Web Caching System by Using Adaptive Replacement Cache Algorithm

Nu Nu Aung, Mar Mar Min
University of Computer Studies, Yangon, Myanmar
Aries283@gmail.com, mptn28@gmail.com

Abstract

Web is fundamentally a very simple idea. Web server that accept request from web Clients for pieces of information called web objects. When a user accesses a web page, the request typically goes out to the Internet to the server on which the page resides. The page content is then returned back across the Internet to the user's workstation. Web Caching works by storing the content of requested web pages on a dedicated caching server inside the network. The system would apply ARC(Adaptive Replacement Cache)Algorithm for caching another web pages when the cache is full. When the client makes a request , the browser is then checked whether it is in server cache or not and request information's frequency and last access time is recorded in the database. If it is in its sever cache after checking, the web information is displayed on the browser. Next step is considering how to cache the web objects and compare the size of the requested page and exciting cache size whether the web object has to be saved in server cache or not. Applying this algorithm, this system reduce network traffic and user latency.

1. Introduction

In the last decade intensive growth of information volumes in the World-Wide-Web has led to the problem of network loads. Development of technical means for transmitting data and their introduction into practice do not correspond with the Internet growth. Hence, there is a need to find other approaches to solve the problem of network loads. The common method of solution without drawing ad-additional technical means is via caching Web objects (text, image, etc.). Web documents are subdivided into two types: static and dynamic. Most of the Web objects are static documents which can be stored in certain data 'storage', i.e a cache, for the further usage. While reassessing these documents the proxy server checks whether there have been any modifications on the source site and if there have not been any, the user has the page downloaded right from the cache. Usage effectiveness of caching allows to use it on different levels such as Internet browser, proxy server of a local net, Internet proxy server. Proxies serving a large set of clients show the highest effectiveness of caching. It is connected with the fact that many Web users have correlation in requests (common interests).Researches have shown that 25-40% of all requested documents account for 70% of users' requests. Moreover, the nets of homogeneous organizations (university, corporation, etc.) have the highest correlation of requests. While caching Web objects on proxy servers as well as in other aspects of caching there appears a problem of finite storage of the cache and hence, a necessity of making room for new documents.

A replacement policy determines which object is to be removed from the cache. Selection of an effective policy can considerably increase caching increase caching effectiveness reducing network traffic by 20% and more. Difficulties in selecting a replacement policy are connected with specific characteristics of the network traffic which are the following:

-HTTP protocol gives access only to files of full size, i.e. a proxy server cache can satisfy the user's request only if the file has been stored completely (i.e. it can not store incomplete objects).

-Documents stored in the proxy server cache are of different size, from several bytes to hundreds of megabytes.

-A stream of requests to the cache is a sum of streams of requests of hundreds and thousands of users.

Because sizes of documents are different there is a need to introduce a new metric of effectiveness called Byte Hit Rate apart from Hit Rate policy as the basic metric of effectiveness. Byte Hit Rate is computed as a ratio of amount of bytes derived from the proxy cache to the total amount of bytes requested by the user. Because of difference in documents sizes metric suggested by Byte Hit Rate is the most adequate to deter-mine the policy effectiveness as it is the one that shows the savings in network traffic.

The first time, a page is accessed; the contents are returned and stored on the caching server. On subsequent requests for the same web page, the cache delivers the page from its local (or cached) storage rather than going out across the Internet link to the originating web site. Every web page changes over time, so each cached page has a useful life, or "freshness". This next section includes the related work. In Section 3 describes Background option: Web caching System. The proposed Scheme in section 4. In section 5,we conclude the paper.

2. Related Work

Brian D. Davison [3] pointed out three features of Web caching make it attractive to all Web participants, including end users, network managers, and content creators. Caching reduces network bandwidth usage, which can save money for both content consumers and creators. It lessens user-perceived delays increases user perceived value and lightens loads on the origin servers. It is saving hardware and costs for content providers and providing consumers to get the shorter response time for non-cached resources.

Mudashiru Busari and Carey William Son proposed for trace-driven simulations to evaluate the performance of different cache management techniques for multilevel Web proxy caching hierarchies. They considered heterogeneous cache replacement policies within a two level caching hierarchy, and size based partitioning across the levels of a caching hierarchy. Three different synthetic Web proxy workloads are used in the study, reflecting complete overlap, partial overlap, and no overlap in the workloads seen by the child-level proxies. The simulation results demonstrated that heterogeneous replacement policies and size-based partitioning each offer modest improvements in caching performance [4].

In paper [5], Guangwei Bai and Carey William Son reported for the Web cache simulation step with two experimental factors: cache size, and cache replacement policy. The cache size determines the maximum number of Web content bytes that can be held in the cache at one time.

The cache replacement policy determines what objects to remove from the cache when more space is needed to store an incoming object. Six cache replacement policies are considered: removing objects at random (RAND), removing objects in the order in which they arrived (First-In-First-Out, FIFO), removing objects based on recent use (Least-Recently-Used, LRU), removing unpopular objects (Least-Frequently-Used, LFU), removing large objects (Greedy-Dual-Size, GDS) and Adaptive Replacement Cache (ARC).

A.Mahanti and C.William Son proposed that the common workload characteristics observed include a high degree of one-time referencing, a Zipf-like document popularity distribution, heavy-tailed file and transfer size distributions, and a temporal locality property in the document referencing behavior [7, 8]. Among these characteristics, the slope of the Zipf-like document popularity distribution is most relevant to Web caching performance [8].

3. Background Option: Web Caching System

Caching has become a significant part of the Web's infrastructure. Caching has even spawned a new industry such as content delivery networks which are also growing at a fantastic rate. This system is familiar with relatively advanced Web caching topics such as the Internet Cache Protocol (ICP), invalidation, and interception proxies, so, now we would like to describe caching, explains how it applies to the Web, and describes when and why it is useful.

3.1 Caching Web Resources

Davison presented that web caching is similar to memory system caching. Web caching system stores web resources in anticipation of future requests. However, it will provides several web caching result from the non-uniformity of web object sizes, retrieval costs, and cache ability because of depending on the configuration of cache memory. To address object size, cache operators and designers track both the overall object hit rate (percentage of requests served from cache) and the overall byte hit rate (percentage of bytes served from cache)[3]. This paper can reveal that traditional replacement algorithms often assume a fixed object size, so variable sizes can affect their performance. And also retrieval cost varies with object size, distance traveled, network congestion, and server load. Finally, some Web resources cannot or should not be cached, for example, if the resource is personalized to a particular client or is constantly updated. Caching is performed in various locations throughout the Web. These locations include two endpoints. They are Web browser and Web server.

Web caching system may be used because of popularity - the more popular a resource is, the more likely it is to be requested in the future. In one study spanning more than a month, out of all the objects requested by individual users, on average close to 60 percent of those objects were requested more than once by the same user. Likewise, much content is the value to more than one user. In fact, the hits recorded in another caching study, up to 85 percent was the result of multiple users requesting the same objects.

Guangwei Bai and Carey William Son explained that the caching effectiveness is traditionally measured by two quantities: the (document) hit ratio is the percentage of the total requests that are satisfied directly by documents stored in the cache and the byte hit ratio is the percentage of the total requested Web content bytes that are satisfied directly by documents stored in

the cache. Both metrics are required since Web objects vary significantly in size. Other metrics such as user-perceived response time are dependent upon the hit ratio and the byte hit ratio, as well as network bandwidth, round-trip delay, and server load[5].

3.2. Potential Problem

This system has to face a number of problems associated with caching. First possibility is the end user seeing stale (that is, old or out-of-date) content, compared to fresh content available on the origin server. HTTP does not ensure strong consistency and thus there is a real potential for data to be cached too long. Second, misses that are processed by a cache generally have decreased speed, as each system through which the transaction passes will increase the latency. Caching tends to improve the latency only for cached responses that are subsequently requested. So, web caching system can be able to increase the percentage of hit rate. Thus, a cache only benefits requests for content already stored in it. Caching is also limited by the frequency with which popular Web resources change, and, importantly, the fact that many resources will be requested only once. Finally, some responses cannot or should not be cached.

To overcome these potential problems, system must be give expiration date far to all static content (buttons, graphics, audio and video files, and pages that rarely change) so that they can be cached for weeks or months at a time. By setting an expiration date far into the future, the content provider trades the potential of caching stale data for reduced bandwidth usage and improved user-perceived response time. A shorter expiration date reduces the chance that the user sees stale content, but increases the number of times that caches will need to validate the resource.

4. The Proposed Scheme

This section is going through the important stages of web caching system with clear explanation.

This system uses a cache for web storage. When the client makes a request, the request on the browser is then checked whether it is in its local cache or not, and the request information is recorded in the database. If it is in its sever cache after checking, the web information is displayed on the browser. But if the request web object is not update in server cache, the sever cache is updated with the requested web object and display desire web page on the browser by directly using the internet. If the request is in Exception, update sever cache. If the request is not in Exception, it will be sent to the server cache and display on the browser by directly using the internet . The system would apply ARC(Adaptive Replacement cache) Algorithm for caching another adding web pages when the cache is full. Next step is considering how to cache the web objects and defines the threshold value first to decide whether the web object has to be saved in the server cache or not. If the user is using any web object frequently, system calculates the utility for it.

Web Filtering Algorithm (For each request)

```
Let f= frequency of request
Let t= current time of request

begin
  while(cache is not full) do
    if (url not include in Exception Table) then
          add this web site into the cache;
                f=1;
    else
          overwrite this web site into the cache
                 f=f+1;
    end if
    t=set the current time
  end if
end
```

The last step would describe the Least Recently Used algorithm. This algorithm removes web objects which have minimum requested frequency value (utility), so that popular web sites can remain in the cache.

Adaptive Replacement Algorithm

```
begin
    if ( cache is full ) then
        while(not end of cache) do
          begin
             search web site w which has min f & et
             if(founded sites are more than one) then
                begin
                    search biggest site
                end
          end
          remove w form the cache
    end
end
```

4.1 Case Study

4.1.1 System performance by user latency

Base on testing, Web Caching System reduces the traffic crossing the LAN by over 85%. It improves the data access performance significantly. For example, one result shows that over a connection with 100ms delay, a 2MB

size web page that took 20 seconds to open normally took only 3 seconds with the web caching system. Web caching system reduces web page access times considerably.

In Figure.1 shows the user latency different between using WCS and without WCS.

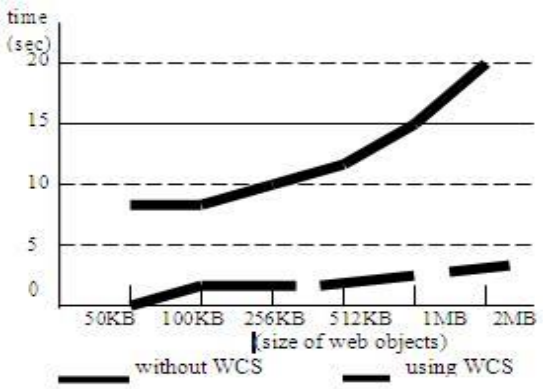

Figure.1: Reduction user latency

4.1.2 System performance by cache size

Caching effectiveness is traditionally measured by a quantity: the (document) hit ratio is the percentage of the total requests that are In the large size, calculation cost for web filtering is high at the start because there are only a few amounts of web objects in t satisfied directly by documents stored in the cache. The formula for the hit ratio is: hit ratio= number of hits / (number of hits + number of miss). So, we calculate number of miss by testing the system many times. Then, we can calculate hit ratio.

But, calculation cost for web filtering will be decrease in the later because of increasing popular web sites in the cache. And, it will be increase system performance (reduce latency, bandwidth, network utilization, server load). Because of cache size is large; it will reduce computational cost for cache replacement process. System can give highest hit rate that user requested web pages are consistent with the cache. If the cache is full, system starts replacement process (ARC).

But this system will do replacement process in a few times because it is not possible to reach easily full state of large cache. If the cache size is large, a large amount of web objects will be stored in the cache so that hit rate will increase.

In Figure.2, shows the hit ratio of this system in the situation of using the large cache size. Hit rate of the system increases in the later because many popular web sites can be stored in the cache.

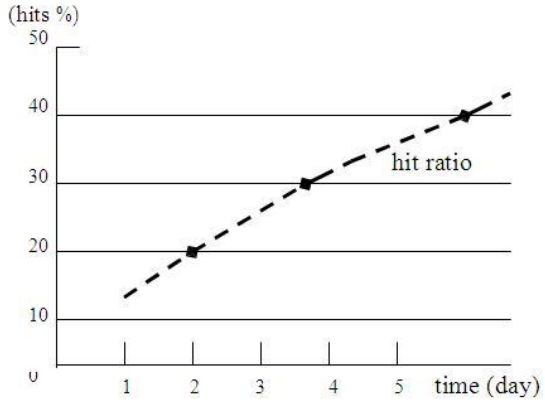

Figure.2: Hit ratio of web caching system in large cache size

If the system uses the cache which has small size, only a small amount of popular web sites can store in the cache so that calculation cost for web filtering will be high. It can not increase system performance by using small cache size because miss rate will be high in the later. The system will do replacement processes in many times because it is possible to reach easily full state of the cache by using the small cache size. But, it can reduce cost for accessing web objects from the cache because only a small amount of web objects will remain in the cache.

In Figure.3, shows the system performance (hit rate) in the situation of using the small cache size. Hit rate of the system can not increase because cache size is not enough to hold large amount of popular web sites.

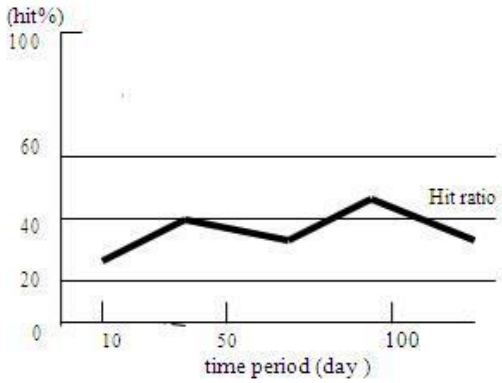

Figure.3: System performance (hit rate) in small cache size

4.1.3. User environment

User environment is also important for measuring system performance. System performance also depends on the quantity of users. There will be two kinds of users (frequent user and guest user). User who frequently used this system is called frequent user. They will have same interest, inspirations or behaviors. They make convenient requests for the system.

So, system can reduce storage space and system loads. If the quantity of frequent users increases, system can reduce computational cost and can give highest performance because of increasing hit rate.

User who sometimes uses this system is called guest user. It is impossible to concur with the web objects stored in the cache. If the quantity of guest users increases, calculation cost will be high and decrease hit rate. So, system performance will be lower than frequent user environment.

4.2 Benefit of the system

This system can provide benefits to all parties on the web users, web proxy, ISPs, and original web servers. It can reduce user latency, bandwidth, network traffic and server loaded. When internet connection failed, this system can continuously serve as online access by responding required web objects which had been stored in the cache. Therefore, users can access their requested web objects in the offline state.

5. Conclusion

As it is with any evolving technology , Web caching techniques are changing rapidly especially being of interest to both the Caches positioned near or maintained by the content provider improve access to a logical set of content. In this thesis we represented into effectiveness of ARC policy for the purposes of caching on a proxy server. In comparison with popular replacement policies, ARC algorithm shows higher efficiency. The results of the experiments allow to make a conclusion about expediency of ARC algorithm on proxy servers in the Web. As privacy of transactions becomes more important and harder and harder to achieve, it may be worthwhile to use caching proxies achieve anonymous access to what may be the most important achievement of the human civilization – written text information.

References

[1] Jon Henrik Bjornstad, University of Oslo, Department of Informatics, Traffic Characteristics and Queuing Theory: Implications and Applications to Web Server Systems, Master Thesis, May 22, 2006.

[2] Ping Du Jaspal Subhlok, Department of Computer Science, University of Houston,Houston,TX77204, {pdu,jaspal}@uh.edu, Evaluation of Performance of Cooperative Web Caching with Web Polygraph

[3] Brian D. Davison, A Web Caching Primer, Department of Computer Science, Rutgers, The State University of New Jersey (USA), http://www.cs.rutgers.edu/davison/davison@cs.rutgers.edu

[4] Mudashiru Busari, Carey William Son, Simulation Evaluation of a Heterogeneous Web Proxy Caching Hierarchy, Department of Computer Science, University of Saskatchewan
Email:carey@cs.usask.ca

[5] Guangwei Bai, Carey William Son, “Workload Characterization in Web Caching Hierarchies”, Department of Computer Science, University of Calgary{bai,carey}@cpsc.ucal.gary.ca

[6] P. Barford, A. Bestavros, A. Bradley, and M.Crovella, Changes in Web client Access Patterns : Characteristics and Caching Implications, World Wide Web, Vol 2,No.1, January 1999,pp 15-18

[7] M. Busari and C.William Son, On the Sensitivity of Web Proxy Cache Performance to Workload Characteristics, Proceedings of IEEE INFOCOM, pp.1225-1234, Anchorage, AL, April 2001.

[8] A. Mahanti, C. William Son and D. Eager, Traffic Analysis of a Web Proxy Caching Hierarchy, IEEE Network, Vol.14,No.3,pp. 16-23, May/June 2000.