Towards An Efficient Web Caching Hybrid Architecture

Maha Saleh El Oneis, Hassan Barada, Mohamed Jamal Zemerly

Computer Engineering Department, Khalifa University of Science, Technology, and Research

P.O. Box 573, Sharjah, United Arab Emirates

maha@kustar.ac.ae, hbarada@kustar.ac.ae, jamal@kustar.ac.ae

ABSTRACT

As noticed by many internet service providers, the number of internet users is increasing exponentially. Such increase in information demand results in a variety of problems such as network congestion and server overloading which are perceived by the user as a delay in information retrieval. This paper reviews some of the common web caching architectures proposed by researchers to reduce the impact of these problems on the user. The paper also proposes a new architecture that is believed to have better performance than the previously proposed architectures as it combines the benefits of many schemes. Two possible scenarios of flow of information in the proposed architecture are presented. The first is a client-client, then client-proxy and proxy-proxy cooperation while the second is a client-proxy, then client-client and proxy-proxy cooperation. The scenarios will be tested using a simultion tool to assess the performance and fine tune the proposed architecture.

Key Words— Web caching architecture, broadband, load balancing, cache coherency, peer-to-peer

1. Introduction

The world today is moving on a fast technological track with the surfacing of various high-tech devices and techniques. Even with all our knowledge of the latest technologies and inventions, it is very difficult to keep track of everything happening in the world of technology. A huge leap has taken place ever since the launching of the World Wide Web (WWW). Today, information can be accessed anywhere, anytime and from a variety of different devices. With such renaissance in the technological sense, one might think that less human interference will be required. On the contrary, such advancement has led to more responsibilities for technical people in finding the right solutions for problems that arise, as well as creating a homogeneous environment for the user from a heterogeneous one [1,2,3].

As the WWW size continues to grow exponentially, some major problems that users face while trying to access large amount of information, includes network congestion and server overloading [4]. Network congestion can occur when a network link is carrying too much data that would affect its quality of service. Server overloading happens when the server receives more service requests than it can handle. One of the most appealing solutions to such problems was acknowledged in a variety of web caching techniques studied and developed by researchers since the early 90's. Web caching is the process of saving copies of content (obtained from the Web) closer to the end user, in order to reduce bandwidth usage, prevent server overload, as well as reduce the user's perceived latency. These studies have resulted in the development of the web caching notion which can be divided into three levels. Level1 (L1) cache is known as the client caching which takes place at the browser level. Level2 (L2) takes place at the proxy level while Level3 (L3) is the cooperation of the proxies in sharing cached objects among the cooperation set [5]. Researchers have agreed that caches on the client browser and proxy level can significantly improve performance [6]. In addition, many studies encouraged the broad use of web caching within organizations that provide internet service to users [7,8,9,10]. Such studies

helped in considering the possibility of constructing large web caches from cooperative proxies and client caches [11] to introduce a new cooperation level. So far this possibility is considered for Local Area Networks (LAN) within organizations with internet connectivity.

A range of studies agreed on the benefits of web caching and its major contribution to Internet services. Still the rapid growth of internet traffic and users has made us witness rapid improvements on the broadband services. Nowadays Internet Service Providers (ISPs) are offering better broadband networking technologies, but still many residential and small-business users are using low-bandwidth connections. Any near promise of the availability of such broadband technologies for users in rural areas is still uncertain because of the associated high cost. However, even with the availability of high bandwidth, there are types of information such as multimedia that always demand more bandwidth. Our area of interest in this research project is to reduce the client latency period in retrieving WWW information in rural areas as well as improving the performance of the broadband technology.

In addition to the obvious benefits of Web caching, some of the important properties desired in a web caching scheme are fast access, robustness, transparency, scalability, efficiency, adaptivity, stability, load balanced, ability to deal with heterogeneity, and simplicity [4].

The rest of the paper is divided as follows. Section 2 describes the main web caching architectures found in the literature and presents the features of each. Section 3 gives a description of a hybrid architecture that is believed to improve some of the drawbacks that are in the existing architectures. Section 4 gives a summary of the paper and the proposed contribution to the web caching architecture area.

2. Brief on web caching schemes

Ever since web caching has been found as a solution for the network congestion and server overloading, different caching architectures were proposed to ease the process of delivering the requested data through inter-cache cooperation.

2.1 Hierarchical Caching Architecture

The idea behind constructing a hierarchical cache is to arrange a group of caches in a tree-like structure and allow them to work together in a parent-child relationship to fulfil the requested objects by the client. If a hierarchical structure is arranged properly, the hit ratio can be increased significantly [4], see Figure 1.

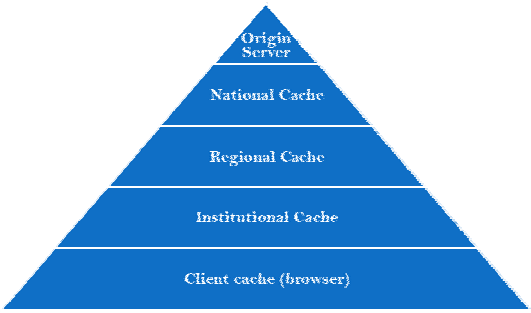

Figure 1. Hierarchical caching architecture

In a hierarchical caching architecture, caches are placed at various levels of the network. As shown in Figure 1, The hierarchy is constructed from the client's cache, institutional cache, regional cache, national cache, and the original server at the top. When a client requests a page, it first checks its browser cache. If the request is not fulfilled, then it is forwarded to the institutional cache. If the request is not satisfied by the institutional cache, then it is passed to the regional cache. If the request is not found at the regional cache, then it is redirected to the national cache. The national cache forwards the request to the original server if it cannot fulfill the request. When the object is found in a cache or the original server, it travels down the hierarchy and leaves a copy of the object in each caching level in its path to the client.

2.2 Distributed Caching Architecture

Researchers have proposed an alternative to the hierarchical caching architecture and eliminated the intermediate tiers except for

the institutional tier. All caches in that tier contain meta-data about the content of every other cache. Another approach proposed in this architecture is to employ the hierarchical distribution mechanism for more efficient and scalable distribution of meta-data [4], see Figure 2. A rough comparison between hierarchical and distributed caching is shown in Table 1.

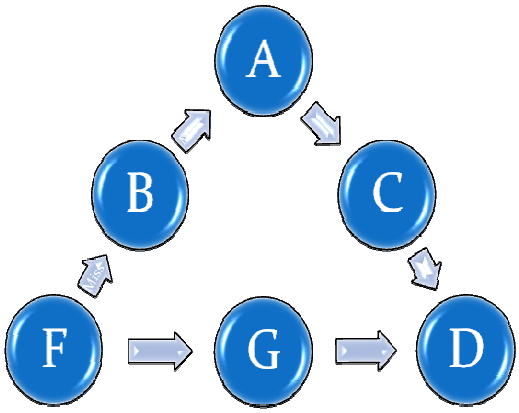

Figure 2. Distributed caching architecture

TABLE 1. COMPARISON BETWEEN HIERARCHICAL AND DISTRIBUTED CACHING ARCHITECTURES

Features	Hierarchical	Distributed
Parent caches	Congested	Slight congestion
latency	High	Low
Connection times	Short	Long
Bandwidth required	Low	High
No. of Hierarchies	< 4	1
Transmission time	High	Low
Network traffic	Unevenly distributed	Evenly distributed
Disk space usage	Significant	Low
Placement of caches in strategic locations	Vital	Not required
Freshness of cached contents	Difficult	Easy
Hit ratio	High	Very high
Response time	Moderate	Fast
Duplication of objects	High	Low

2.3 Hybrid Caching Architecture

A hybrid scheme is any scheme that combines the benefits of both hierarchical and distributed caching architectures. Caches at the same level can cooperate together as well as with higher-level caches using the concept of distributed caching [4].

3. Towards a better Architecture

The proposed architecture is a cooperative client-client, client-proxy, proxy-proxy caching system that aims to achieve a broadband-like access to users with limited bandwidth.

The proposed architecture is constructed from the caches of the connected clients as the base level, and a cooperative set of proxies on a higher level, as shown in Figure 3. The construction of the large client web cache is based upon some of the novel peer-to-peer (P2P) client web caching systems, where end-hosts in the network share their web cache contents.

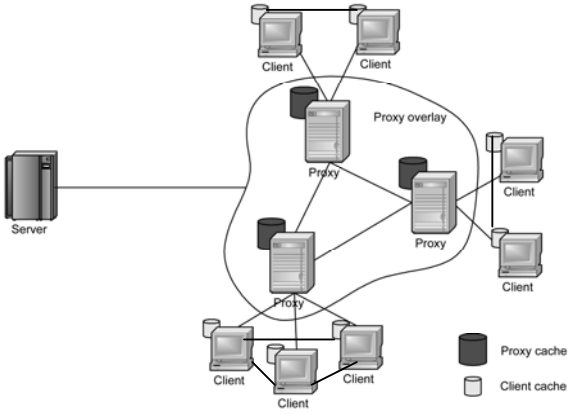

Figure 3. Proposed hybrid architecture

3.1 Desired properties in the proposed architecture

The proposed architecture is based upon the idea of a hybrid scheme. It consists of two tiers of cooperative caches, client caches and proxy caches. The properties that we wanted to achieve while designing the architecture are as follows:

- Slight congestion in the parent caches.
- Low latency and data transmission time.
- Evenly distributed network traffic for faster transmission time and low latency achievement.
- Long connection times.

- Low bandwidth usage which is the priority in this architecture along with the low latency property.
- A maximum of two hierarchical levels.
- Low disk space usage therefore low duplication of objects.
- Maintain an easy plan to keep the cached objects fresh.
- Test different object retrieval approaches to achieve a high - very high hit ratio and grant the user a fast response time.

3.2 Considerations and design issues

There are many challenges in the proposed approach since we are dealing with an unknown number of clients in an unstable environment. We have chosen to deal with the following issues:

3.2.1 Cache Communication

The main challenge in cooperative cache architecture is how to quickly locate the location of the requested cached object.

Malpani *et al.* [12] proposed a scheme where a group of caches function as one. When the user requests a page, the request is sent to some random cache. If the page was found in that cache, then it is returned to the user. Otherwise, the request is forwarded to all the caches in the scheme via IP multicast. If the page is cached nowhere, the request is forwarded to the home site of the page.

Harvest cache system [13] uses a scheme where caches are arranged in a hierarchy and uses the Internet Cache Protocol (ICP) for cache routing [14]. When a user requests a page, the request travels up the hierarchy to locate the cached copy without overloading the root caches by allowing the caches to consult their siblings in each level before allowing the request to travel up the hierarchy.

Adaptive Web Caching [15] builds different distribution trees for different servers to avoid overloading any root. This scheme is robust and self-configuring. It is more efficient with popular objects. For less popular objects, queries need to visit more caches, and each check requires a query to and responses from a group of machines. It is suggested to limit the number of caches the query visits, to decrease the added delay.

Provey and Harrison [16] propose a distributed caching approach to address the problems faced in the previously proposed hierarchical caching. They construct a manually configured hierarchy that must be traversed by all requests. Their scheme is promising in the way that it reduces load on top-level caches by only keeping location pointers in the hierarchy [4]. A simulation study was done as well, where the results showed that this proposed approach performs well for most network topologies. Results have also shown that in topologies where the number of servers in the upper levels is low, the performance of the hierarchical caching is better than the proposed approach. The conclusion of this paper is that the overall results show that there is no significant performance difference between the old and the proposed approach.

Wang [17] describes an initial plan in cache mesh system to construct cache routing tables in caches. To guide each page or server to a secondary routing path if the local cache does not hold the document. A default route for some documents would help to keep table size reasonable [4].

Legedza and Guttag [18] offer to reduce the time needed to locate unpopular and uncached pages or documents by integrating the routing of queries with the network layer’s datagram routing services [4].

3.2.2 Cache Coherency

The most outstanding benefit of web caching is that it offers the user lower access latency. The side-effect of providing the user with stale pages – pages which are out of date with respect to their home site. The importance of keeping the cache's content coherent is to provide the user with fresh and up-to-date pages.

Some of the proposed mechanisms to keep cache coherency are strong cache consistency and weak cache consistency [4].

- **Strong cache consistency**
  - *Client validation*. This approach is also called polling-every-time. The proxy initially considers the cached pages are expired on each access and sends an If-Modified-Since header with each access of the resources.
  - *Server invalidation*. When a resource is changed at the server, it sends invalidation messages to all clients that have recently accessed and cached the resource. The server has to keep a list of clients who requested and cached the changed resources which becomes unmanagable for the server when the number of the clients is large.

- **Weak cache consistency**
  - *Adaptive TTL*. The resource freshness problem is dealt with by adjusting the time-to-live parameter based on observations of its lifetime. If a file has not been modified for a long time, it tends to stay unchanged. Thus, the time-to-live attribute to a document is the current time minus the last modified time of the document.
  - *Piggyback Invalidation*. Whenever a cache has to communicate with the server, it adds along with it a list of resources that are potentially out-of-date and asks for validation.

3.2.3. Cache Contents

Proxy caches has been recognized as an effective and effecient solution to improve the web performance. A proxy serves in different roles: data cache, connection cache, and computation cache. A recent study has shown that caching Web pages at proxy reduces the user access latency 3% - 5% comparing to the no-proxy scheme [4].
It is very important to set the architecture and preapre it to deal with different types of resources. Most of the web resources are becoming synamic with the invasion of web services. It is very helpful to use computation caching to retrieve dynamic data. It can be done by caching dynamic data at proxies and migrating a small piece of computation to proxies to generate or maintain the cached data. Also the architecture should be able to retrieve information about the requested resource before adding delay to the request by looking for it in the caches when it is ana uncachable resource.

3.2.4. Load balancing

The hot spot problem is one of the issues that triggered the web caching research area. It occurs any time a large number of clients access data or get some services from a single server. If the server is not set to deal with such situations, clients will perceive a lot of delay and errors and the quality of service will be degraded. Several approaches to overcoming the hot spots have been proposed. Most use some kind of replication strategy to store copies of hot pages/services throughout the Internet; this spreads the work of serving a hot page/service across several servers [4]. Another approache that can be used is to set the server to work in a cooperative set with other servers or caches.

3.3 Flow of information in the architecture

The flow of information in the architecture can have different scenarios and paths. The two scenarios chosen for this architecture are as follows:

Scenario1: each client keeps a search history log of the clients that contacted it. When a client initiates a request it first looks into its local cache. If the requested page is found, then it is fetched from the local cache of the client. Otherwise, it looks into its search history log and search for the last client who requested this page. If found, it fetches the requested page from the client otherwise it consults the proxy to fetch the requested page. If the proxy found it in its cache, it forwards the requested page to the client. Otherwise, it consults the proxies in its cooperative set. If none have it, then the request is forwarded to the home server.

Scenario2: each proxy in the cooperative set is responsible of a group of clients that are geographically grouped. And each proxy acts as the leader of the peer-to-peer connected clients and contains cache and routing information of the client caches. When a client initiates a request, it will first check in its local cache. If it was found, then it is fetched from the local cache of the client. Otherwise, it will send the proxy a page location request. If the page was cached in one of the client's caches, then it would forward the information of the client that have the page in its cache to the requesting client. Otherwise, the proxy will consult the proxies in its cooperative set and check if any of the proxies have the requested page in its cache. If none have the requested page, then it is fetched from the home server.

Both of the mentioned scenarios are to be tested and analysed using a simulator. The simulation could result in the superiority of one of them or the need for a hybrid of both.

The reason such scenarios are chosen is to explore and benefit from the free space offered by the client's caches when they are connected to the internet. This architecture aims to reduce the load on the upper tier, the proxy, by initiating direct communication between the clients in P2P-like atmosphere which are geographically close to each other. The communications between the clients better stay as simple as possible as not to produce more delay and load on the client and organize the flow of the network traffic.

4. Conclusion

This paper presented web caching architectures that have been found as a solution for network congestion and server overloading problems. A rough comparison of the most common architectures was presented to show the pros and cons of each. The paper also proposed a new hybrid web caching architecture that is believed to offer a better performance which is due to combining the benefits of many architectures and schemes. The proposed architecture will look into some design issues such as the communication between the caches, the path to keep the caches' contents coherent, cache contents, and load balancing at the client and proxy side. Current work is on going to simulate the architecture's flow of information scenarios, using OMNET++ network simulator, to analyse and fine tune the architecture design issues.

Acknowledgment

This project is funded by Intel Innovation Center, Dubai, UAE.

References:

[1] P. Eaton, E. Ong, and J. Kubiatowicz, “Improving Bandwidth efficiency of peer-to-peer storage” *in Proceedings Fourth International Conference on Peer-to-Peer Computing*, 2004, pp. 80-90.

[2] B. Zenel, and D. Duchamp, “Intelligent communication filtering for limited bandwidth environments” *in Proceedings Fifth Workshop on Hot Topics in Operating Systems*, 1995, pp. 28-34.

[3] Li Fan, Pei Cao, Wei Lin, and Q. Jacobson, “Web prefetching between low-bandwidth clients and proxies: potential and performance”, Performance Evaluation Review, vol. 27, issue 1, p 178-187, June 1999.

[4] Jia Wang, “A Survey of Web Caching Schemes for the Internet”, *Computer Communication Review*, vol. 29, issue 5, pp. 36-46, Oct. 1999.

[5] S.G. Dykes and K.A. Robbins, “Limitations and benefits of cooperative proxy caching”, *IEEE Journal on Selected Areas in Communications*, vol. 20, issue 7, pp. 1290-1304, Sep. 2002.

[6] M. Abrahams, C.R. Standridge, G. Abdulla, S. Williams, and E.A. Fox,

"Caching Proxies: Limitations and potentials", *in Proceeding of the 4th International World-Wide Web Conference*, 1995, pp. 119-133.

[7] M.R. Korupolu and M. Dahlin, "Coordinated placement and replacement for large-scale distributed caches", *in Proceedings of the 1999 IEEE Workshop on Internet Applications*, 1999, pp. 62–71.

[8] S. Gadde, M. Rabinovich, and J. S. Chase, "Reduce, reuse, recycle: An approach to building large internet caches" *in Proceedings of the Workshop on Hot Topics in Operating Systems*, 1997, pp. 93–98.

[9] A. Wolman, G. Voelker, N. Sharma, N. Cardwell, A. Karlin, and H. Levy, "On the scale and performance of cooperative Web proxy caching" *In Proceedings of the 17th ACM Symposium on Operating Systems Principles (SOSP'99)*, 1999, pp. 16–31.

[10] K. W. Lee, S. Sahu, K. Amiri, and C. Venkatramani, "Understanding the potential benefits of cooperation among proxies: Taxonomy and analysis". Technical report, IBM Research Report, Sept. 2001.

[11] Y. Zhu, and Y. Hu, "Exploiting client caches: an approach to building large Web caches" *in Proceedings 2003 International Conference on Parallel Processing*, 2003, pp. 419-426.

[12] R. Malpani, J. Lorch, and D. Berger, "Making World Wide Web caching servers cooperate", *in Proceedings of the 4th International WWW Conference,* Boston, MA, 1995.

[13] A. Chankhunthod, P. B. Danzig, C. Neerdaels, M. F. Schwartz, and K. J. Worrel, "A hierarchical Internet object cache", Usenix'96, 1996.

[14] D. Wessels and K. Claffy, Internet cache protocol (IPC), version2, RFC2186.

[15] S. Michel, K. Nguyen, A. Rosenstein, L. Zhang, S. Floyd and V. Jacobson, "Adaptive Web Caching: towards a new caching architecture", *Computer Network and ISDN Systems*, 1998.

[16] D. Povey and J. Harrison, "A distributed Internet cache" *in Proceedings of the 20th Australian Computer Science Conference*, 1997.

[17] Z. Wang, "Cache mesh: a distributed cache system for World Wide Web", *Web Cache Workshop*, 1997.

[18] U. Legedza and J. Guttag, "Using network-level support to improve cache routing", *Computer Networks and ISDN Systems*, 1998 , pp. 2193-2201.