International Journal of Electronics, Communication & Soft Computing Science and Engineering (IJECSCSE)
Volume 1, Issue 1

Effective Web Cache Algorithm

Vinit A. Kakde, Sanjay K. Mishra

***Abstract*— Web caching is a very important feature. In order to alleviate the demands on the network bandwidth and improve the server's quality. A key component of a cache is its replacement policy, which is a decision rule for evicting a page currently in the cache to make room for a new page. Our approach combines classical replacement policies (LFU, MRU, LRU, SLRU etc.) to optimize the overall performance (based on criteria such as network traffic, hit ratio application execution time.etc) The LRU is generalized algorithm such as to take account access cost and expiration time. The performance of LRU is poor in terms of hit ratio as compared to LFU and MRU. To optimize the performance of LRU we proposed the Segmented Least Recently Used replacement policy which has two partitions and maintains both frequency and regency.**

I. INTRODUCTION

1.1 Internet Impact

The internet and web have changed the way we conduct business and get educated. Whether utilizing e-commerce applications, outsourcing internal processes, or simply improving communications with suppliers and partners, business is increasingly conducted over the internet. Internal enterprise intranets and extranets are more and more common. In schools, classroom activities increasingly are enriched through internet-based instruction while research gets done over the web from libraries, schools, and home. This connectivity explosion offers enterprises the opportunity to be more productive than ever before. Communications with suppliers can be handled entirely online, reducing paper work and saving time while providing more sophisticated tracking of transactions. Business functions such as payroll and human resources can be outsourced to third parties and managed using the web. And employees can communicate with management, customers, and each other more effectively, increasing productivity and efficiency. As these trends continue to gain momentum, web traffic will increase as organizations become more reliant on internet and web connectivity. It is therefore essential that networks keep pace, providing a secure, reliable, efficient, and cost-effective infrastructure for conducting business or learning online.

1.2 Web Caching

In simple terms, web caching is a technology that can significantly enhance end-user's web browsing experience and, at the same time, save bandwidth for service providers. In details, a web cache is a temporary storage place for data content requested from the internet. After an original request for data has been successfully fulfilled, and that data has been stored in the cache, further requests for those files (e.g., HTML pages, images) results in the information being returned from the cache, if certain conditions are met, rather than the original location. Web Caching is the widely used technique, used by Internet Service Providers (ISPs) all around the world, to save bandwidth and to improve user response time. In short, web caching temporarily stores web objects – HTTP and FTP data – flowing into ISP's network.

1.2.1 Need for Web Caching

The internet depends on the Client Server Model in which clients request services from the server through for example their web browsers, and the server could take the requests, process them and return the results back to the clients. As shown in figure 1- (a) above the server is overloaded, and in figure 1-(b) however there is less overload on the server since some responses can be returned straight from the cache.

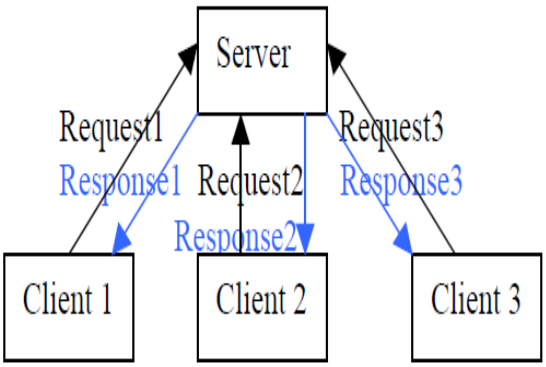

Figure 1.1- (a) Client-Server Model.

If too many clients make requests at the same time from the server, then the server will be overloaded with requests and therefore there might be delays before some of these requests can be processed. This is where caching becomes useful since its architecture makes use of caching proxies which reside between the client and the server.

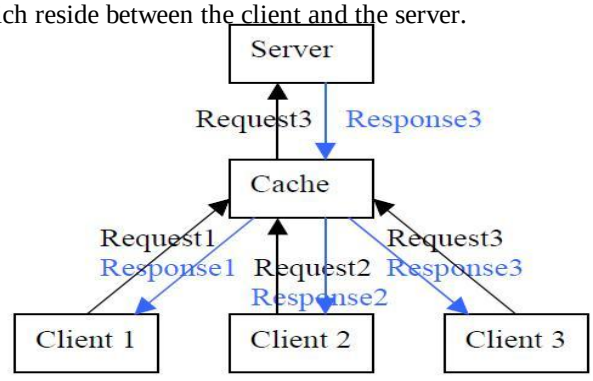

Figure 1.1 (b) Client-Server Model.

In this architecture when a client makes a request to the server, and gets a response back, the popular and useful data will be stored somewhere close to the client for a period of time so that when making the same request in the future, the first place to look at will be in the client's cache system, rather than having to connect to the network to pull the information again. If the data is already stored in the client's system, then there is no need to waste network bandwidth and overload the server with unnecessary requests. The response will just be returned from the cache. It is therefore safe to say that from a client's point of view, the web cache has the same functionality and effect of a web server and a web cache would respond to the client in the same manner as a web server would, [1].

Web caching enhances web browsing in much the same way. When a user visits a site, say www.yahoo.com, web caching (if in use and available) will retrieve the page from yahoo's web server and store a copy of that page locally in cache server. The next time a user requests www.yahoo.com the web cache delivers the locally cached copy of the page (without fetching it from yahoo's web server). The user will experience a very fast download because the request did not have to traverse the entire internet – all the way to where yahoo server is located. If the contents are dynamic then image, logos or such type of objects store on cache so whenever we visits second time browser will only load some dynamic contents or updated contents and rest of contents will get loaded from the local cache instead of web server so, the bandwidth that would normally be used to download the web site is not required and is free for other information retrieval or delivery.

The flowchart shows the working flow of caching system which loads the contents and saves it on local cache.

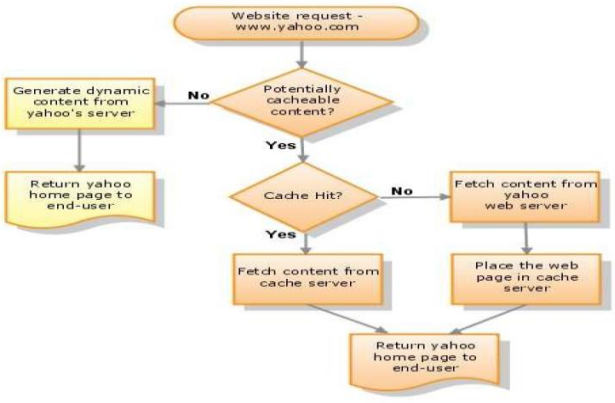

Figure 1.2 Flowchart depicts how web cache works from end-user perspective

1.2.2 Working of Cache System

A web cache is a mechanism for the temporary storage (caching) of web documents, such as HTML pages and images to reduce bandwidth usages, server load and perceived lag. A web cache stores copies of documents passing through it; subsequent requests may be satisfied from the cache if certain conditions are met.

Web sites are continually updating their contents. News headlines change, stock quotes change, weather changes. It may seem that caching is not worthwhile if it is returning outdated material. A traffic report that is two hours old doesn't do you much good. Fortunately there are checks and balances in place to ensure that the content you are viewing is current. Web sites are made up of many small pieces that come together to make a complete page. A site might have logos, photographs, tables, text, and sounds. Each item will be cached as a different object, and some items may not cache at all. For example, when you access CNN.com frequently your cache may hang on to the CNN logo objects some advertising bars, and the rest of the stuff that makes up the basic look of the CNN Web site. But the news items will not sit in cache because they change so often. In this case your cache has made the CNN site much easier and faster to download because all the static graphics are already on hand and the only thing you need to complete the picture is the news content.

II. CACHE REPLACEMENT POLICIES: FORMAL DEFINITION

Cache replacement policy plays an extremely important role in web caching. Hence, the design of efficient cache replacement algorithms is required to achieve highly sophisticated caching mechanism. In general, cache replacement algorithms are also called web caching algorithms. As cache size is limited, a cache replacement policy is needed to handle the cache content. If the cache is full when an object needs to be stored, the replacement policy will determine which object is to be evicted to allow space for the new object. The optimal replacement policy aims to make the best use of available cache space, to improve cache hit rates, and to reduce loads on the origin server. The different page replacement policies are as fallows review.

2.1Least Recently Used (LRU) Page Replacement Policy

The simplest and most common cache management approach is Least-Recently Used (LRU) algorithm, which removes the least recently accessed objects until there is sufficient space for the new objects.

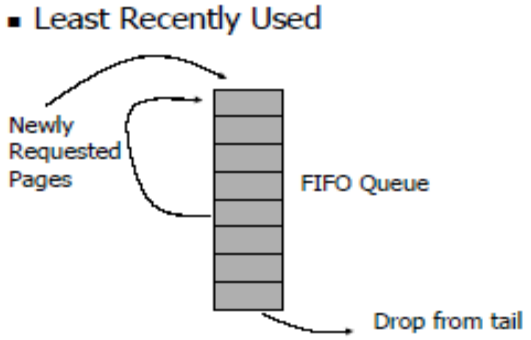

Figure 2.1: Example of LRU page replacement policy

LRU is easy to implement and proficient for uniform size objects, like in the memory cache. However, it does not perform well in web caching since it does not consider the size or the download latency of objects. This algorithm exploits the temporal locality of the user's accesses, and it is very simple to implement because the eviction mechanism requires only the access time-stamp.

2.2 Least-Frequently-Used (LFU) Page Replacement Policy

It is another common web caching that replaces the object with the least number of accesses.

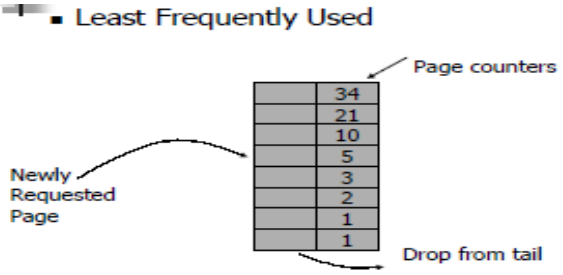

Figure 2.2: Example of LFU page replacement policy

LFU keeps more popular web objects and evicts rarely used ones. However, One potential drawback of LFU is that some objects may accumulate large reference counts and never become candidates for replacement, even if these objects are no longer in the active set (i.e., the cache could become polluted with inactive objects).

2.3 Most Recently Used Page Replacement Policy

The MRU is also called as fetched and discard policy. MRU algorithm removes the most recently used resource first. As shown in figure 1.14 the pages are drop from head in the queue which maintains the list of older web pages. MRU works contrast to LRU policy. The algorithm is the best choice when access of resources is highly unpredictable. MRU policy is most often used where historical information is to be accessed.

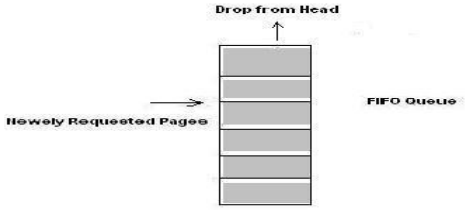

Figure 2.3: Example of MRU page replacement policy

2.4 Segmented LRU (SLRU) Page Replacement Policy

To alleviate the web object pollution problem The Segmented LRU (SLRU) policy designed and it is use in a disk cache we include it in this study because it considers both frequency and regency of reference when making a replacement decision. The SLRU policy partitions the cache into two segments: an unprotected segment and a protected segment (reserved for popular objects). On the initial request for an object, the object is added to the unprotected segment. When a cache hit occurs, the object is moved to the protected segment. Both segments are managed with the LRU policy. However, only objects in the unprotected segment are eligible for replacement. This allows the once popular objects to remain in the cache for a longer period of time in case they regain their popularity. If space is needed to add these objects, the least recently used objects in the unprotected segment are removed. This policy requires one parameter, which determines what percentage of the cache space to allocate to the protected segment. The SLRU policy performs best when a balance is found that allows for popular objects to be retained for long periods of time without becoming susceptible to pollution, [4].

III. IMPLEMENTATION STRATEGY

3.1 Motivation and Implementation Scheme

The primary reason is that based on the behavior of the average web clients, the set of regularly accessed web pages will stay the same for a while. Thus the client system is a good candidate of implementing a web caching system. This caching system will honor those frequently accessed web pages retrieve with the faster response time as well as save the network bandwidth because the cached web pages are already resided on the client's hard disk. Based on this study, the work also tries to determine if there is any performance limitation caused by the physical system barrier, and will the physical barrier overwhelm the benefit obtained from the caching performance.

3.2 Implementation Flow

As shown in figure 3.1 the flow of the data is described that is any pages from web can be accessed by client after it send some request through web browser and wait for the response. The server processes the client request and analyze that it was in cache folder or it was get from the web server, if it retrieves from the local cache folder itself it simply update the timestamp and size of the web page and forward to the client, else if get from the web server, it saves a copy in local folder then forward to client, [7].

Web browser has folder which stores the webpage being retrieved from web server. There is eviction method which evicts the page from the cache to make the room for next coming pages from web server. We proposed and implemented the page replacement policies based on regency and frequency of the web pages.

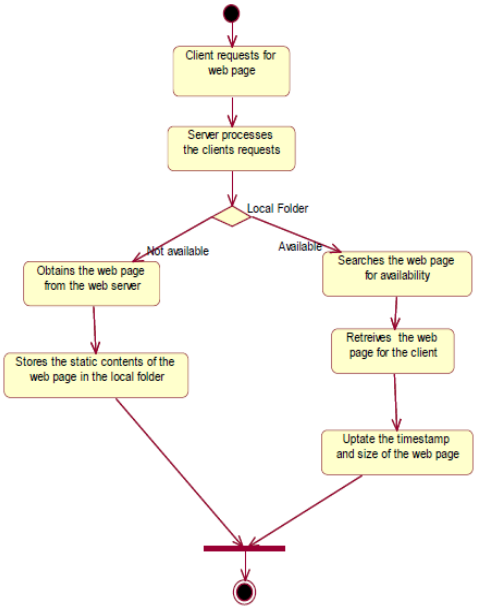

Figure 3.1 Flowchart of Basic Caching System

With the simulation study and web traces we analyzed the performance metric in terms of hit ratio on page replacement algorithms like least frequently used pages (LFU) works on hit counts of web pages , least recently used pages (LRU) based of timestamp, Most recently used pages (MRU) contrast to LRU. And Segmented LRU (SLRU) which maintains the frequency and regency to improve the performance of LRU.

3.3 Module Implementation

3.3.1 Implementation of Generalized Web Browser.

Client-side caching is a feature that stores frequently used information on the client's machine. It provides performance enhancements on the client side by allowing the client to quickly access a file that would have normally be accessed from a server. Client caching is especially effective when the client disconnects from the server, in that case files can still be accessed from the local cache. Client-side caching is usually defined in the client's browser settings.

A web browser or internet browser is a software application for retrieving, presenting, and traversing information resources on the World Wide Web. An *information resource* is identified by a Uniform Resource Identifier (URI) and may be a web page, image, video, or other piece of content. Although browsers are primarily intended to access the World Wide Web, they can also be used to access information provided by web servers in private networks or files in file systems. Browser can also be used to save information resources to file systems.

As stated the number one function of a web browser is to give user's access to the World Wide Web and saves the pages and load from local cache when user demands the same page second time. However, as the internet has developed the web browsers have also enhanced their capabilities to take advantage of the changing environment - In this way the web browser purpose has extended to cover other aspects of the internet.

IV. Desirable Properties Of Web Caching System

Besides the obvious goals of web caching system. We would like a web caching system to have a number of properties. They are fast access. Robustness, transparency, scalability, efficiency, adaptively, stability, loads balanced, ability to deal with heterogeneity, and simplicity.

Fast Access: From user's point of view, access latency is an important measurement of quality of web service. A desirable caching system should aim at reducing web access latency. In particular, it should provide user a lower latency on average than those without employing a caching system.

Robustness: from user's prospect, the robustness means availability, which is another important measurement of quality of Web service. Users desire to have Web service available whenever they want. The robustness has three aspects. First, it's desirable that a few proxies crash wouldn't tear the entire system down. The caching system should eliminate the single point failure as much as possible. Second, the caching system should fall back gracefully in case of failures. Third, the caching system would be design in such a way that it's easy to recover from a failure.

Transparency: A Web caching system should be transparent for the user the only results user should notice are faster response and higher availability.

Scalability: We have seen an explosive growth in network size and density in last decades and is facing a more rapid increasing growth in near future. The key to success in such an environment is the scalability. We would like a caching scheme to scale well along the increasing size and density of network.

Efficiency: There are two aspects to efficiency. First, how much overhead does the Web caching system impose on network? We would like a caching system to impose a minimal additional burden on the network.

Load balancing: It's desirable that the caching scheme distributes the load evenly through the entire network. A single proxy/server shouldn't be a bottleneck (or hot spot) and thereby degrades the performance of a portion of the network or even slow down the entire service system.

V. Conclusions And Future Scope

The enormous raise of the traffic in the internet due to the exponential augmentation of user's interactions with web servers is generating a lot of blockages. As a corollary, users frequently experience high delay when access these web pages and even more aborting or resetting connection with real-time web applications, such as online flight booking,

online banking amongst others. This topic finds the elucidation to this predicament with the prologue of and web cache. The study is made on different types of page replacement policies in cache system and the comparative study is made between them. After performing the experimentation on real-time application experimental results have revealed that the proposed approach can improve the performance of hit ratio (HR).

Personalization provides the web pages as per the needs of the web users. Prefetching refers to fetching information from web servers even before they are requested. The prefetching process will be highly essential for the personalization of web details. The majority of web objects including streaming media (video and audio cache) remain cacheable as the web evolves.

REFERENCES

[1] Web Caching: Architectures, Models and Importance to the Internet By Sarmed AL-Najim.
[2] Survey of Web Caching Schemes for the Internet. Jia Wang.
[3] Survey of Web Caching and Prefetching. Waleed Ali, Siti Mariyam Shamsuddin, and Abdul Samad Ismail Int. J. Advance. Soft Computing. Appl.,Vol. 3, No. 1, March 2011 ISSN 2074-8523; Copyright © ICSRS Publication, 2011.
[4] The performance of a client-side web caching system by Ying-Lin Chen.
[5] Adaptive web caching algorithms. Geetika Tewari and Kim Hazelwood.
[6] J. Wang, "Fundamentals of erbium-doped fiber amplifiers arrays (Periodical style—Submitted for publication)," *IEEE J. Quantum Electron.*, submitted for publication.
[7] C A Fuzzy Algorithm for Web Caching. Maria Carla Calzarossa, Giacomo Valli.
[8] Evaluation of Efficient Web Caching and Prefetching Technique for Improving the Proxy Server Performance G.N.K.Suresh Babu and S.K.Srivatsav.
[9] Caching Behaviors of web browser by Dawn Pazych AccelerationSystem Architecture (ACA) Nov-07.
[10] An Adaptive Coherence-Replacement Protocol for Web Proxy Cache Systems. Jose Aguilar, Ernst L. Leiss**.**
[11] Web Caching: Optimizing for internet and Web Traffic (White paper).
[12] J. Distributed Caching System for n-tier Web Application Using Java: A Comparative Study between JCS, Ehcache, OSCache and Cache4J Riktesh Srivastava, Invertis Journal of Science & Technology Vol. 2, No. 3, 2009; pp. 143-152.
[13] Measurement and Analysis Web Page Response Time Understanding and measuring performance test results by Alberto Savoia.

AUTHOR'S PROFILE

Vinit A. Kakde

(vinit.kakde@gmail.com) received the B.E. degree in Information Technology from SGBAU University, Amravati, Maharashtra, in 2007. From 2008 to2010.he worked for the Government College of Engineering, Amravati as a Lecturer in IT Department. He is currently working toward his M.Tech. degree at the University of RGPV, Bhopal. His research interests are in theory of computation, traffic analysis, network systems modeling, and performance evaluation.

Prof. Sanjay K. Mishra

is-currently working in TIT, Bhopal as assistant lecturer in Information Technology Department. He is currently working toward his PhD degree. His research interests are in network systems modeling and performance evaluation.