "Characteristics of WWW Client-based Traces". Computer Science Department, Boston University [en línea]. Disponible en: http://www.cs.bu.edu/techreports/pdf/1995-010-www-client-traces.pdf.

CHERKASOVA, L. (1998). "Improving WWW proxies performance with greedy-dual-size-frequency caching policy". *Technical Report HPL-98-69R1*. Hewlett-Packard Laboratories [en línea]. Disponible en: http://www.hpl.hp.com/techreports/98/HPL-98-69R1.pdf.

CROVELLA, M. y BESTAVROS, A. (1997). "Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes". *IEEE/ACM Transactions on Networking,* 5**,** 835-846 [en línea]. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.84.3248&rep=rep1&type=pdf.

CROVELLA, M. y TAQQU, M. (1999). "Estimating the heavy tail index from scaling properties". *Methodology and computing in applied probability,* 1**,** 55-79 [en línea]. Disponible en: http://www.cs.bu.edu/~crovella/paper-archive/aest.ps.

DUSKA, B., MARWOOD, D. y FEELEY, M. (1997). "The measured access characteristics of world wide Web client proxy caches". Proceedings of the USENIX Symposium on Internet Technologies and Systems, 1997 [en línea]. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.40.9044&rep=rep1&type=pdf.

FIELDING, R., GETTYS, J., MOGUL, J. C., FRYSTYK, H., MASINTER, L., LEACH, P. y BERNERS-LEE, T. (1998). "Hypertext Transfer Protocol - HTTP/1.1". DRAFT, I., ed. HTTP Working Group, 1998 [en línea]. Disponible en: http://www.w3.org/Protocols/rfc2616/rfc2616.html.

FLICKENGER, R. (2006). "How to Accelerate Your Internet". R., F. (ed.) *A Practical Guide to Bandwidth Management and Optimization Using Open Source Software. INASP/ICTP, Seattle.* INASP/ICTP [en línea]. Disponible en: http://www.linuxrulz.org/nkukard/bwmo-ebook.pdf.

GROSS, D. y HARRIS, C. (1985). "*Fundamentals of Queuing Theory"*, John Wily and Sons.

JACKSON, J. R. (2004). "Jobshop-Like Queueing Systems ". *Management science,* 50**,** 1796-1802 [en línea]. Disponible en: http://www.jstor.org/pss/2627213.

LELAND, W., TAQQU, M., WILLINGER, W. y WILSON, D. (1994). "On the Self-Similar Nature of Ethernet Traffic Extended Version". *IEEE/ACM Transactions on Networking,* 2**,** 1-15 [en línea]. Disponible en: http://eeweb.poly.edu/el933/papers/Willinger.pdf.

LI, W. (2000). "Random texts exhibit Zipf's-law-like word frequency distribution". *Information Theory, IEEE Transactions on,* 38**,** 1842-1845 [en línea]. Disponible en: http://www-personal.umich.edu/~warrencp/WLi.pdf.

LUOTONEN, A. (1998). "*Web Proxy Servers"*, Prentice-Hall.

MAHANTI, A. (1999). "*Web Proxy Workload Characterization and Modeling"*. University of Saskatchewan [en línea]. Disponible en: http://www.cse.iitd.ernet.in/~mahanti/papers/mscthesis.pdf.

MAHANTI, A. y WILLIAMSON, C. (1999). "Web Proxy Workload Characterization". *Technical Report.* Department of Computer Science, University of Saskatchewan [en línea]. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.68.3089&rep=rep1&type=ps.

MAHANTI, A., WILLIAMSON, C. y EAGER, D. (2000). "Traffic analysis of a Web proxy caching hierarchy". *IEEE Network,* 14**,** 16-23 [en línea]. Disponible en: http://pages.cpsc.ucalgary.ca/~mahanti/papers/ieee00.pdf.

MALTZAHN, C., RICHARDSON, K. y GRUNWALD, D. (1997). "Performance Issues of Enterprise Level Web Proxies". Proceedings of SIGMETRICS '97, 1997 Seattle, Washington [en línea]. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.91.5049&rep=rep1&type=pdf.

MENASCE, D. y ALMEIDA, V. (1998). "*Capacity Planning for Web Performance: Metric, Models, and Methods"*, Prentice-Hall.

MOGUL, J. C. (1995). "Network Behavior of a Busy Web Server and its Clients". *WRL research.* Digital Western Research Laboratory [en línea]. Disponible en: http://www.hpl.hp.com/techreports/Compaq-DEC/WRL-95-5.pdf.

NISHIKAWA, N., HOSOKAWA, T., MORI, Y., YOSHIKA, K. y TSUJI, H. (1998) "Memory-based Architecture for Distributed WWW Caching Proxy". Proceedings World Wide Web Conference, 1998. 205-214 [en línea]. Disponible en: http://www7.scu.edu.au/1928/com1928.htm.

PATIL, J. B. y PAWAR, B. V. (2011). "Improving Performance on WWW using Intelligent Predictive Caching for Web Proxy Servers". *International Journal of Computer Science Issues,* 8**,** 402-408 [en línea]. Disponible en: http:// www.ijcsi.org/papers/IJCSI-8-1-402-408.pdf.

PODLIPNIG, S. y BÖSZÖRMENYI, L. (1998). "A Survey of Web Cache Replacement Strategies". University Klagesnfurt [en línea]. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.167.5825&rep=rep1&type=pdf.

REDDY, S., SWETHA, L. y PRASAD, C. G. (2011). "Analysis and Design of Enhanced HTTP Proxy Cashing Server ". *International Journal of Computer Technology and Applications* 02**,** 537-541[en línea]. Disponible en: http:// ijcta.com/documents/volumes/vol2issue3/ijcta2011020326.pdf.

RIZZO, L. y VICISANO, L. (1998). "Replacement Policies for a Proxy Cache". *Technical Report RN.* Department of Computer Science University College London [en línea]. Disponible en: http://info.iet.unipi.it/~luigi/caching.ps.gz

ROADKNIGHT, C., MARSHALL, I. y VEARER, D. (2000)."File popularity characterisation". *ACM Sigmetrics Performance Evaluation Review,* 47 [en línea]. Disponible en: http://eprints.lancs.ac.uk/27228/1/27228.pdf.

RODRIGUEZ, P. y BIERSACK, E. 1998. "Continuous multicast push of Web documents over the Internet". *IEEE Network,* March/April**,** 18-31 [en línea]. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.56.8480&rep=rep1&type=pdf.

ROUSSKOV, A. y SOLOVIEV, V. (1998). "On Performance of Caching Proxies". Proceedings of the ACM SIGMETRICS "98 Conference, 1998 [en línea]. Disponible en: http://www.cs.ndsu.nodak.edu/~rousskov/research/cache/squid/profiling/papers/on.performance.old.ps.gz.

SHANMUGAVADIVU, S. y RAJARAM, M. (2010). "Web caching and response time optimization based on eviction method ". *International Journal of Engineering Science and Technology,* 2**,** 4912-4921 [en línea]. Disponible en: http://*www.ijcaonline.org/volume10/number9/pxc3871855.pdf.*

SLOTHOUBER, L. (1996). "A model of Web server performance". *5th International World Wide Web Conference.* Paris, France [en línea]. Disponible en: http://citeseer.ist.psu.edu/viewdoc/download;jsessionid=D94BACA9BC801E77516E0024B5DE5632?doi=10.1.1.88.7107&rep=rep1&type=pdf.

TANENBAUM, A. S. (1996). "Computer Networks Third Edition". Prentice-Hall ISBN.

WESSELS, D. y CLAFFY, K. (1998). "ICP and the Squid Web Cache". *IEEE Journal on Selected Areas in Communication,* 16**,** 345-257 [en línea]. Disponible en:

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.32.7137&rep=rep1&type=ps.

WILLIAMS, S., ABRAMS, M., STANDRIDGE, C., ABDULLA, G. y FOX, E. (1996) "Removal Policies in Network Caches for World-Wide Web Documents". Proceedings ACM SIGCOMM 96, 1996 Stanford, CA [en línea]. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.40.9549&rep=rep1&type=pdf.

YU, P. y MACNAIR, E. (1998)."Performance Study of a Collaborative Method for Hierarchical Caching in Proxy Servers". Proceedings World-Wide Web Conference, 1998. 215-224 [en línea]. Disponible en: http://www7.scu.edu.au/1829/com1829.htm.

ANEXOS

Anexo I Comandos para evaluación del tiempo de respuesta con Matlab.

```
Txc = 1./((1./Ixc)-lambda)+ p.*((1./((1./((F/Bxc)*(Yxc+(Bxc/Rxc))))-lambda1))+(F/Nc)) + (1-p).*((1./((1./Is)-lambda2)) + (1./((1./((F/Bs)*(Ys+(Bs/Rs))))-lambda2./q))+(F/Ns) + (1./((1./((F/Bxc)*(Yxc+(Bxc/Rxc))))-lambda2))+(F/Nc))

p = 0:0.1:1;

lambda = 20;

lambda1 = p*lambda;

lambda2 = (1-p)*lambda;

F = 5000;

Bs = 2000;

Is = 0.004;

Ys = 0.000016;

Rs = 1.25 * 10^6;

alpha = 1;

beta = 1;

gama = 1;

Bxc = alpha.*Bs;

Rxc = beta*Rs;

Yxc = beta*Ys;
```

```
Ixc = gama*Is;

Ns = (1.544/8) * 10^6;

Nc = (128/8) * 10^3;

q = min(1,(Bs/F));
```