ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 5, May2017

An Efficient Technique for Runtime Distributed Web Caching with Knowledge based Clustering

Ankita Sharma

M.Tech(S.E), Department of Computer Science & Engineering, Shri Ram Murti Smarak College of Engineering & Technology, Bareilly, India

ABSRTACT: The Distributed Web Caching System experiences scalability and less robustness issue because of over-burden and congested Proxy servers. Load Balancing and Clustering of proxy servers helps in quick recovery of pages, however can't guarantee robustness of system. We have given answer for scalability and robustness of Distributed Web caching system and for load balancing clustering and metadata manageability. We have likewise refined our method utilizing Proxy server groups with Knowledge based clustering and dynamic assignment of requests. We audit a algorithm for Distributed web caching ideas with Knowledge based cluster of Proxy server in view of land locales. It increases the Scalability by keeping up metadata of neighbors. We are making clusters based on knowledge proxy serves having similar data are collectively make a cluster. Based on which hit ration will be high. Based on which hit ratio will be high. It builds the scalability by keeping up metadata of neighbors altogether and equalizations heap of intermediary servers during Runtime to other less congested proxy servers, so system doesn't get down unless all Proxy servers are completely loaded so higher robustness of system is accomplished.

KEYWORDS: Distributed Knowledge based Clustering,Origin server, Proxy Server, Latency, Hit Ratio, Metadata, Robustness.

I. INTRODUCTION

The introduction of World Wide Web has changed the perspective of the client group to the system in regard to availability and ease of use. With the Web the client can search for and recover all sort of data from the system without having any knowledge of the network**.** Now a day's most popular web sites are suffering from server congestion, since they are getting thousands of requests every second. The heterogeneity and complexity of services and applications provided by web server systems is continuously increasing. Geologically distributed web servers [1] and intermediary server aim to decrease user latency time through network access redistribution and reduction of amount of data transferred, respectively. In this we consider diverse web systems, specifically web clusters that use a tightly coupled distributed architecture. cluster technology is a cost effective arrangement since it is less expensive than a solitary faster machine. From the client's perspective, any demand to a Web cluster is exhibited to an intelligent server that goes about as a delegate for the website. Group designs with Web switch dispatcher have been embraced with various arrangements in various academic and commercial Web clusters.

A Web cluster is subject to very extraordinary workload. The hit rate of a Web cache can be expanded altogether by sharing the interests of a bigger group [3]; the more individuals are getting to a same cache, the higher the likelihood that a given data is available in the reserve. To expand the powerful customer population using a cache, several caches can cooperate. Two common approaches to deal with execute an extensive scale reserve collaboration plan are hierarchical and distributed caching.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 5, May2017

II. CHALLENGES IN DISTRIBUTED WEB CACHING SYSTEM

In distributed web caching system, documents can be cached at the clients, the proxies, and the servers. A client always requests page from its local proxy if it doesn't have a valid copy of such page in its own browser's cache. Upon receiving a request from client, the proxy first checks to see if it has the requested page. If so, it returns the page to the client. If it doesn't have the requested page in its cache, it sends a request to its cooperative proxies or the server. Upon receiving a request from another proxy, a proxy checks if it has the requested page. If so, it returns the page to the requesting proxy. If not, the proxy may further forward the request to other proxies or the server.
If none of the cooperative proxies has such page, the requested page is fetched from the server. In order to make a WWW caching system work, the following problems need to be solved properly:

1. Proxy placement: Where to place a cache proxy in order to achieve optimal performance?
2.Caching contents: What can be cached in the caching system, data, connection, or computation?
3.Proxy cooperation: How do proxies cooperate with each other?
4. Data sharing: What kind of data/information can be shared among cooperated proxies?
5.Cache resolution/routing: Cache resolution/routing solves how does a proxy decide where to fetch a page requested by a client?
6.Pre-fetching: How does a proxy decide what and when to pre fetch from Web server or other proxies to reduce access latency in the future?

III.RELATED WORK

Numerous paper discussed about web caching [1] and distributed web caching [4][5].Web caching in refrence paper focuses on static content caching .We here focus on runtime distributed web caching where problem of extra overhead and cache coherence is reduced and focusing on increasing the hit ration and reducing latency time.
By making cluster on geographical region based the extra overhead problem is reduced but not upto that extent as after introducing clustering on basis of knowledge .

IV.PROPOSED WORK

In this section we give the solution for extra overhead problem,problem of unmanageable data , cache coherence ,less robustness,scalability and problem of data retrieval from proxy server. We get a high benefits by clustering but in terms of hit ration by adding the concept of knowledge based clustering where cluster are formed based on similar knowledge example like in same city requirement is for computer defined data then all the cluster fall in the region which will rich In data containing computer.

Proposed Algorithm:-
These algorithms are for Runtime Distributed Web Caching System using the concept of Knowledge based Clustering. This section of Algorithms basically contains three algorithms For: Client, Proxy Server and Origin Server.

1. **Proxy Servers:**

/* queuelength: It is associated with every proxy server, which tells how many client's requests can be made to a proxy server.*/

- PS: Proxy Servers.
- Noofservices: tells how many connections are active with proxy server.
- CIP: Client's Internet Address.
- Reply: has either requested page or message "NO".
- Rpage: Requested page or file.
- Ps_ip []: is a stack of Internet Addresses of all the proxy servers in same Cluster.
- OS []: is a stack of Internet Addresses of all the Origin Servers.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 5, May2017

- Cluster []: is a stack of Internet Addresses of all Clusters.
- MB: Match Bit
- KBC: Match Bit for Knowledge based clustering
- KBC[]: is a stack of Internet Addresses of all the proxy servers in same Knowledge based
- Clustering.
- KBC_CSE[]: is an array of computer science Engineering
- KBC_ECE[]: is an array of Electronic & communication Engineering
- KBC_ME[]: is an array of Mechanical Engineering
- KBC_CVE[]: is an array of Civil Engineering

Step 1: queuelength(ql)=0;

Step 2: If Request from OS [] for connection then

2.1 Establish connection with the origin server.

2.2 Connection Established.

(1) A new thread is established.

(2) Receive data from origin server.

(3) Update its metadata and broadcast information to all clusters.

Step 3: Proxy Server will wait for connection with clients or from other proxy servers.

Step 4: If request is to update data by any other proxy server then update metadata.

Step 5: if (q11+q12+q13) <=240

Step6: if (q1<80)

Step7: A Connection is established by creating a new thread to deal with it & client's locative address in CIP.

Step8: Get client's locative on in IP address in CIP.

Step9: if Incoming request from Client CIP /*request is from client*/

(a) If Pattern match is on same KBC [] /*KBC_CSE[], KBC_ECE[], KBC_ME[] , KBC_CVE[] */

9.1 wait();

9.2 ql=ql+1;

9.3 Signal();

9.4 Search metadata();

9.5 If matchfound then

(1) If matchfound is on current cluster

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 5, May2017

1. If (same proxy server)
i. wait();
ii. ql=ql+1;
iii. signal();
iv. Search for Rpage in cache.
v. Return Rpage to CIP.
vi. ql=ql-1;
(2) else/*else of 1.*/
1. Send request to another proxy server.
2. looking for reply.
3. Return Rpage to CIP.
9.6 else /*else of (1) */
(1) if(MB==1)/* Data on Neighbour cluster*/
1. wait();
2. ql=ql+1;
3. signal();
4. Return Rpage to CIP.
5. ql=ql-1;
(2) else If (MB==0)/*else of (1) */ /*Increment cluster to search*/
1. (Owncluster+2)% n
2. If (r page=NO)

- No Such page Exist in This Cluster & Goto step A

3. else /* Else of (2).2*/
i. Wait();
ii. Ql=ql+1;
iii. Signal();
iv. Return Rpage to CIP.
v. Ql=ql-1;
else /*else of (a) */
(1) if(KBC==1)/* Data on Neighbor knowledge based cluster*/
1. wait();
2. ql=ql+1;
3. signal();
4. Repeat step 9
5. Return Rpage to CIP.
6. ql=ql-1;
(2) else If (KBC==0)/*else of (1) */ /*Increment knowledge based cluster to search*/
1. (Ownknowledgecluster+2)% n
2. If (r page=NO)

- No Such page Exist in This Cluster & Goto step A

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 5, May2017

3. else /* Else of (2).2*/
i. Wait();
ii. Ql=ql+1;
iii. Signal();
iv. Repeat step 9
v. Return Rpage to CIP.
vi. Ql=ql-1;
Step 10: Else/*else of step 9*/
10.1. If(request is from proxy server)&&(MB==1)&&(Same Cluster)
(1) Accept Connection ();
(2) wait();
(3) ql=ql+1;
(4) signal();
(5) Return Rpage to Proxy Server.
(6) ql=ql-1;
10.2 .Else If(request is from proxyserver)&&(MB==1)&&(Different Cluster)/*else of 10.1*/
(1) Accept Connection();
(2) wait();
(3) ql=ql+1;
(4) signal ();
(5) Return rpage to proxy server.
(6) ql=ql-1;
10.3 Else if (request from proxy server) && (MB==0)/* Else of 10.2*/
(1) Accept Connection ();
(2) wait ();
(3) ql=ql+1;
(4) signal();
(5) Search Metadata();
(6) if(DataFound)
(7) GoTo Step 9.5
10.4 Else /*else of 10.3.(6)*/
(1) Send Request to Origin server.
(2) Accept Connection from origin server.
(3) If datafound

- Send Rpage to proxy server.

(4) Else /* Else of 10.4.(3)*/

- B. No page exist.

Step 11. If (Request is from origin server)
11.1. Accept Connection();
1.2. Update metadata & cache;

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

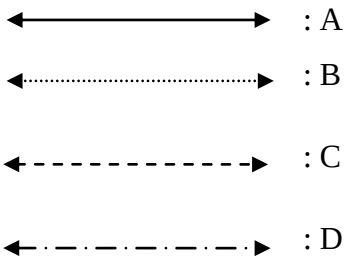
Vol. 5, Issue 5, May2017

Step 12 Else If(ps2.ql<80)/*Else of step 6*/

12.1. Forward request of client to ps2.

Step 13. Else Forward request of client to ps3. /*Else of step 12*/

Step 14. Else Send Request of client to any other cluster. /*Else of step 5*/



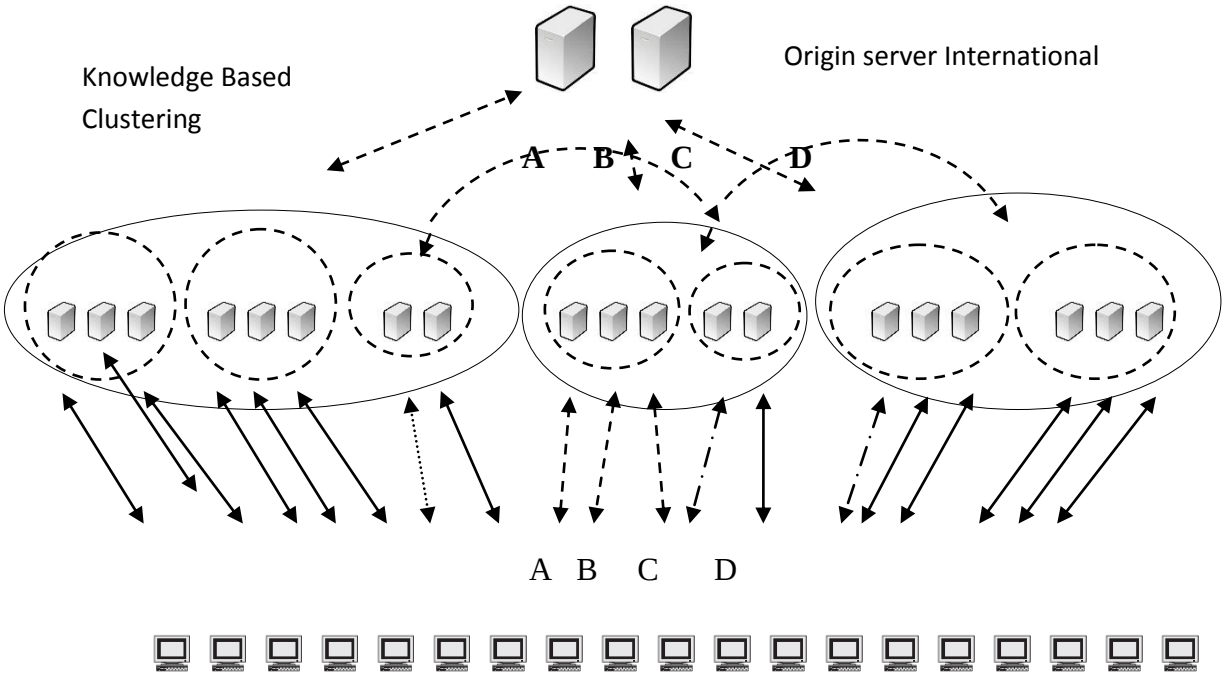


Fig.1 Overall Architecture of Proposed System.

The algorithm works mainly on four phases, when client send request for some pages on proxy server.The queuelength is being checked ,suppose queuelength of cluster is 150 (fixed) then each time when request send to particular cluster or proxy server it increases by 1 if requested data found in current knowledge cluster if not found then check in it's metadata.if it's found in neighbour cluster then match bit of knowledge based cluster increased by one.

This works in following four phases:

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 5, May2017

Process A is, when client send request to a proxy server and proxy server search in its metadata and make connection with matched proxy server in same cluster and get the response and in turn send response to client.

Process B is, when client send request to proxy server and proxy server search in its metadata and no match found there then forward request to the next to neighbor cluster and proxy server search in its metadata if match found then get response in same way otherwise send request to the next cluster in array.
Process C is, when client send request to proxy server and proxy server search in its metadata and no match found there then forward request to the next to neighbor cluster and search in its metadata if match found then get response in same way otherwise send request to the next cluster in array and if data not found in all the clusters then request send to the origin server.
Process D, when client send request to proxy server and proxy server search in its metadata if it is found in neighbor cluster then send request to that proxy server and get response in the same way .

VI. RESULTS AND DISCUSSION

Based on this algorithm we get better results for hit ration and latency time as compared to previous algorithm.These graphs show the results of previous work and current work

1. Result from the above graphs is population has effect on the Latency Time.
2. Hit ratio gradually increases with the smaller number of population.

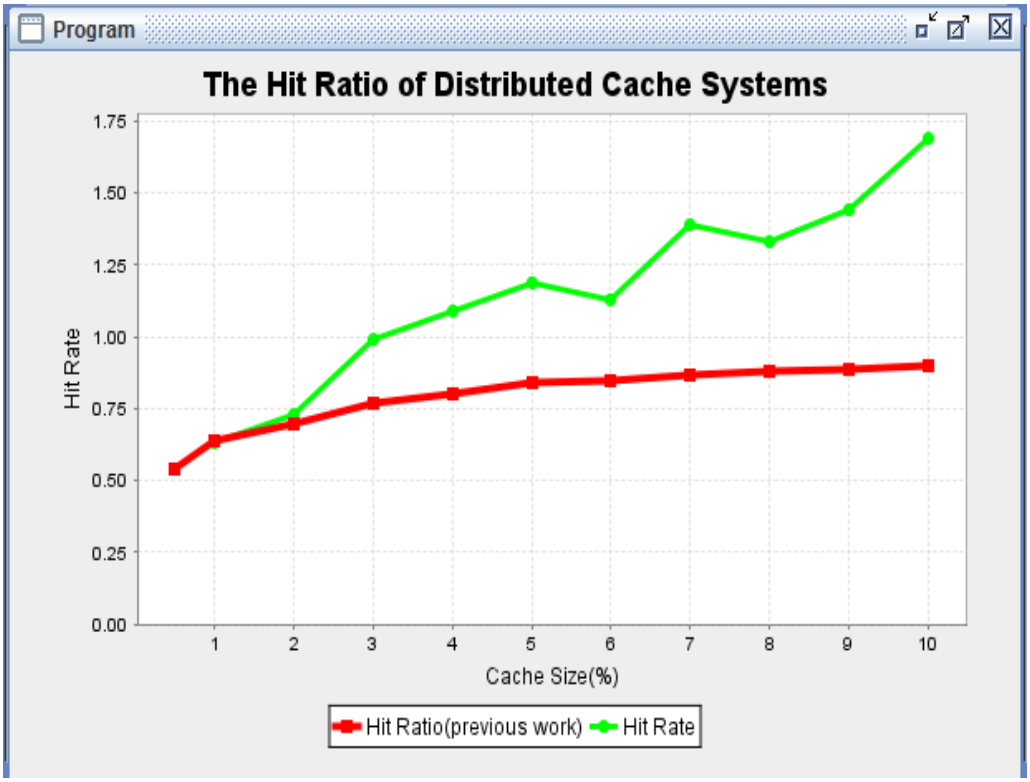


Fig 2. Hit ratio increases

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 5, May2017

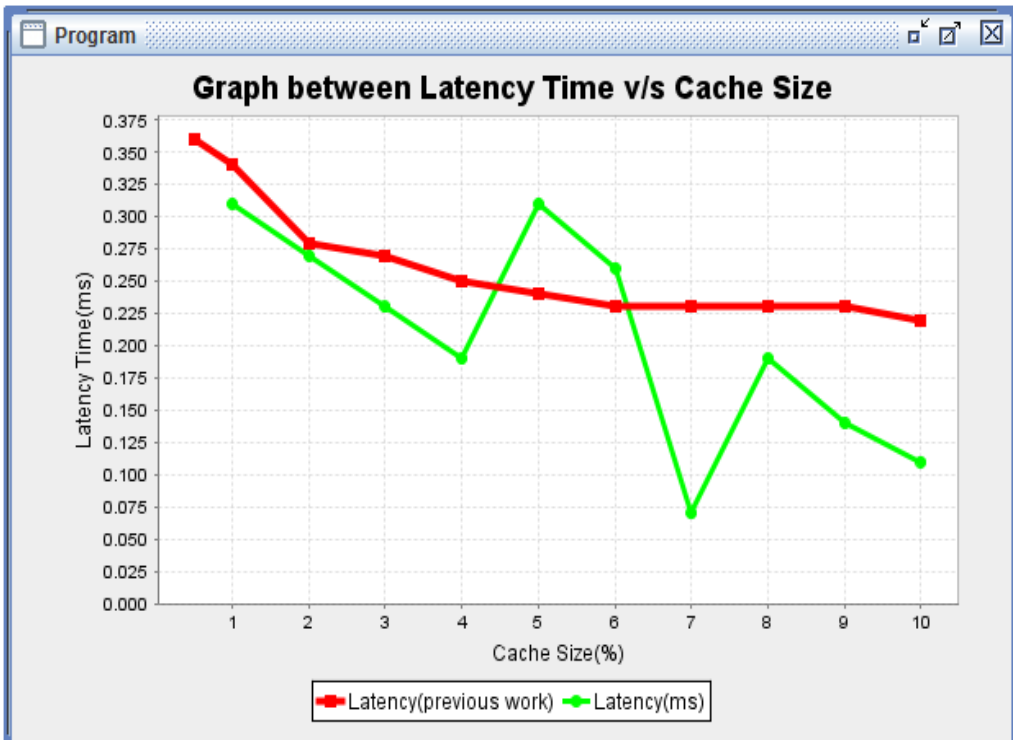


Fig 3.Latency time decreases

VII .CONCLUSION AND FUTURE WORK

As Web services turns out to be increasingly popular, clients are enduring system congestion and server over-burdening. Awesome endeavors have been made to enhance Web performance. Web storing is perceived to be one of the successful methods to mitigate server bottleneck and decrease network traffic, in this way limit the client access latency. In this work, I proposed a algorityhm to make a more robust system and to reduce the Extra Overhead, takes care of the issue of Cache Coherence (Get if Modified), issue of Scalability alongside taking care of every one of these issues it additionally enhances the Hit Ratio and the Latency Time .By looking over past deals with Web caching.we see that there are still some open issues in Web storing, for example, intermediary situation, reserve directing, dynamic information reserving, adaptation to internal failure, security, and so on. The research frontier in Web performance change lies in creating productive, scalable, robust, adaptive, and stable Web caching scheme that can be easily deployed in current and future network. Further we can improve performance by adding page replacement algorithm and till now we have used static and dynamic hierarchies, can improve by adding concept of pure dynamic hierarchies.

REFERENCES

[1] Dimitrievski, Ace, Vladimir Trajkovik, and Sonja Filiposka."Web caching and content switching for performanc of content delivery " Telecommunications Forum Telfor (TELFOR), 2015 23rd. IEEE, 2015.
[2] Hasslinger, Gerhard, Konstantinos Ntougias, and Frank Hasslinger. "A new class of web caching strategies for content delivery." Telecommunications Network Strategy and Planning Symposium (Networks), 2014 16th International. IEEE, 2014.
[3] Kumar, Rajeev, and Hina Hashmi. "Semantic Web System using Web Caching Algorithm at Origin Server for different Web Services"(2014).
[4] Sarhan, Amany, Ahmed M. Elmogy, and Sally Mohamed Ali. "New web cache replacement approaches based on internal requests factor." Computer Engineering & Systems (ICCES), 2014 9th International Conference on. IEEE, 2014.
[5] Tiwari, Rajeev, and Neeraj Kumar. "Dynamic Web caching: For robustness, low latency & disconnection handling." Parallel Distributed and Grid Computing (PDGC), 2012 2nd IEEE International Conference on. IEEE, 2012.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 5, May2017

[6] Sathiyamoorthi, V., and V. Murali Bhaskaran. "Web caching through modified cache replacement algorithm." Recent Trends In Information Technology (ICRTIT), 2012 International Conference on. IEEE, 2012.
[7] Tiwari, Rajeev, and Neeraj Kumar. "A novel hybrid approach for web caching." Innovative Mobile and Internet Services in ubiquitous Computing
(IMIS), 2012 Sixth International Conference on. IEEE, 2012.
[8] Xincan, Fan."Web caching system research and implementation of cluster."Electronic and Mechanical Engineering and Information Technology (EMEIT), 2011 International Conference on. Vol. 2. IEEE, 2011.
[9] Atassi, Mohamed R., Sherif G. Aly, and Amr El -Kadi. "Cooperative web caching of dynamic web content. " Computer Systems and Applications (AICCSA), 2011 9th IEEE/ACS International Conference on IEEE, 2011.
[10] Bitarafan, Katayoon, et al. "Mathematical modelling approach in web cache scheme." Research and Innovation in Information Systems (ICRIIS), 2011 International Conference on IEEE, 2011.
[11] Liu, Hai, and Maobian Chen. "Evaluation of web caching consistency." Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference on. Vol. 5 IEEE, 2010.
[12]. Load Balancing through distributed Web Caching with clusters, Rajeev Tiwari, Gulista, Proceeding of the 2010 Springer,Chennai,India.
[13] Tay, T. T., and Y. Zhang. "Peer-distributed web caching with incremental update scheme." IEE Proceedings-Communications 152.3 (2005)
[14]. A. Bestavros and C. Cunha, "Server-initiated document dissemination for the WWW", IEEE Data Engineering Bulletin,*IEEE* Transactions on Knowledge and Data Engineering, v.15 n.5, p.1266-1276, *September* 2003
[15] Z. Wang, Cachemesh: "A distributed cache system for World Wide Web", Web Cache Workshop, Pages: 1 – 10, Year of Publication: 2003