Prioritizing Documents and Applying Hybrid Caching Strategy for Network Latency Reduction

Benzir M. Ahmed[1], Tanjina Helaly, Syed Rahman

Department of Computer Science

North Dakota State University

Fargo, North Dakota 58105, USA

{Md.Ahmed, Tanjina.Helaly, Syed.Rahman}@ndsu.edu

Abstract

World Wide Web is the fastest growing applications on the Internet. The exponential growth of its traffic volume causes a great demand for bandwidth and server capacity, which in turn results in acute problems including network congestion, high bandwidth demands, high latency for extracting documents, and server overloading. Web caching has been recognized as an attractive solution to reduce network traffic and all the associated problems. By storing frequently accessed contents at a location closer to the user, a great deal of latency and unpredictable delay in the Internet can be eliminated. This paper proposes a hybrid-caching scheme where a certain number of caches cooperate at every level of a caching hierarchy using distributed caching to reduce latency for popular documents.

[1] Corresponding author

1. Introduction

WWW is the fastest growing applications on the Internet. Both the number of web users and the amount of web content accessed are predicted to accelerate dramatically in the years ahead. This is because the web is becoming a major center for business transactions of all types, with an increasing proportion of traffic taking the form of e-Commerce. The attraction of the WWW that has led to its exponential growth is that it allows people to access vast amounts of information from geographically distributed sites. In addition, the information can be accessed faster than it is possible by using other means. Moreover WWW has documents that are diverse in nature, and people can find information according to their interests. But this explosive growth of the web causes a great demand for network bandwidth, capacity and performance, which creates problems like slow connection speed, unpredictable performance, limitations in available bandwidth, overwhelmed web sites, etc. And finally contributes to the high latency for retrieving documents from the Internet.

The inefficiency of the Internet results from obtaining all contents directly from the server that is the original source for those contents. This is equivalent of having everyone fly to hollywood to see the latest movie. There is no distribution mechanism designed into the web that is analogous to the system of movie theaters that offer first-run films in every viewer's hometown. Scaling the Internet by simply adding more resources by increasing bandwidth and processing power may not be sufficient to provide reasonable reliability; rather it will continue to lag behind demand for the foreseeable future.

On the other hand, there are other mechanisms aimed at using available resources like server and bandwidth, more efficiently and reduce the latency to the users. Among them one way to improve performance and make it more predictable is to minimize the variability introduced by long trips across the Internet from the origin server to the user's browser. By storing frequently accessed content at a location closer to the user, a great deal of latency and unpredictable delay in the Internet can be eliminated. The way or technique for doing this is called caching.

There are billions of Web pages out there, but only a small fraction of those pages (or objects on pages) are requested frequently. That means many users request the same popular pages and objects. A simple example is the logo image at the top left corner of most Hotmail.com pages. This image must be delivered to a browser every time the browser accesses one of Hotmail's pages and these pages are requested many times a day. As many people are accessing the same server simultaneously, one effective solution is to use the existing network infrastructure to localize traffic patterns, enabling content requests to be fulfilled locally. Caching is a means of storing content objects from a Web server closer to the user, where they can be retrieved more quickly. The storehouse of objects, including text pages, images, and other content, is called a *Web Cache*.

The rest of the paper is organized as follows. In section 2, we briefly introduce the web caching. In section 3, we discuss network caching and existing network caching

architectures. In section 4, we present our hybrid caching architecture. And finally, in section 5, we conclude and discuss future works.

2. Web Caching

A Web cache is a dedicated computer system within the Internet that monitors Web object requests and stores objects it retrieves from a server. On subsequent requests for the same object, the cache delivers the object from its storage rather than passing the request on to the origin server. Every Web object changes over time, so each Web object has a useful life, or *freshness*. Caches determine whether or not their copy of an object is still *fresh*, or whether they need to retrieve a new copy from the origin server. The higher the number of people requesting the same object during its useful life, the more upstream traffic the cache eliminates.

By handling object requests rather than passing them upstream to the origin server, caches reduce network traffic and improve the browsing experience for users. Caches can be located anywhere on a network, and each cache will store a different set of objects or documents based on the needs of the users it serves.

Caching technology is already familiar in other applications. Many hardware devices cache frequently used instructions and data in order to speed processing tasks. For example, data that is frequently used by a computer's Central Processing Unit (CPU) is stored in very fast memory, sometimes right on the CPU chip, thereby reducing the need for the CPU to read data from a slower disk drive. In addition, Web browsers are designed to cache a limited amount of content on a user's PC. That is why selecting *Back* or *Forward* on a browser toolbar typically results in near instantaneous retrieval.

With Web caching, the same concept is applied more widely, using a server or specialized appliance. Web content is placed close to users in the form of a network cache, reducing the number of routing/switching hops that are required to retrieve content from a remote site. In other words, viewers aren't required to travel to Hollywood to see a movie, rather movies are sent to local theaters where people can watch them – or better yet, the viewers themselves determine which movies are made available locally.

Documents can be cached on the clients (i.e., browsers), the network (i.e., proxies or other intermediate network caches), and the servers. In this paper our only concern is the network level caching. More detailed description on web caching can be found in [2, 4].

3. Network Level Caching

To limit bandwidth demand due to the uncontrolled growth of Internet use, local caching is extended to the network level [3].

3.1. Types of Network Level Caching

The two current types of network-level caching products are proxy servers and network caches.

- Proxy Servers (non-transparent caching)
- Network Caches (transparent caching)

3.1.1. Proxy Servers

Proxy servers are software applications that run on general-purpose hardware and operating systems. A proxy server is placed on hardware that is physically between a client application, such as a Web browser, and a Web server. The proxy acts as a gatekeeper that receives all packets destined for the Web server and examines each packet to determine whether it can fulfill the requests itself; if it can't, it forwards the request to the Web server. Proxy servers can also be used to filter requests, for example, to prevent employees from accessing a specific set of Web sites.

Unfortunately, proxy servers are not optimized for caching, and they fail under heavy network loads. In addition, because the proxy is in the path of all user traffic (like a bump in the cable), few problems arise. All traffic is slowed to allow the proxy to examine each packet, and failure of the proxy software or hardware causes all users to lose network access. Further, proxies are not transparent to client, they require configuration of each user's browser which is an unacceptable option for service providers and large enterprises. Expensive hardware is required to compensate for low software performance and the lack of scalability of proxy servers.

3.1.2. Network Caches

In response to these shortcomings, *network caches* have been created. These caching-focused software applications are designed to improve performance by enhancing the caching software and eliminating the other slow aspects of proxy server implementations. Further, *network caches* are transparent to client, they do not require configuration of each user's browser. Transparent redirection technology, at key points within the network, enables network caches work by intercepting Hypertext Transfer Protocol (HTTP) requests (Transmission Control Protocol (TCP) traffic destined to port 80) and redirecting them to web cache servers of cache clusters. This style of caching establishes a point at which different kinds of administrative control are possible, for example, deciding how to load balance requests across multiple caches.

Network caches comprise the Web Cache Control Protocol (WCCP) [8] and one or more caches that are associated with the WCCP. WCCP is installed in routers, and it intercepts clients' request made to origin server, determines whether to forward it to the origin server or redirect it to any of the network caches.

The way network caches work transparently (Figure 1) is as follows:

1. First, a client sends a request for a web document to the origin server.
2. The router, running the WCCP, intercepts TCP port 80 Web traffic (i.e., HTTP traffic) and routes it to the cache engine. (The client is not involved in this transaction. And the client or browser requires no changes).
3. If the cache does not have the requested document, it sends the request to the origin server in the normal fashion. The document retrieved from the origin server is returned to, and is stored at the cache.
4. The cache returns a copy of the document to the client. Upon subsequent requests for the same document, the cache fulfills the request from the local storage.

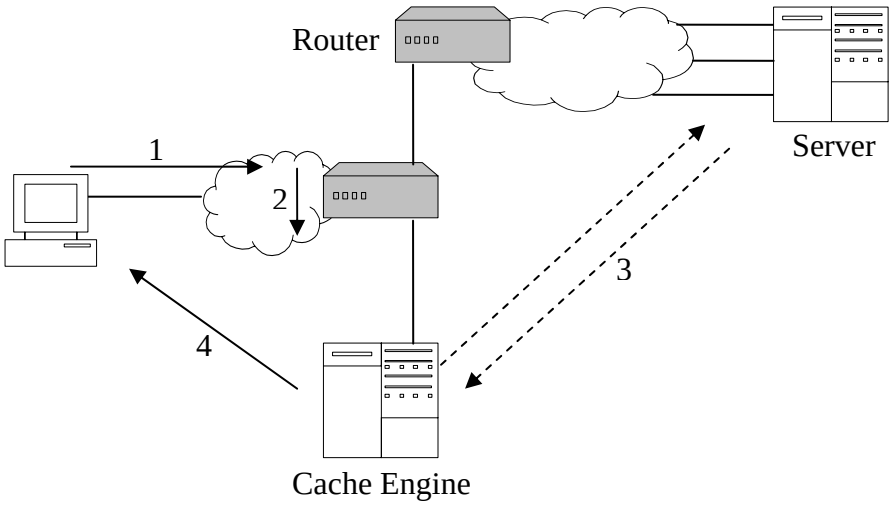

Figure 1: The figure shows how web cache operates.

3.2. Network Caching Architectures

The performance of a Web cache system depends on the size of its client community; the bigger is the user community, the higher is the probability that a cached document (previously requested) will soon be requested again. Caches sharing mutual trust may assist each other to increase the hit rate. A caching architecture should provide the paradigm for proxies to cooperate efficiently with each other. Two basic types of caching architectures are: hierarchical and distributed caching architecture.

3.2.1. Hierarchical caching architecture

One approach to coordinate caches in the same system is to set up a caching hierarchy (Figure 2). With hierarchical caching, caches are placed at multiple levels of the network. For the sake of simplicity, we assume that there are four levels of caches: bottom, institutional, regional, and national levels. At the bottom level of the hierarchy there are the client/browser caches. When a request is not satisfied by the client cache, the request

is redirected to the institutional cache. If the document is not found at the institutional level, the request is then forwarded to the regional level cache, which in turn forwards unsatisfied requests to the national level cache. If the document is not found at any cache level, the national level cache contacts directly the origin server. When the document is found, either at a cache or at the original server, it travels down the hierarchy, leaving a copy at each of the intermediate caches along its path. Further requests for the same document travel up the caching hierarchy until the document is hit at some cache level.

Hierarchical Web caching was first proposed in the Harvest project [12]. Other examples of hierarchical caching are Adaptive Web caching [11], Access Driven cache, etc. A hierarchical architecture is more bandwidth efficient, particularly when some cooperating cache servers do not have high-speed connectivity. In such a structure, popular Web pages can be efficiently diffused to-wards the demand.

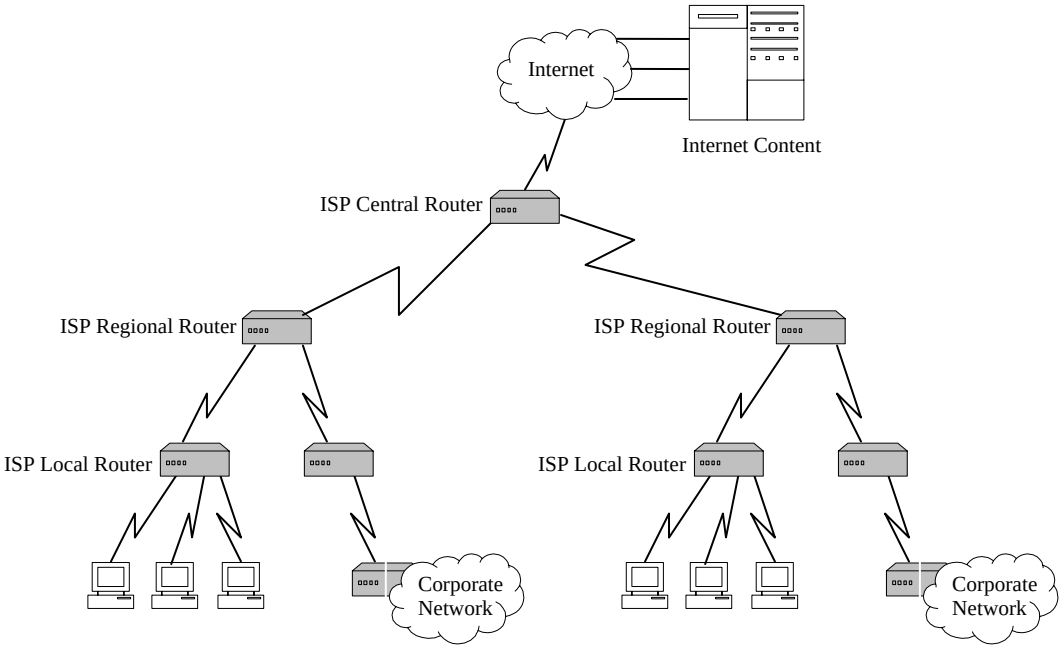

Figure 2: The hierarchical implementation of web cache.

However, there are several problems associated with a caching hierarchy:

1. To set up such a hierarchy, cache servers often need to be placed at the key access points in the network. This often requires significant coordination among participating cache servers.
2. Every hierarchy level may introduce additional delays.
3. High-level caches may become bottlenecks and have long queuing delays.
4. Multiple copies of the same document are stored at different cache levels.
5. Top-level cache of a caching hierarchy requires hundreds of GBytes to satisfy the needed capacity.

More detailed description on hierarchical caching can be found in [1, 5, 9].

3.2.2. Distributed caching architecture

In distributed Web caching systems (Figure 3), there are no other intermediate cache levels than the institutional caches, which serve each other's misses. In order to decide from which institutional cache to retrieve a miss document, all institutional caches keep metadata information about the content of every other institutional cache. To make the distribution of the metadata information more efficient and scalable, a hierarchical distribution mechanism can be employed. However, the hierarchy is only used to distribute directory information about the location of the documents, not actual document copies.

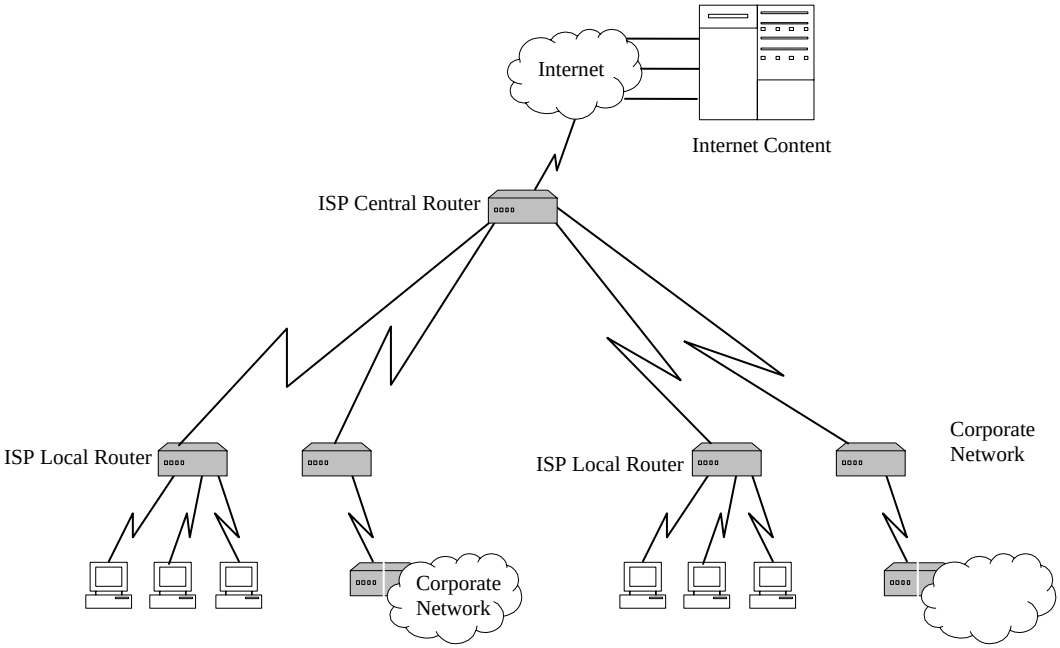

Figure 3: The distributed implementation of web cache.

With distributed caching, most of the traffic flows through low network levels, which are less congested and no additional disk space is required at intermediate network levels. In addition, distributed caching allows better load sharing and is more fault-tolerant. Nevertheless, a large-scale deployment of distributed caching may encounter several problems such as high connection times, higher bandwidth usage, administrative issues, etc.

There are several approaches to the distributed caching. One of them is the Internet Cache Protocol (ICP) [10] that supports discovery and retrieval of documents from neighboring caches as well as parent caches. More detailed description on hierarchical caching can be found in [1, 5].

4. Hybrid Caching Architecture

In this section we propose a hybrid-caching scheme that comprises the characteristics of the previously described two caching architectures, where a certain number of caches cooperate at the same network level or at a higher level of a caching hierarchy using distributed caching. ICP is a typical example. Our proposed hybrid-caching scheme tries to avoid the inefficiencies that are associated with those two architectures by implementing some modification in the conventional hybrid caching architecture.

4.1. Proposed Hybrid-Caching Scheme

In this scheme (Figure 4) clients' request is first redirected to the bottom level local cache. If the document is not found in the local cache, the cache checks if the document resides in any of the cooperating neighbors' caches that are considered by the WCCP. If multiple caches happen to have the document copy, the neighbor cache that provides lowest latency (i.e., the lowest Round Trip Time (RTT)) is selected. If the document does not reside among the considered caches at a given level, the request is then forwarded to the immediate parent cache or to the origin server. When the document is found in any of the caches or retrieved from the origin server, it travels down the hierarchy, leaving a copy at each of the intermediate caches along its path.

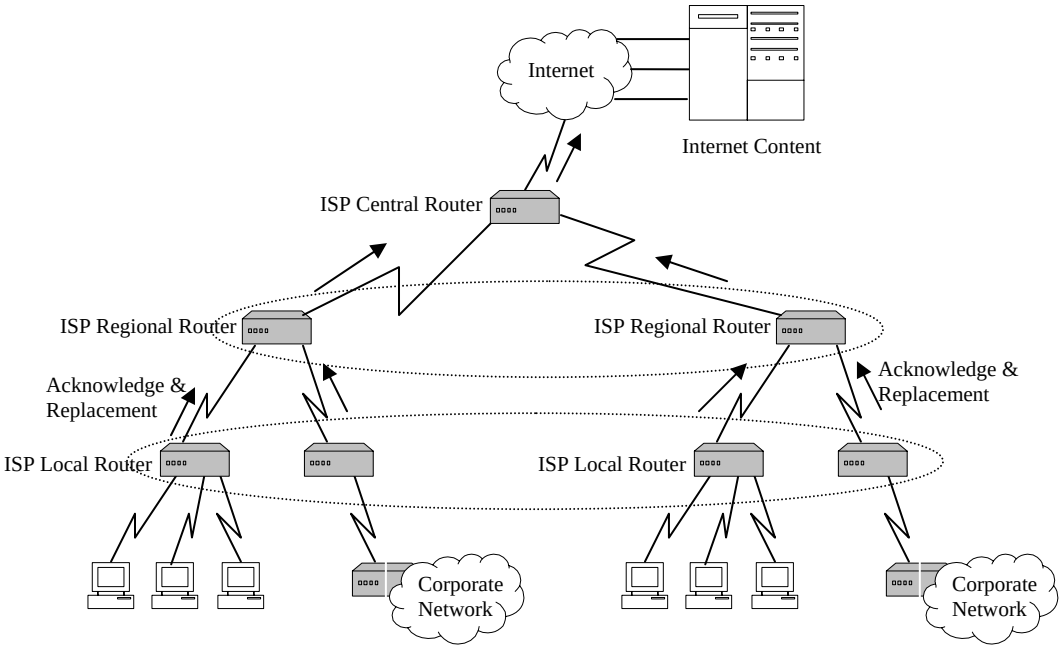

Figure 4: The hybrid implementation of web cache.

To avoid having multiple copies of the same document in different levels, every child send acknowledgement to its parent when it receives a copy. And the parent removes its

copy of the document when gets acknowledge from its child. So it results in the existence of the copy of a document only in the bottom level. It also results in reduction of disk space requirement in the top-level cache of a caching hierarchy.

In our approach most popular objects are kept at the bottom level caches. Less popular documents propagate up the hierarchy. A document may loss its popularity in its lifetime. So we define a popularity parameter (popularity threshold) that determines when a document that losses its popularity will be removed from the cache. If one document's popularity reduces below the popularity threshold then that cache also checks whether any of the caches of the same level has a copy of the same document. If any of them has that document, it will only remove it from its storage, but if none of them has it then it will send a copy of that document to its parent while removing it from its own storage.

The number of caches that should be resided at every cache level should be limited to avoid high latency. We have to investigate the optimal number of caches that should cooperate at every cache level to minimize the document retrieval latency.

4.2. Performance of Hybrid Caching Architectures

It is comprehensible that our proposed hybrid-caching scheme with an optimal number of cooperating caches at every level improves the performance of hierarchical and distributed caching. The improved performances are:

1. Reduced connection and transmission time.
2. Minimized total latency.
3. Reduced traffic and bandwidth usage of the Internet.
4. Avoided occurrences of multiple copies in different levels of the hierarchy.
5. Reduced disk requirements for the caches, especially for the top level caches.
6. Redistributed load at every cache level, so reduced load on the caches even in the top-level.
7. Reduced load on the server.

6. Conclusions and Future Work

Web caching is recognized to be one of the effective techniques that plays a critical role in improving content availability, addressing increasing bandwidth demand, alleviating network latency and server load. Caches need additional resources and placement in the network. The placement of caches will eventually happen because ISPs are very interested in saving network bandwidth and in reducing the latency to their receivers.

In this paper, we have proposed Hybrid caching architecture for improving caching that has the lowest latency delivery time for popular documents. A hybrid-caching scheme combines the advantages of both hierarchical and distributed caching, reducing the connection time as well as the transmission time. The optimal number of caches that

should be placed at each level depends on the number of hops from the client to the caches, the congestion of the network, the parent cache/server load, and the document's size.

Although the scope of our proposed caching architecture has been limited to the popular but static documents, we hope to address the challenges in the caching of popular but highly changing documents, unpopular or dynamically generated documents in the future. And we will also implement our proposed architecture to investigate about the optimal number of caches that can reside in one level.

References

[1] Rodriguez, P., Spanner, C., Biersack, and E., *Analysis of Web Caching Architectures: Hierarchical and Distributed Caching*, IEEE/ACM Transactions on Networking, Vol. 9(4):404-418, August 2001.

[2] Barish, G. and Obraczka, K., *World Wide Web Caching: Trends and Techniques*, IEEE Communications Magazine, 38(5):178-184, 2000.

[3] Legedza, U. and Guttag, J., *Using Network-level Support to Improve Cache Routing*, In the Proceedings of the 3rd International WWW Caching Workshop, Manchester, England, June 1998.

[4] Intel Corporation, *Caching for Improved Content Delivery*, White paper, June 2002.

[5] Wang, J., *A survey of web caching schemes for the Internet*. ACM Computer Communications Review, 25(9):36–46, 1999.

[6] Ford, M., Kim Lew, H., Spanier, S., and Stevenson, T., *Internetworking Technology Overview*, Second Edition, Cisco Press, June 1999.

[7] Chandhok, N., *Web Distribution Systems: Caching and Replication*, ftp://ftp.netlab.ohio-state.edu/pub/jain/courses/cis788-99/web_caching/index.html, July 2002.

[8] Cieslak, M., Forster, D., Tiwana, G., and Wilson, R., *Web Cache Coordination Protocol V2.0*, Internet Draft, draft-wilson-wrec-wccp-v2-00.txt, IETF, 13 July 2000.

[9] Che, H., Wang, Zz., and Tung, Y., *Analysis and Design of Hierarchical Web Caching Systems*, In the Proceedings of IEEE INFOCOM'01, Vol. 3:1416-1424, May 2001.

[10] Wessels, D. and Claffy, K., *Internet Cache Protocol*, RFC 2186, September 1997.

[11] Michel, S., Nguyen, K., Rosenstein, A., Zhang, L., Floyd, S., and Jacobson, V., *Adaptive web caching: towards a new global caching architecture*. In Computer Networks and ISDN Systems, November 1998.

[12] Bowman, C., Danzig, P., Hardy, D., Manber, U., and Schwartz, M., *Harvest: A Scalable, Customizable Discovery and Access System*, Technical Report CU-CS-732-94, Department of Computer Science, University of Colorado – Boulder, August 1994.