ISSN 2278 – 1021

*International Journal of Advanced Research in Computer and Communication Engineering*
*VOL. 1, ISSUE 1, MARCH 2012*

# Design and Implementation of Server Side Web Proxy Caching Algorithm

**Dr.K.Ramu[1] and Dr.R.Sugumar[2]**

Asssitant Professor, Dept., of CSE, Arignar Anna Institute of Science & Technology, Chennai, India.
Asssitant Professor, Dept., of IT, R.M.D. Engineering College, Chennai, India.

**ABSTRACT –** Due to the increase in popularity of the World Wide Web (WWW) has introduced new issues such as Internet traffic and bandwidth consumption. Recently, much research has focused on improving Web performance by reducing the bandwidth consumption and WWW traffic. Web proxy caching is a well-known technique for reducing access latencies and bandwidth consumption. As in other caching systems, a replacement policy is necessary for determining when and what to evict from the cache, and many proxy caching algorithms have been earlier. In this paper, we propose a server side web caching algorithm in oreder to overcome various issues of web proxy caching such as load balancing, transparency, scalability and cache miss. Web proxy caching is a well-known strategy for improving performance of Web-based systems by keeping Web objects that are likely to be used in the near future closer to the client.

**Keywords–** Web proxy, Bandwidth, Cache, Server Side Web Caching Algorithm, LFU, LRU

## I. INTRODUCTION

Web proxy caches have an important role in reducing server loads, client request latencies, and network traffic. Web caching is a well-known technique for reducing access latencies and bandwidth consumption. As in caching systems, a replacement policy is necessary for determining when and what to evict from the cache and many proxy caching algorithms have been proposed and evaluated. This topic analyzes the distribution of current web contents and re-evaluates various proxy cache replacement algorithms including LFU, LRU and MRU.

A web proxy cache sits between one or more web servers and a client or many clients, and processes the requests and saves the copies of the responses made to the requests. Then, if there is another request for the same URL, it can use the response that it has, instead of asking the web server for it again. Web caches are used to reduce latency and to reduce network traffic**.** Web caching is different from traditional caching in different ways. An important difference is that the size of web objects is not uniform, and the transfer time costs are not identical. Due to the increased web usage and its development a huge amount of network traffic prevails in present web services. It is very much essential to reduce this traffic for utilizing the internet facilities in an efficient manner. Web caching improves the overall performance of web and its related services.

The objective of this paper is to design and impelmentation of a server side proxy caching algorithm which can be used to reduce the network traffic and bandwidth.

A web cache is a mechanism for the temporary storage (caching) of web documents, such as HTML pages and images, to reduce bandwidth usage, 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.

## II. RELATED WORK

Web proxy tracing and caching are highly active research areas. Recent studies of Web traffic include analyses of Web access traces from the perspective of browsers proxies. Earlier tracing studies were limited in request rate, number of requests, and diversity of population. The most recent tracing studies have been larger and more diverse. In addition to static analysis, some studies have also used trace-driven cache simulation to characterize the locality and sharing properties of very large traces and to study the effects of cookies, aborted connections, and persistent connections on the performance of proxy caching [6].

Mogul (2003) proposed a dependency graph (DG) algorithm is constructed for prediction of next page. The DG consists of nodes that represent Web pages, and arcs that indicate that the target node is accessed after the original node within a time window. Every arc is assigned to weight that represents the probability that the target node will be the next one. The DG based prefetching approach predicts and prefetches the nodes whose arcs connect to the current accessed node and have weights higher than a threshold. Recently, has presented double dependency graph (DDG) prediction algorithm that considers the characteristics of the current web sites to improve the performance achieved by web prefetching. Palpanas (2004) proposed Prediction-by-

Partial-Matching (PPM) algorithm. PPM is one of the most successful Web prefetching techniques, which depends on high-order Markov mode. In PPM, prefetching decisions are made based on historical user requests in a Markov prediction tree. However, as more pages are requested and added into the history structure, its size will grow linearly. Several researchers proposed different methods to limit the size of the history structure.

Yang (2006) proposed the association Rules Based Prefetching Approach. In this, Association rule discovers groups of homogeneous pages that are commonly accessed together in same user session. The user session is defined as the sequence of pages made by a single user during a visit to a particular site. In association rules, support and confidence are two important measures for benchmarking strength of any association rule. Support is defined as the discovery of frequent pages, while confidence is defined as the discovery of association rules from these frequent pages. Wang (2008) noted a predictive prefetching, can be implemented by having an engine which, after processing the past references, derives the probability of future access for the documents accessed so far.

The prediction engine can reside either in the client or in the server side. In the former case, it uses the set of past references to find correlations and initiates prefetching. No modifications need to be made neither to the current Web infrastructure (e.g., HTTP protocol, Web servers) nor to Web browsers if the prefetcher module runs as a proxy in the browser.

Liu (2009) proposed a proxy cache prefetching is a major web caching technique that attempts to serve user Web requests from one or a network of proxies located between the end user and Web servers hosting the original copies of the requested objects. Proxy caching effectively reduces the network resources that Web services consume, while minimizing user access latencies. Deploying Web caching proxies over the Internet is not easy, however, and presents several cache management challenges.

## III. ISSUES OF WEB PROXY CACHING

Due to the explosive and ever growing size of the web, distributed caching has received considerable attention. The major aim of cache is to move the frequently accessed information closer to the users. Caching system should improve performance for end users, network operators, and content providers. Caching can be recognized as an effective way to: speed up web access, reducing latency perceived by the users, reduce network traffic, reduce server load, and improve response time to the users.

*A. Load Balancing*

The situation occur at any time for large number of clients who wishes to simultaneously access data or get some services from a local cache with single server. If the site is not provisioned to deal with all of these clients simultaneously, service may be degraded or lost. Several approaches to overcoming this issue have been proposed. The most frequently used method is replication. This replication strategy stores copies of hot pages or services throughout the Internet; this spreads the work of serving a page or service across several servers.

*B.Transparency*

Transparency of cache systems enables users to get the benefits of caches without knowing that they exist, or without knowing their physical location. The advantages of this technique are easy to use, no configuration required by the end user and no users can bypass the cache.

*C.Scalability*

It is vital that the cache system be scalable as the number of users and servers increases. It can be clustered or cooperative and stand-alone caches. Stand-alone caches are better suited for individual systems and are easier to maintain. However, cooperation between caches could provide more information about cached data, which could be communicated between caches without referring to the originating servers.

*D.Cache miss*

Cache systems should be capable of efficiently handling cache misses. When a request cache misses, a decision should be taken on where to forward the request. And also a cache system should decide on which data to be cached or should all cached data be treated equally.

## IV. DESIGN AND IMPLEMENTATION OF SERVER SIDE WEB PROXY CACHING

Web proxy caching is a well-known strategy for improving performance of Web-based systems by keeping Web objects that are likely to be used in the near future closer to the client. Most of the current Web browsers still employ traditional caching policies that are not efficient in Web caching. Web caching is an emerging technology in Web and in Web caching if the client is requesting a page from a server, it will fetch from the server and will give response to the server (Barish et al. 2000). According to the locations where objects are cached, Web caching technology can be classified into three categories i.e., client’s browser caching, client-side proxy caching, and server-side proxy caching.

A Server-Side Web Proxy Caching (SSPC) is a server that acts as an intermediary for requests from clients seeking resources from other servers. A client connects to the proxy server, requesting some service, such as a file, connection, web page, or other resource available from a different server. The proxy server evaluates the request according to its filtering rules. For example, it may filter traffic by IP address or protocol. If the request is validated by the filter, the proxy provides the resource by connecting to the relevant server and requesting the service on behalf of the client. A proxy server may optionally alter the client's request or the server's response, and sometimes it may serve the request without contacting the specified server. In this case, it 'caches' responses from the remote server, and returns subsequent requests for the same content directly.

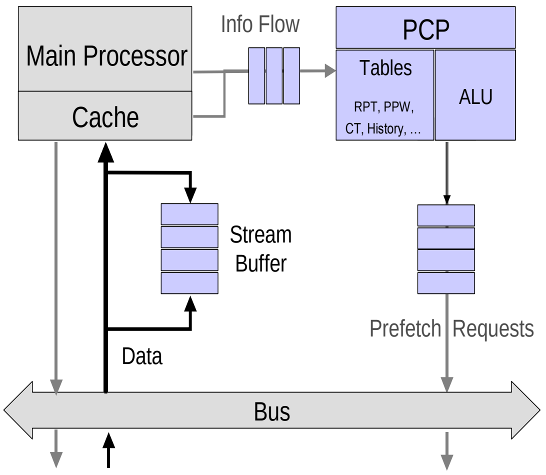

Fig.1: Architecture of Proxy Caching

The Figure 1 show a Server Side Proxy Caching and which runs with many features such as Reduces network traffic, Reduces Latency time, reduce load on web server and saves processing power. This architecture also inherently helps speeder browsing of web pages with use of least grade page replacement algorithms. This server is successfully implemented with a few numbers of clients but it could be implemented for more of them. As mentioned before it is more reliable, more advantageous than the existing one which uses the old Data structures concept. It can work in a larger network and also maintains load balancing. This Server Side Proxy Caching system is executable under any platform and with any number of clients.

### *A. Server Side Web cache algorithm*

There has been extensive theoretical and empirical work done on exploring web caching policies that perform best under different performance metrics. Many algorithms have been proposed and found effective for web proxy caching.

These algorithms range from simple traditional schemes such as Least-Recently Used (LRU), Least- Frequently Used (LFU), First-In First-Out (FIFO), and various size-based algorithms, to complex hybrid algorithms such as LRU-Threshold, which resembles LRU with a size limit on single cache elements, Lowest-Relative Value (LRV), which uses cost, size and last reference time to calculate its *utility*, and GreedyDual, which combines locality, size and cost considerations into a single online algorithm.

The table I show the proposed Server Side Web Proxy Caching algorithm. This algorithm focused on the second aspect, algorithm of replacing documents. With the study of web coaching's characteristics going further; algorithms of replacing documents based on the statistics of collected web data are proposed. Each of this considers one or more of the following factors into its scheme:

1. Document reference frequency
2. Document reference recency
3. Document size
4. Strong or loose consistence of documents
5. Replica document in different proxies
6. Non-replica document in different proxies.

Efficient schemes combine more than one of factors in their implementation of web cache. Some algorithms consider different cache architecture to improve caching performance, such as a fully-connected network of N cache. The Web cache algorithm's hit ratio for different values of the fetch unit is shown in Table II. It is clear that the hit ratio of the proposed algorithm has improved gradually for different number of units and fetches.

Table I: Server side web cache algorithm

| | |
|---|---|
| 1 | WHILE there is a page p € Cache in the current window w |
| 2 | Serve first such p and mark the page |
| 3 | Shift w if possible |
| 4 | IF all pages in the cache are marked |
| 5 | Unmark all the pages |
| 6 | Evict randomly an unmarked page in the cache |
| 7 | Fetch and mark the first page in the current window and Serve |
| 8 | Shift the current window w |

Table II : Hit ratios for different values of the fetch unit

ISSN 2278 – 1021

*International Journal of Advanced Research in Computer and Communication Engineering*
*VOL. 1, ISSUE 1, MARCH 2012*

| Fetch Unit | Number of Fetches | Hit Ratio |
|---|---|---|
| 1 | 4496503 | 0.426 |
| 2 | 3473175 | 0.515 |
| 3 | 3099268 | 0.567 |
| 4 | 2964553 | 0.586 |
| 5 | 2861390 | 0.600 |
| 10 | 2723683 | 0.620 |
| 20 | 2658159 | 0.629 |
| 40 | 2527111 | 0.640 |

## V. SIMULATION AND EXPERIMENTAL RESULTS

The proposed algorithm has performed all experiments on a PC with Pentium IV and 1 GB RAM, under the Windows XP using J2EE. Comparison of hit ratio for client browser caching, client side proxy caching and server side proxy caching is shown in Table 4.3. It is clear that the Hit-ratio of the proposed server side proxy caching has improved compared to the existing methods.

Table III : Comparison of Cache Misses for CBC, CSPC and SSPC

| No.of Fetches | No.of Cache Misses for CBC | No.of Cache Misses for CSPC | No.of Cache Misses for SSPC | % of Cache Misses Reduced |
|---|---|---|---|---|
| 4496503 | 5198 | 5084 | 4134 | 18.3 |
| 3473175 | 4644 | 4351 | 3876 | 20.5 |
| 3099268 | 4148 | 3922 | 3529 | 21.7 |
| 2964553 | 4014 | 3776 | 3130 | 23.2 |
| 2861390 | 3891 | 3613 | 2868 | 24.5 |
| 2723683 | 3707 | 3482 | 2689 | 26.3 |
| 2658159 | 3563 | 3314 | 2435 | 27.7 |
| 2527111 | 3459 | 3198 | 2217 | 29.1 |

The Table III shows the comparative analysis of cache misses for CBC, CSPC and SSPC with different number of fetches. For the proposed SSPC, cache misses has reduced gradually for different number of fetches made.

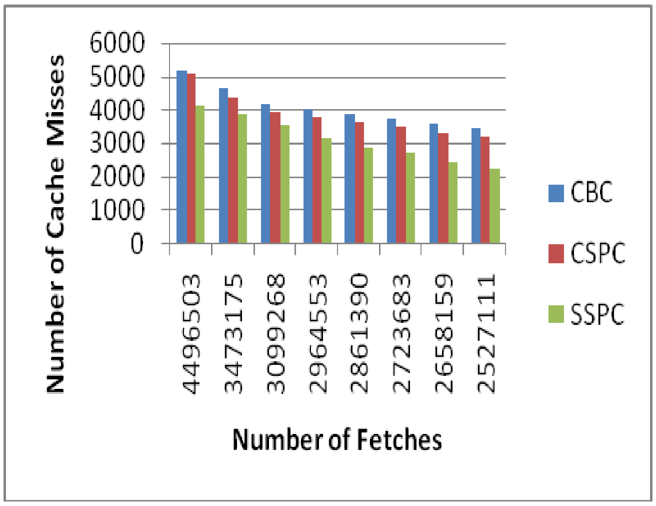

Fig. 2: Performance comparisons of CBC, CSPC and SSPC with number of fetches

The Figure 2 shows the Performance comparisons of CBC, CSPC and SSPC with different number of fetches made. It is clear that the Number of Cache Misses for the proposed SSPC algorithm has reduced to 18.3%, 20.5, 21.7, 23.2, 24.5, 26.3, 27.7 and 29.1 respectively for different number of fetches made. As it is shown in Table 4.3, the number of cache misses for SSPC method is smallest compared with the existing methods.

## VIII CONCLUSIONS

This paper has analyzed various types of web caching methods and implemented Server Side Proxy Caching (SSPC) web cache algorithm in order to reduce the cache misses and increase the hit ratio. The various design issues of web caching algorithms such as load balancing, cache miss, transparency, scalability and cache coherence were analyzed. The proposed algorithm's performance is compared with the existing methods namely Client Browser Caching and Client Side Proxy caching in terms of cache misses and the hit ratio. This chapter also has given the various needs and benefits of web caching.

## REFERENCES

[1] A.Belloum, B. Hertzberger; Maintaining web cache coherency, Information Research, vol, 6, No. 1, October 2009, pp.423-534.

[2] A. T. S. Ip, J. Liu, and J. C. S. Lui, "COPACC: An architecture of cooperative proxy- client caching system for on-demand media streaming," IEEE Transactions on Parallel and Distributed Systems, Vol. 18, 2007, pp. 70-83.

[3]I. Bose, H.K.Cheng; Performance Models of a firms Proxy Cache Server; Decision Support Systems and Electronic Commerce, 2000, No. 29, pp. 45-57.

[4]S. Chen, B. Shen, S. Wee, and X. Zhang, "Adaptive and lazy segmentation based proxy caching for streaming media delivery," in Proceedings of the 13th International Workshop on Network and Operating Systems Support for Digital Audio and Video, 2003, pp. 22-31.

[5] M. Busari; Comparison of Cache Replacement Algorithms in Web proxies; http://www.cs.usask.ca/faculty/carey/papers/ Muda 855.doc

[6] A. Feldman et. al.; Performance of Web Proxy Caching in Heterogeneous Bandwidth Environments, Proceeding of INFOCOM 99, 1999.

[7] L. Guo, S. Chen, and X. Zhang, "Design and evaluation of a scalable and reliable P2P assisted proxy for on-demand streaming media delivery," IEEE Transactions on Knowledge and Data Engineering, Vol. 18, 2006, pp. 669-682.

[8] Rousskov, A., and Soloviev, V., On Performance of Caching Proxies, Short version appears as poster paper in ACM SIGMETRIC'98 Conference, 1998.

[9]C. F. Kao and C. N. Lee, "Aggregate profit-based caching replacement algorithms for streaming media transcoding proxy systems," IEEE Transactions on Multimedia, Vol. 9, 2007, pp. 221-230.

[10] Q. Zhang, Z. Xiang, W. Zhu, and L. Gao, "Cost-based cache replacement and server selection for multimedia proxy across wireless Internet," IEEE Transactions on Multimedia, Vol. 6, 2004, pp. 587-598.

[11] Davison B. D.: A Survey of Proxy Cache Evaluation Techniques. In: Proceedings of the Fourth International WWW Caching Workshop (WCW99), San Diego, CA, March 1999.

[12] Mahanti A., Williamson C., Eager D.: Tra±c analysis of a Web proxy caching hierarchy.IEEE Network, 14, 3, May/June 2000, pp. 16-23.

[13] Zhendong Ma, J¨urgen Manglery, Christian Wagner, Thomas Bleier, (Aug. 2011), Enhance Data Privacy In Service Compositions Through A Privacy Proxy, Sixth International Conference on Availability, Reliability and Security (ARES 2011), 615 - 620 .