Forecasting Load Balancing Method by Prediction Hot Spots in the Shared Web Caching System

Sung C. Jung* and Kil T. Chong**

*Control and Instrumentation Engineering, Chonbuk National University, Chonju 561-756, Korea
(Tel: +82-63-270-2478; Fax: +82-63-270-2451; Email:sungchil_jung@hotmail.com)
**Faculty of Electronics and Information Engineering, Chonbuk National University, Chonju 561-756, Korea
(Tel: +82-63-270-2478; Fax: +82-63-270-2451; Email:kitchong@moak.chonbuk.ac.kr)

Abstract: One of the important performance metrics of the World Wide Web is how fast and precise a request from users will be serviced successfully. Shared Web Caching (SWC) is one of the techniques to improve the performance of the network system. In Shared Web Caching Systems, the key issue is on deciding when and where an item is cached, and also how to transfer the correct and reliable information to the users quickly. Such SWC distributes the items to the proxies which have sufficient capacity such as the processing time and the cache sizes. In this study, the Hot Spot Prediction Algorithm (HSPA) has been suggested to improve the consistent hashing algorithm in the point of the load balancing, hit rate with a shorter response time. This method predicts the popular hot spots using a prediction model. The hot spots have been patched to the proper proxies according to the load-balancing algorithm. Also a simulator is developed to utilize the suggested algorithm using PERL language. The computer simulation result proves the performance of the suggested algorithm. The suggested algorithm is tested using the consistent hashing in the point of the load balancing and the hit rate.

Keywords: cache, consistent hashing, HSPA, Hot spot

1. Introduction

As the World Wide Web becomes an important medium in providing information to the users, it is necessary to have better methods and techniques to transfer information effectively and reliably. However, the current content delivery network system may have failed to deliver prompt service because network states are prone to unpredictable delays and frequent failures. The swamped servers or the bottleneck in the network usually cause these delays and failures. Swamped servers, facing more simultaneous requests than their resources can support, will either refuse to serve certain requests or will serve them very slow. Any proxy is limited in the number of users it can serve, and becomes a bottleneck during periods of intense use. These problems can be effectively addressed using Web caching, however network and server infrastructure expansions have not kept pace with the tremendous growth in Internet use. Therefore, it will be impossible to meet users' requests for quick service.

Recently, in order to satisfy a user's request and utilize the network's resources effectively, it makes a requested quantity to do a load balancing by distributing a proxy to several networks in the Shared Web Caching System (SWCS)[4]. SWCS makes the response time quicker by connecting a user to a proxy that is nearest to the network or to a proxy whose RTT (Round Trip Time) [6] is less. And when utilized for its hashing function [2, 3], it allows for the efficiency of proxies to be maximized by preventing several proxies from duplicating the items. The conventional SWC distributes the items based on the current status of the proxies, which could be, referred to as the static load balancing method. The suggested method HSPA distributes the hot spot using the information predicted for the proxies, and this could be called dynamic load balancing. The information of the proxies is obtained by the dnshelper. The dnshelper monitors every proxy in the system calculating the load applied to the proxies. The HSPA and the consistent hashing [2] use the information of the hot spots for the load balancing.

The conventional SWCS patches the item that has been requested and also the items that might be needed in the future based on the current state of the requests. If the some of the popular hot spot predicted patches to the proper proxies in advance, it will improve the load balancing and the hit rate of the SWCS.

In this study, the Hot Spot Prediction Algorithm (HSPA) has been suggested to improve the conventional SWCS. This method predicts the popular hot spots using a prediction model. And the predicted hot spots will be patched to the proper proxies according to the load balancing algorithm. The suggested algorithm has advantages in the load balancing, hit rates and the shorter response time over the existing SWC systems. The data file used for this study is the access_log file of NASA home page. The suggested algorithm tested used a simulator developed using PERL language. The computer simulation result proves the performance of the suggested algorithm. The suggested algorithm is tested using the consistent hashing in the point of the load balancing and the hit rate

2. Web Caching and Consistent Hashing

Web caching is usually categorized as a single mode and shared web caching. The single mode is a simple Web caching method that provides service by connecting multiple users to a single proxy. If the proxy does not have the information requested by a user, it will request the item from the original content provider (CP) server. It stores the received information for future requests from other users. But the single mode has limitations in proxy load, network bandwidth, hit ratio, latency, etc. since it processes all the requests in a single proxy.

But the SWC [4], that is a processing method to distribute a request to several proxies. In the SWC, if the item is in the primary server the request is cleared up at the primary server, however, if the primary server does not have the item stored, the primary server will broadcast to get the item from another server taking advantage of the ICP (Internet Cache Protocol)[6] or the hash function [2, 3]. If none of the proxies have the requested data, it will do an action like as the single mode. Since it responds to requests in this manner, it has better performance in proxy load, hit ratio, latency, network bandwidth, etc. than the single mode.

One of the main issues in the SWCS is the load balancing problem that makes requested quantities to distribute in each of the proxies properly. In this study, by using the HSPA and the consistent hashing [2] the load balancing will be demonstrated. The load balancing is highly related to the hit rate that is the most important factor in the content delivery networks. The hit rate of the suggested algorithm is also improved by the suggested HSPA proven by the computer simulation.

A hashing function is used in the consistent hashing. First of all, the standard hash function is discussed before reviewing the operation principles of SWC using a consistent hashing. For example, it is assumed there are 7 proxies in the networks. Then the items undergo a course of **Proxy ID = H(item) Mod 7** when it maps an item requested by a user to the proxy, where Mod calculates the modulation of 7 and a function H is the hashing function. If a proxy will be added up or deleted from the networks, it will go through whole mapping again. Consequently, the hit rate decreases and the performance of proxy depreciates rapidly since it always makes a new re-mapping if a proxy is added or deleted.

However, the consistent hashing makes a re-mapping for part of the items not for all the items when a proxy is added or deleted. This algorithm makes a mapping for hash values of all the items to the unit circle between [0, 1]. A mapping to the unit circle becomes random using the hash function. With the same way, each of the proxies in the SWCS makes a mapping to the unit circle using the hash function as well. Method to search for a proxy in which each of the items are allocated, that is to move clockwise from the item hash value on the corresponding unit circle and to meet with the proxy hash value, and the proxy takes charge of the item.

3. Hot Spot Prediction Algorithm (HSPA)

The method suggested in this study is similar to consistent hashing with the round robin DNS[1] except the prediction of the hot spot cached in the proxies. A prediction of hot spot is accomplished by analyzing the items requested to a particular proxy and several items that have been requested at the most. The several hot items that have been indicated by the hot spot finder are patched in prior to the proxies which are possible to distribute by using the existing requested quantities and the time series or another method on the based of predicted request quantities. They have been patched to a proxy, which is near to a user using this method, it will be possible to increase the performance of the content delivery network by preparing a huge amount of requests suddenly. Using an access_log in the NASA home page directly in this study, not a prediction model has represented the prediction effect. Data generation and system constitution according to this suggested method are treated with in the following chapters in detail.

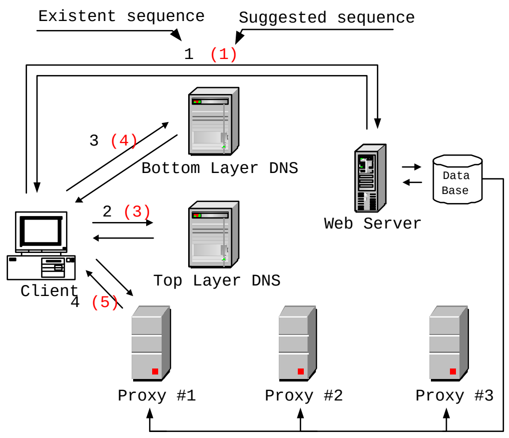

Fig. 1. System constitution of HSPA

The HSPA system constitutes as shown in Fig. 1. This system has analogous structure as in the global hosting system [10]. When a user requests to a CP server, the upper DNS interprets the domain and then transfers it to a lower DNS address which is nearest to a user geographically. And then the lower DNS transfers it to a nearest host IP address. Therefore, the lower DNS makes a virtual name to map to a real IP. At that time a user requests a desired item using an address of proxy provided with the lower DNS. If the proxy has item, he will bring the item, if not, the proxy will request an item to the original CP server to provide a user with it and save it in its memory, for future use.

As shown in Fig. 1 progress of 5 has been conducted in prior by analyzing NASA access_log and predicting hot spot. In the consideration of the loads given to each of the proxies, they are patched to the proxy of low load.

4. Experimental Data

The access_log file in the NASA site has been used for producing the prediction data. The access_log analysis method necessary for the process to predict a hot spot, characteristics of the data that are explained as follows:

4.1. Analysis of access_log file

An access_log is a file in which all the jobs conducted through a web server have been recorded. When a user connects to a web server (or proxy), all the jobs will have been stored in an access_log as data. Explaining in detail, a demand from a user to look at a particular web page makes many items related to the corresponding web page accessible to the web server. Therefore, the information on all stages to process a demand from a user including not only a particular web page demanded by a user but also embedded (image data, link data, etc) files, and the like related to the corresponding web page are saved in a access_log. Through an analysis of the access_log, we can know the information on the number of demanded data, access time, number of visitors, visiting route, and so on. So the access_log becomes important data to comprehend the purposes that a user has when visiting a site.

Also an access_log contains various information on the jobs conducted by the web server, not only a demand and success or failure of a particular job, but also information on the resolution method when it fails. A NASA access_log used in this study records the following contents.

$in24.inetnebr.com - -[01/Aug/1995 : 00 : 00 : 01 - 0400]"GET/shuttle/missions/sts - 68/news/sts - 68 - mcc - 05.txtHTTP/1.0"2001839$

The above recorded contents are composed of 7 kinds of fields. The fields consist of Host, Identification, User Authentification, Time Stamp, HTTP Request Field, Status Code, and Transfer Volume in sequence.This study is used to the Time Stamp and HTTP Request Field among the 7 kinds of fields.

4.2. Characteristics of the Experimental Data

Analysis of the NASA access_log, indicated that the data pattern follows Zipf's law [5]. But the percentage of each popular item is not large enough to apply the prediction. Therefore, the experimental data file is generated using the Surge [7] program, a load generator developed in Boston University. The experimental data used in this study have been regenerated with 1,000 items using the Surge,and the total size is 18MB.

The round robin DNS method executes a load balancing in several proxies sequentially by reading in a network status from the DNS zonefile. It is assumed that to update the zonefile takes Δ period of time. The consistent hashing will not know the hot spot change by looking at its current data if the hot spot changes while the zonefile is updated. The consistent hashing will recognize the change of the hot spot after the zonefile updated. The HSPA has advantages over the consistent hashing during the time period of Δt. HSPA aims to minimize the performance degeneration caused by a change of hot spot by predicting the hot spot that will change during the time period of Δt and distributing them in each of the proxies.

Fig. 2 has been prepared by adding up for the hot spots of rank No.1 and 2 as a function of time. When a load balancing is carried out in a real system for all the hot spots through round robin, if the requested quantities in each of the proxies are 40, 50, and 50 respectively, it shall be equal to the total sum. Therefore, a load balancing can be carried out in each of the proxies by using the data and predicting the hot spots.

5. Simulation

The simulator has been constituted to conduct a simulation by the existing round robin and the round robin using hot spot prediction method suggested in this study as well.

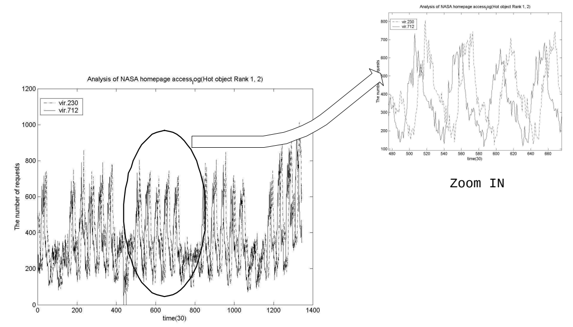

Fig. 2. Trend of total requested quantities as a function of time

5.1. Simulation Method

Varying the number of proxies affects the memory size and processing time of the server to restrict the performance of the CP server allowed for simulations. The cache policy of memory has followed the type of Least Recently Used (LRU) to restrict a storage space of the proxy and the type of service is First In First Out (FIFO) [9]. A processing time of the proxy is controlled by an option called bound, which is a time to process one item. Parameter bound was used in variable that decide success and miss of service. Also the processing state of item is decided according to a time delay. Character A is defined as the time interval required in each of the items, such as A is equal to 30 min. divided by the number of items in 30 minutes, if the simulator has been processing with data of 30 minutes.

Therefore, it passed a time of A upon a request of item. It did not make a miss if a request has been processed within a time of bound less than A. But it made a miss if a request has not been processed within a time of bound greater than A. When a requested data has been stored in a queue, the process is called a hit, unless it is called miss. Therefore, number of **Hits / total number of items * 100** becomes hit rate in this simulation.

The processing procedures of simulation are as follows: Initially, various performance options are set up in Test.pl program, and collect load states of each proxy in the SWCS to make a prediction. And then the items produced by Surge are inputted according to the date and hours. The input item is hashed using MD5 [9], and then stored into the selected proxy server hashed according to the designed algorithm. The loads of proxy servers are measured in the same time. In the experiment, 1,000 virtual caches have been generated to apply consistent hashing.

A subroutine called 'run' in the module Virtual.pm takes charge of a practical simulation, in which all the data are processed. At first, items have hashed using the MD5 in the module Dns.pm, and then have mapped in 1,000 virtual caches where each of the proxy servers took charge. And then the hashed items allocated to the proper proxies. At this stage, the current position of hot spot should be determined among the proxies in order to process the hot item by round robin, since it could be allocated in a next proxy when a next hot spot has entered in. When the input is not a hot spot,

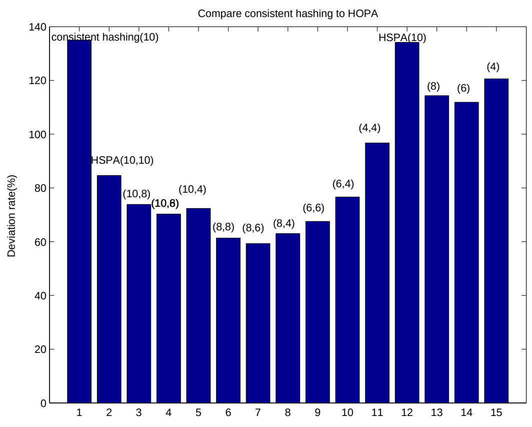

Fig. 3. Load deviation in proxy servers

it hashed by MD5 and mapped by the hashed virtual cache.

5.2. Performance Evaluation

The load balancing and hit rates have been investigated for the consistent hashing algorithm and the HSPA predicting the hot spots based on round robin DNS method through this experiment. Fig. 3 shows the standard deviations of load distributions in each of the proxies of the consistent hashing and HSPA, in which the numbers of hot spots to be distributed have been considered as well.

As shown in the figure, a better load distribution is obtained when it has been participated up to the priority rank No. 2 than when it did the priority rank No. 1 only. When two of the hottest items have been participated in the load distribution such that the hottest item is distributed to 8 proxies and the second hottest items are distributed into six proxies, it gives the best performance. Therefore, when several hot spots are distributed, it is the key point to what number of proxies has been distributed. It shows us the principle that a load distribution should be done dynamically.

5.3. One Hot Spot Distribution

5.3.1 Performance in the Equal Processing Time

Fig. 4 and 5 show the comparison of hit rates at different speeds of the consistent hashing and the HSPA. Ten proxy servers are used in this experiment. The number of proxies 10 up to 4 did an experiment for the methods by reducing with 2. Among those, the prediction methods HSPA whose hit rates are the best when the number of proxies was 4 and the queue size was 10. And as a queue size increased, it had the highest hit rate when the number of distribution proxies was 6.

The second experiment was done when the processing time was slow. In this case, the highest hit rate arises when the queue size was 10, which was the minimum, and the number of distribution proxies was 6. And as a queue size increased, it had the highest hit rate when the number of distribution proxies was 6.

5.3.2 Performance in the Equal Queue Size

In Fig. 6, it shows us a graph how the hit rate varies as the processing speed slows under a condition that the queue size was equaled to 30 constantly. As shown in this Fig.6, the prediction method has better hit rates than the method of

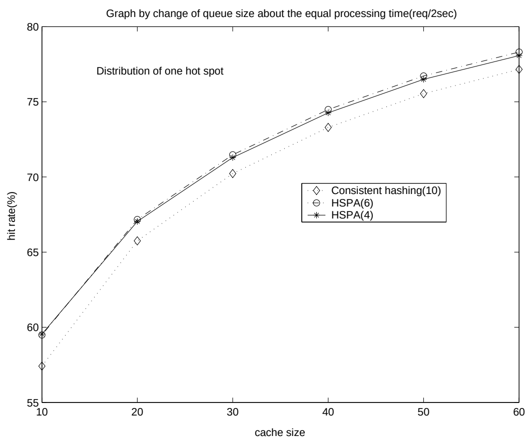

Fig. 4. Hit rete when speed is req/2sec

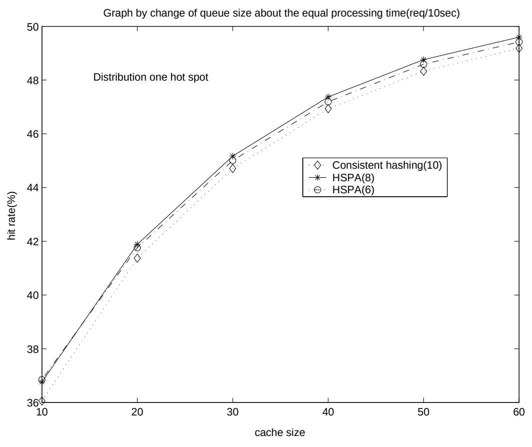

Fig. 5. Hit rate when speed is req/10sec

consistent hashing algorithm. It had better hit rates when the items were distributed into six proxy servers whose processing time was fast and when the items were distributed into 8 proxy servers for the part whose processing time was slow.

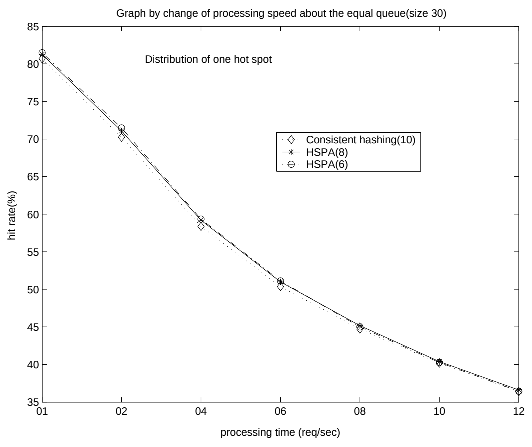

Fig. 6. Hit rates when queue size is equal(30)

As shown above, a better hit rate for a request from a user could be obtained to lower the number of proxies to be distributed by considering the total job quantities in the area

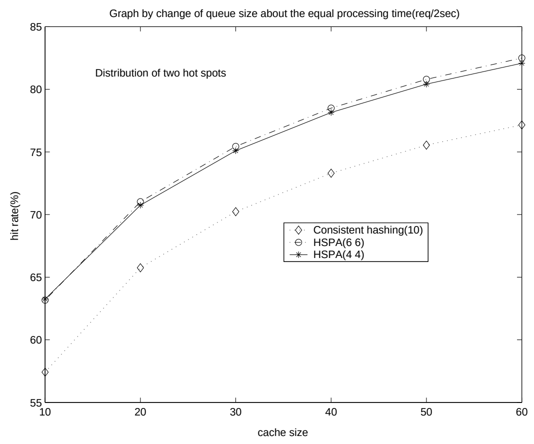

Fig. 7. Hit rete when speed is req/2sec

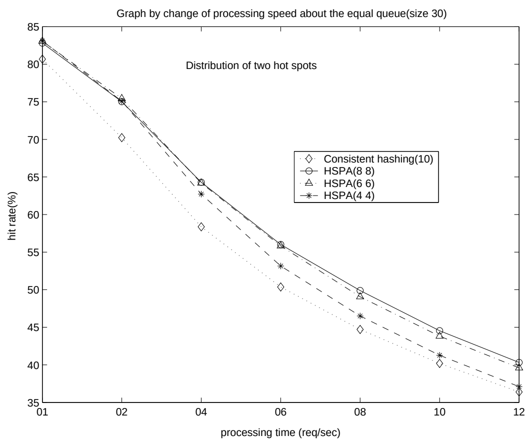

Fig. 9. Hit rates when queue size is equal(30)

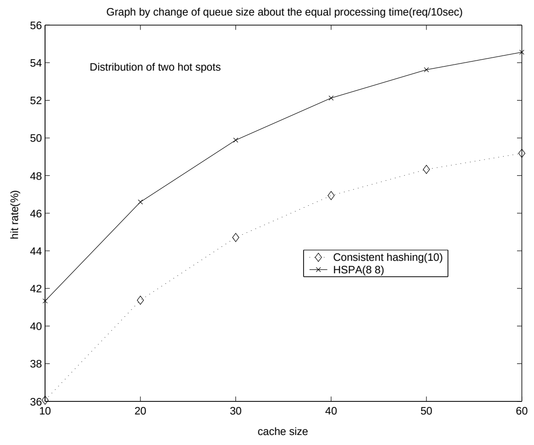

Fig. 8. Hit rate when speed is req/10sec

that the processing speed is fast, and as the processing speed is decreased as more number of proxies are distributed.

5.4. Two Hot Spots Distribution

5.4.1 Performance in the Equal Processing Time

In section 5.3.1, only one hot spot of the most popular was distributed, however, in this section two hot spots were participated in the load distribution that shown in Fig.7 and 8. The better of hit rates when two hot spots were participated than when one hot spot did. The prediction method HSPA has an averaged hit rate of more than 5% compared with the method of consistent hashing.

For a slow processing time, the best hit rate has been obtained when both first hot spots and second hot spots were distributed to 8 proxies. The prediction method has an averaged hit rate of more than 4% compared with the method of consistent hashing .

The legend of the figure Prediction(a, b) indicates the first number a in the parenthesis means the number of proxies for the rank No.1 hot spot distributed and the second element b means the number of proxies that the rank No.2 hot spot will be distributed. The results show the better hit rates when the most hot spots are distributed into four proxies and the second hot items are also distributed into four proxies when the proxies processing time was fast. As the processing time gets slower the pair of proxies for the hottest items and the second hot spots are distributed into (6, 6) and (8, 8) gives good hit rates.

5.4.2 Performance in the Equal Queue Size

In Fig. 9, same method with that of in Fig. 6, it shows us a graph how the hit rate varies as the processing time slows under a condition that the queue size was equaled to 30 constantly.

It is important for a number of proxies to distribute as the processing time becomes slower. However, it could get a worse result even when a larger number of proxies have been distributed. It was lower 0.6% when (10, 10), a larger number, has been distributed than when (8, 8).

The total workload should be considered to determine a number of distribution proxies in the SWCS. A small number of proxies should be distributed when a small workload and a fast processing time but a many number of proxies should be distributed when large workload and a slow processing time. All the simulation showed the HSPA gives better hit rates than the consistent hashing.

6. Conclusion

In this study, the Hot Spot Prediction Algorithm (HSPA) has been suggested to improve the performance of the consistent hashing algorithm. This method predicts the popular hot spots using a prediction model. Also, the predicted hot spots will be patched to the proper proxies according to the load-balancing algorithm.

The modified access_log file of NASA home page has been used as the experimental data. The data obtained from the NASA access_log file cannot be directly applied to this study, therefore, the new experimental data has been produced using the Surge program based on the access_log file of the NASA home page that follows the Zipf's law. The suggested algorithm was tested using a simulator that had been developed using the PERL language. The results of the computer simulation prove the performance of the suggested algorithm. The suggested algorithm is compared to the consistent hashing in the view of the load balancing and the hit rate for the various combinations of hot spots and the proxies used. The

computer simulation results shows that the suggested HSPA is superior to the consistent hashing algorithm.
In conclusion, the suggested HSPA introduce a new prediction method into the network system area, improve the hit rate of the request in the Internet and also it improves the load balancing of the shared web caching system.

References

[1] Han-Chieh Chao and Tin Yu Wn *et al.*: The network topology based domain name service. Parallel Processing, Proceedings. 1999 International Workshops, (1999) 528 – 533

[2] David Karger *et al.*: Web Caching with Consistent Hashing. In Proceedings of the 8th International World Wide Web Conference, (1999)

[3] V. Valloppillil and K. Ross.: Cache array routing protocol v1.1. Internet Draft, http://www.globecom.net/ietf/draft/draftvinod-carpv103.html, February (1998)

[4] Ross, K.W.: Hash routing for collections of shared Web caches.IEEE Network, Vol.11. Issue: 6 , Nov/Dec (1997) 37 – 44

[5] Lee Breslau *et al.*: Web Caching and Zipflike Distributions : Evidence and Implications. IEEE INFOCOM, (1999) 108–121

[6] D. Wessels and K. Claffy.: Application of Internet Caching Protocol(ICP), version 2. Internet Draft:draft-wesselsicpv2appl00. Work in Progress., Internet Engineering Task Force, May (1997)

[7] P. Barford and M.E.Crovella.: Generating representative web workloads for network and server performance evaluation. In Proceedings of ACM SIGMETRICS Conference, (1998) 151 – 160

[8] Woei-Luen Shyu and Cheng-Shong Wu *et al.*: Efficiency analyses on routing cache replacement algorithms. Communications, ICC 2002. IEEE International Conference, Vol. 4. (2002) 2232 – 2236

[9] R. Rivest.: The MD5 Message-Digest Algorithm. RFC 1321, Network Working Group, April (1992)

[10] Leighton *et al.*: Global hosting system. united states patent & trademark office, auguest 22 (2000)