International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 17 (2018) pp. 13386-13391

Enhanced Bloom Filter Technique for Latency Reduction in Mobile Environment

P. Amudha Bhomini[1], Dr. Jayasudha J.S[2]

[1]Research Scholar, St. Xavier's College, Palayamkottai, Manonmaniam Sundaranar University, Abishekapatti, Tirunelveli-627012, Tamil Nadu, India.

[2]Professor and Head, Department of Computer Science and Engineering, Sree Chithra Tirunal College of Engineering, Trivandrum, India.

Abstract

The rapid growth of internet causes increase in web traffic. This becomes major issue in the network world. No user has patience in waiting more than few seconds for downloading a web page. Thus reducing the download latency of web documents is crucial for web users and web content providers. Caching is one of the primary mechanisms for lessening the latency as well as bandwidth requirements for delivering web content. Web cache replacement policies decide which data is to reside in cache and objects that must be removed from the cache to make space for new ones. Prefetching is also used to reduce the latency by finding the anticipated web pages and fetching it before actually required.

This paper gives a methodology for caching using Message Digest (MD5) and bloom filter (BF) techniques, replacing the web pages using Least Recently Used (LRU) integrated with First in First Out (FIFO) algorithm and prefetching web pages based on user log and bandwidth which is adequate enough to access the documents instantly through the internet. These caching and prefetching techniques are highly recommended to increase cache hit ratio and to reduce latency for accessing the web pages instantly.

Keywords: Message Digest (MD5), Bloom Filter (BF), Least Recently Used (LRU), caching prefetching, bandwidth

INTRODUCTION

Wireless mobile communication is an expeditious growing sector in communication industry. In mobile communication, Internet plays a major role. Evolution of internet over these years has been accelerating the web traffic. It is being a biggest challenge to overcome the web traffic. By storing the data in cache future requests for that data can be served faster. Web caching techniques make use of temporal and geographical locality principle to cache the most frequently used web objects near the clients. The cache hit occurs when the required data can be found in a cache, while a cache miss occurs when not.

In mobile caching the contents are stored in the local cache of mobile memory while downloading. These contents can be shared with other users via Bluetooth or WIFI. Psephos is a mechanism used for determining the caching policy of each mobile user. In this technique users compute their own policies individually in distributed manner. Neighbor Graph cache mechanism is used to reduce scanning latency while a mobile station tries to make a link-layer handover [1]. A novel caching optimization mechanism is available for Gnutella network to meet the mobility and the network requirements.

Web browsers and web proxy servers bring web caches to store preceding replies from web servers such as web pages and images. Web caches lessen the extent of information that has to be carried across the network as information preceding stored in the cache that can usually be re-used. This reduces bandwidth and processing requirements of the web server and helps to enhance the responsiveness for users. Each web browser has built-in web cache, but some Internet Service Providers (ISPs) or organizations also use a caching proxy server, which is a web cache that is shared among all users of that network [2]. Locally cached objects ensures faster browsing experience to the clients since caching can be done everywhere including personal machines, servers and telecommunication companies, thereby reducing the overall network traffic [3]. Reducing network latency in mobile applications is an effective way of improving the mobile user experience and has tangible economic benefits [4].

To be cost-effectual and to facilitate effective use of data, caches should be small. Cache hits are provided by viewing data from the cache, which is faster than estimating a result or viewing it from a data store. Thus, the more request acknowledged from the cache, the faster the system operates. When the cache consumer wish to access data pretending to exist in the store, it initially checks the cache. If an entry is brought in with a tag matching with that of the desired data, the data in the entry is used on behalf of the desired data in the entry. This is known as a cache hit. If the cache is conferred and discern not to attain data with the resulted tag has become known as a cache miss [5]. As the storage capacity of mobile devices is very limited, only less amount of data can be stored in the web cache of a mobile device [6]. Thus replacement of document is necessary when the cache becomes full.

Caching and prefetching techniques are complement to each other in terms of locality of references. But they can be integrated together for reducing latency and increasing cache hit ratio. Integrated prefetching and caching schemes are available in mobile environment. Adaptive network schemes

and online learning mechanisms are available for prefetching files for mobile devices [7]. Predictive web prefetching refers to the mechanism of finding the forthcoming page access of a client based on its past accesses [8]. Throughput is the time evaluated in seconds, which comprises the total time required for the log file cleaning, log entries categorization, mapping and prefetching [9].

The rest of this paper is organized as follows: Section 2 describes enhanced Bloom filter technique, Section 3 discusses performance evaluation and Section 4 presents concluding remarks.

PROPOSED ENHANCED BLOOM FILTER TECHNIQUE

The Uniform Resource Locator (URL) of a desired document is given as input. Divide the URL into two parts namely domain name and path with file name. Hexadecimal values are obtained when message digest algorithm (MD5) is applied by appending padding bits, appending length, initializing buffer and processing the message in blocks. Then the hexadecimal values are converted into binary values of 256 bits. The incoming bit pattern which corresponds to 1's in the bit positions are compared with the existing Bloom filter array. If it is same with the existing one it means that the web page is already available in cache or prefetch area. So, no need to store the bits in Bloom filter array otherwise change the Bloom filter array by changing the corresponding bit position of Bloom filter array to 1, if it is already 0. The request is forwarded to the server to get the requested web page. Accumulate the content of the URL in cache database with their corresponding weight set to 1.

The hit count gets increased if the desired content is obtained from cache and the weight of the corresponding URL is incremented by 1 else miss count gets increased. The weight of document in the cache gets reduced periodically for excluding the older web pages in the cache. If the cache is full and another URL comes in, the URL with the least weight will be deleted. When two URL's have same weight; then the First in First Out (FIFO) algorithm is used for deletion. Thus, LRU integrated with FIFO is used for page replacement in the cache. Cache consistency is also checked frequently. While checking, If modified since header has the status code value as 200 indicates that the modification is relevant. Depending on the status code value the database is updated. For prefetching total number of links in a URL is calculated. Links are downloaded based on user log, number of links to be downloaded and available bandwidth. If any link needed is not in the prefetch area, when prefetch area is full the existing one with the least weight is deleted to accommodate the newly prefetched web page.

The different processes involved in caching/prefetching using enhanced bloom filter technique are given below

i. Monitoring the user access pattern
ii. Apply MD5 algorithm for each URL
iii. Convert the hex value into its equivalent binary value
iv. Maintaining the bloom filter array
v. Calculating the cache hit ratio and maintaining cache
vi. Prefetch the anticipated web pages based on user log and bandwidth and store it in prefetch area
vii. Adopt LRU integrated with FIFO as the cache replacement policy

Enhanced Bloom Filter (BF) for web catching

Bloom filters are space-efficient probabilistic data structures used to test whether an element is a member of a set.

Bloom filters find an extensive sort of usages, counting and tracking articles that one has read, speeding up Bitcoin clients, detecting malicious web sites, and improving the performance of caches.

An array of 256 bits is considered as the size of Bloom filter. Initially all positions in the bloom filter array are set to zero. Bloom filter array is divided into two parts first part contains the hashed value of MD5 of the domain name and the second part contains the hashed values of MD5 of path and file name.

The detailed steps are given below

1. Compute the MD5 of each URL (domain name and path with file name).
2. Convert the hexadecimal values of MD5 into binary, say x
3. Apply x mod size of bloom filter vector or use any hash function.
4. Find out 1's in the bit position of the output and make corresponding bit position in the bloom filter vector as 1.

Let n be the number of items in the bloom filter

p be the probability of false positives, fraction between 0 and 1 or a number indicating 1-in-p

m be the number of bits in the filter (or a size with KB, KiB, MB, Mb, GiB, etc)

k be number of hash functions

The equations used for a setting up of bloom filter array are given in equations 1 to 4.

n = ceil(m / (-k / log(1 - exp(log(p) / k))))……..(1)

p = pow(1 - exp(-k / (m / n)), k)…….(2)

m = ceil((n * log(p)) / log(1 / pow(2, log(2)))); …… (3)

k = round((m / n) * log(2)); ……. (4)

The incoming bit pattern (1's) of each URL is compared with the existing Bloom filter array. If the bit pattern which corresponds to 1's matches with the corresponding positions in the existing bloom filter array it indicates that the web page is already available in cache or prefetch area and no need to store the bits in Bloom filter array. Hence the page is retrieved from cache or prefetch area, otherwise Bloom filter array is to be modified by changing the corresponding bit positions as 1 when compared with the bit positions of the hash function applied to the MD5 of the URL. The request of URL is forwarded to the server to get the requested web page. As the size of Bloom Filter array increases the false positive rate decreases.

The flow chart for enhanced bloom filter technique used for web catching is given in figure 1.

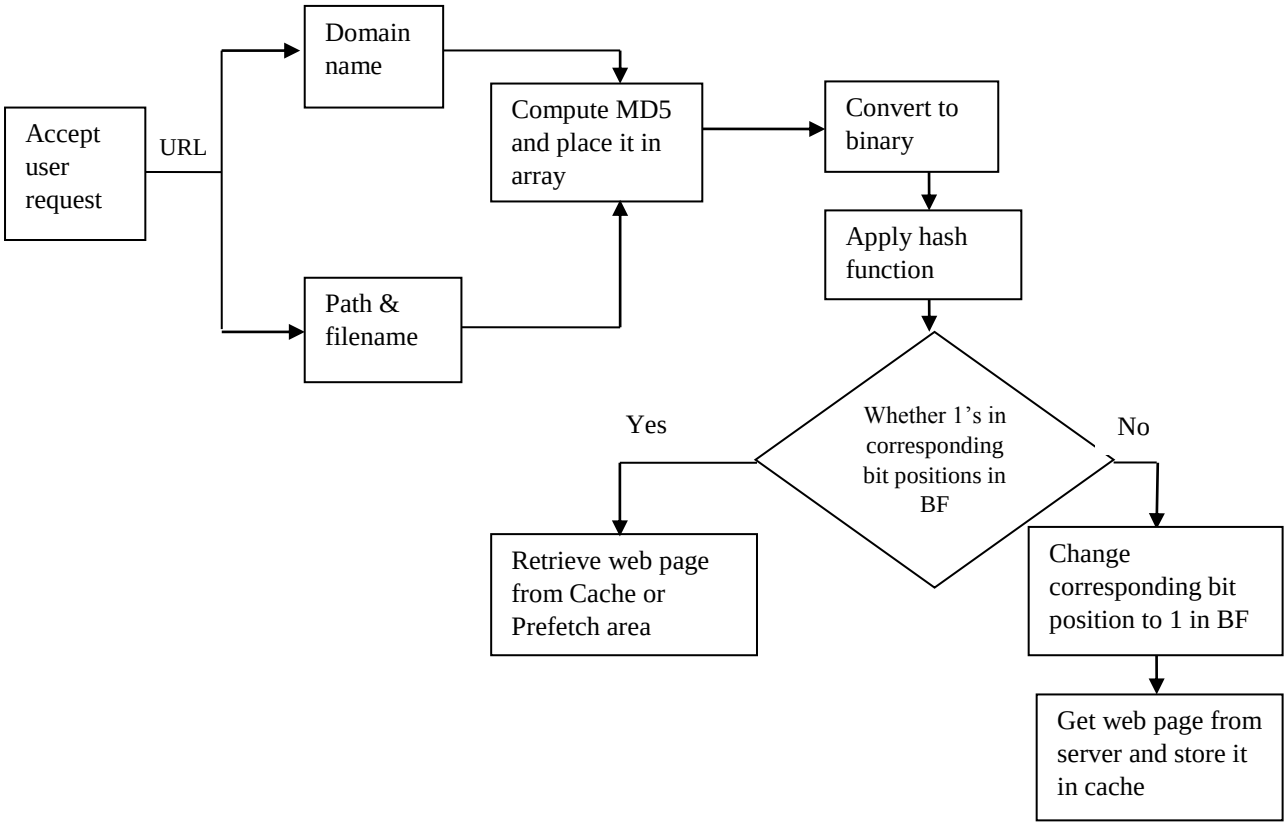

Figure 1. Flow chart for web caching using Enhanced bloom filter technique

Cache Hit Ratio

Web caching affords substantial advantages when the demanded pages are served from a near by server. The cache hit ratio is determined from the ratio of number of web pages provided from the cache to the total number of request for web pages. The efficiency of caching system is intensified by aspects such as the cache policy, the total number of cacheable objects, the size of the cache memory and the expiration period of the object.

A professional cache policy exploits the number of cache hits while reducing the number of cache misses, which lead to an extreme cache hit ratio, lower latency and better resource exploitation [10]. A cache hit ratio of 80% and prohibitive means that the majority of the requests are convinced by the cache [11]. Many cache servers have inherent tools that examine metrics such as cache hits, cache misses and cache hit ratio.

When a web page is retrieved from cache it is considered as cache hit and when it is retrieved from server it is treated as cache miss. The cache hit ratio can be calculated as given in equation 5.

- Find out the number of cache hits and misses over an offered period of time
- Divide the cache hits by total number of hits and misses
- Multiply that number by 100

Cache hit ratio = [Cache Hits / (Cache Hits + Cache Misses)] x 100 %.....(5)

By using MD5 and Bloom filter array, the requirement of memory for storing the cache directory is reduced and achieved high cache hit ratio. Cache consistency is also checked by using the If modified since header of Hyper Text Transfer Protocol (HTTP).

Prefetching Technique

Anticipated web pages are prefetched before they are actually required. The prefetched web pages are stored in the prefetch area. If the user access the web pages which have already prefetched, then it is provided to the user from prefetch area and its weight is increased by one. During prefetching, hyper links are prefetched depends on the maximum weight acquired by the user access pattern and the user preferences which are stored in database. The access pattern of each user is stored in prefetch database. The total number of hyper links available in a web page is determined. Depends on user

preferences and the weight acquired for each URL, the web pages are prefetched. The number of links to be prefetched is based on available bandwidth. The subsequent links of the webpage are prefetched only if the web pages are not available in cache or prefetch area and based on the current bandwidth usage.

The sequence of steps involved are given below

1. User can start browsing by entering URL.
2. If the requested page is already available in cache or prefetch area then it is directly fetched from there and its weight is incremented by one.
3. If the requested page is not in cache, fetch the web page from server and store it in to cache and set weight to 1, weight represents the number of times the page accessed.
4. When cache is full least recently accessed page with least weight is replaced with the newly accessed one, thus creating space in cache. If more web pages having same weights then FIFO is used for replacement.
5. The total number of links and sub links in an URL is found out. The links to be prefetched is based on user's access pattern and available bandwidth.
6. The prefetch module extracts the hyperlinks and stores in prefetch area with its weight set to 1. If the prefetched web pages are actually fetched by users, increment its weight.
7. The objects that have been previously accessed frequently may have more weight even if they are not accessed any longer. To reduce the weight of them, a timer is maintained. When timer expires weight of all links are reduced by 1.
8. Prefetched pages are also checked for consistency using If modified since header.
9. The web pages of sub links are also stored in prefetch area using the web prefetching technique. Domain name, links and sub links are stored in hierarchal structure as bloom filters to identify the user access pattern.

Cache Replacement Policy

Least Recently Used (LRU): This cache replacement algorithm keeps recently used items at the top of the cache. Whenever a new URL is accessed, the LRU places it at the top of the cache. When the cache reaches the limit, items that have been accessed less recently will be eliminated starting from the end of the cache. This can be an affluent algorithm to use as it needs to keep "age bits" that show precisely when the URL is accessed. In addition, when an LRU cache algorithm deletes an URL the "age bit" changes on all the other items [12]. LRU is a most commonly used technique.

First in first out (FIFO): Using this procedure, the cache performs in the similar way as a FIFO queue. The cache throws out the first block accessed first without concerning how frequently or how many times it was retrieved before [13].

LRU integrated with FIFO replacement algorithm

When the cache is full the web page with the least weight is deleted here. If more than one entry has the same weight the one which comes first (FIFO) is deleted. Hence both LRU and FIFO are integrated together to replace web page.

PERFORMANCE EVALUATION

Bloom filter array of size 256 is used for cache directory. 128 bits are used as the bloom filter for domain name and remaining 128 bits are used as the bloom filter for path and file name.

The user requests are forwarded to the web server only if the corresponding web pages are not available in cache or prefetch area. If the documents are fetched from web server; a copy of it is stored in cache. Subsequent request for the same URL yields cache hit. The hyper links and sub links are fetched as described in Section 2.3.

The performance of proposed enhanced bloom filter technique is analyzed in terms of cache hit ratio and latency.

i. Hit ratio

Hit ratio is calculated using the following equation

CHR = [CH / (CH + CM)] x 100 %

where,

CHR- Cache Hit Ratio

CH - Number of cache hit

CM - Number of cache miss

The graph that shows time vs cache hit ratio for the proposed technique is given in figure 2.

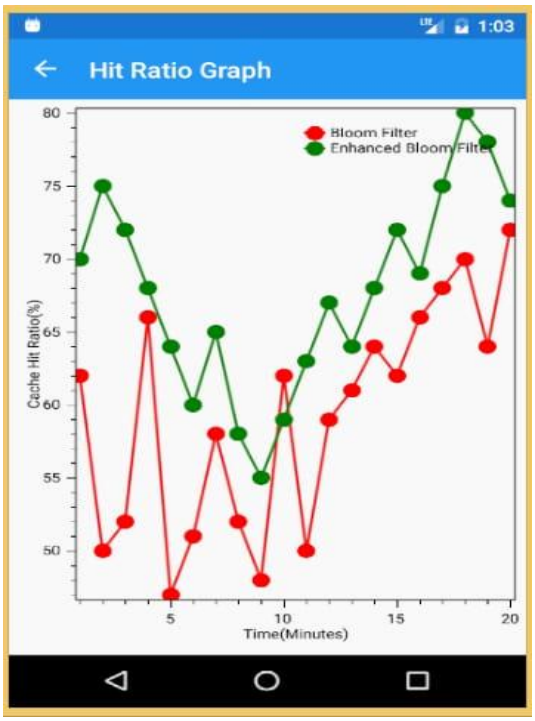

Figure 2: Time vs Cache hit ratio

ii. Latency

Network latency is a term of how much time it takes for a packet of data to get from one chosen point to another. Response time is reduced very much.

The graph used for the analysis of latency is given in figure 3.

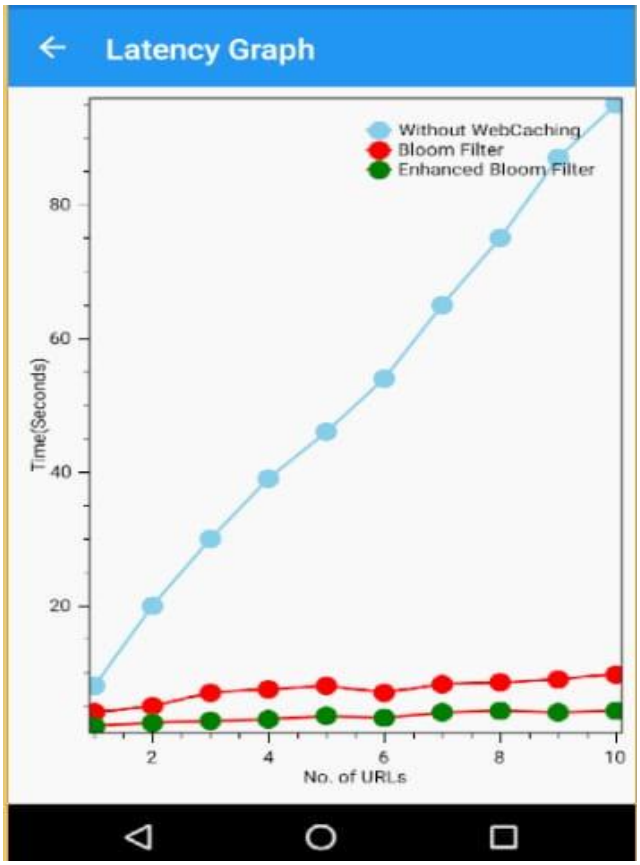

Figure 3: Number of URLs vs Time

CONCLUSION

Web caching and prefetching are the two techniques to reduce the user perceived latency. Web cache is used to store the most popular web objects recently accessed by the users close to the client side inorder to avoid the subsequent request be forwarded to servers. Web prefetching is fetching of anticipated web pages in advance by client before the actual request is sent by client. The browser application implemented in mobile phone using enhanced bloom filter technique for caching and prefetching is very useful to reduce web traffic, bandwidth consumption and user perceived latency. An appropriate decision for replacement is taken when the cache is full using the integrated LRU and FIFO algorithm. The bandwidth monitoring agent regulates the web traffic by controlling the number of hyperlinks to be prefetched. The experimental results have been conducted which shows that the integrated catching and prefetching reduces much more latency and increases the cache hit ratio from 50 to 80%.

REFERENCES

[1] Stratisloannidis, Laurent Massoulie and Augustin, 2010, "Distributed Caching Over Heterogenous Mobile Networks", SIGMETRICS ACM 978-1-60558-005-0/08/06.

[2] Paul, S; Z Fei., 2001, "Distributed Caching with Centralized Control", Computer Communications 24(2):256–268.

[3] MazenZari, HosseinSaiedian, Muhammad Naeem, 2001, "Understanding and Reducing Web Delays", Computing Practices IEEE pp. 30-38.

[4] Yixue, Marcelo Schmitt Laser, Yingjun Lyu, Nenad Medvidovic, 2018, "Leveraging Program Analysis to Reduce User Perceived Latency in Mobile Applications", ICSE.

[5] John L. Hennessy; David A. Patterson, 2012, "Computer Architecture: A Quantitative Approach", Elsevier. pp. B–12. ISBN 978-0-12-383872-8. Retrieved 25.

[6] Dr. Jayasudha J.S , Greeshma G. Vijayan, 2012, "A Survey on Web Pre-Fetching and Web Caching Techniques in Mobile Environment", Proceeding of the First International Conference on Information Technology Convergence and Services, Vol. 1 pp. 119-135.

[7] BeihongJin, Sihua Tian, Chen Lin, Xin Ren and Yu Huang, 2007, "An Integrated Prefetching and Caching Scheme for Mobile Web Caching System", Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing.

[8] Monti BabuLal Pal, Dinesh C. Jain, 2014, "An Approach for Web Prefetching to Enhance User Interaction of Web Application Using Markov Model", Fourth International Conference on Communications Systems and Network Technologies, pp. 373- 377.

[9] Dr. M. Thangaraj, Mrs. V. T. Meenatchi, 2014, "Domain Based Prefetching in Web Usage Mining", International Journal of Advanced Computer Science and Applications, Vol. 5, No. 6, pp. 53-60

[10] Q. Ren and M. Dhunham., 2000, "Using Semantic Caching to Manage Location Dependent Data in Mobile Computing", Proceedings of ACM/IEEE MobiCom, 99:210–221.

[11] B. Zheng, J. Xu, and D. L. Lee, 2002, "Cache Invalidation and Replacement Strategies for Location-Dependent Data in Mobile Environments", IEEE Trans.on Comp, 51(10):14–21.

[12] Preetha Theresa Joyi and K. Poulose Jacob, "Cache Replacement Policies for Cooperative Caching in Mobile Ad Hoc Networks", Dept. of Computer Science, Cochin University of Science and Technology.

[13] K. Lai, Z. Tari, and P. Bertok., 2004, "Mobility Aware Cache Replacement for Location Dependent Information Services", Technical Report T R- 04-14, (R.M.I.T. School of C.S & I.T).