International Journal of Research in Science and Technology http://www.ijrst.com

(IJRST) 2015, Vol. No. 5, Issue No. III, Jul-Sep **e-ISSN: 2249-0604; p-ISSN:2454-180X**

MAINTAINING STRONG CACHE CONSISTENCY USING FUZZY INTERFACE SYSTEM

***Prerna Purohit, **Mahesh Manchanda**

**M.Tech CSE, Graphic Era Hill University
Dehradun, India
**Assistant Professor, CSE Departement
Graphic Era Hill University
Dehradun, India*

ABSTRACT

The World Wide Web is of an increasing growth in size which results in network congestion, server overloading and latency. Web caching is one approach which reduces these drawbacks. However one drawback of web caching is that it is not able maintain consistency of cache data. Web Consistency is a approach which maintain consistency of cache documents. This paper proposed Consistency policies with fuzzy interface system with input parameters such as Frequency, Latency, Bytesent and stale hits of web objects for maintaining both consistency and caching. Because, Unlike another replacement policies(LRU, LFU, FIFO etc); fuzzy interface system not only consider arrival time but also consider parameters related to web objects for deciding cacheable or not.

Keywords: *World Wide Web, Proxy Server, Cache Consistency, Invalidation, Polling-every-time, Leases, Web Caching, Fuzzy Interface System*

1.INTRODUCTION

The WWW has become the most successful application over the internet since it provide access to wide range of information and services and putting much demand on limited network infrastructure. A possible solution to the problems of increasing network traffic is addition of some new resources to the network infrastructure and distribution of network congestion or traffic across more resources. Proxy server is one popular approach to reduce these drawbacks. In a proxy caching architecture, clients request data from a proxy; the proxy services user requests using locally cached data or by fetching the requested data from the server. By caching frequently accessed objects, a proxy can reduce the load on network infrastructure and servers as well as client access latencies. Proxy server acts as an intermediary between the web server and the web client requesting the web documents. However proxy server cache has mainly two drawbacks 1) proxy cache has limited space, 2) consistency of cached copy.

Web Caching is one approach which uses replacement policy to maintain problem of limited size of cache. Web caching replacement policies remove objects from proxy cache to making space for new coming data. There are following replacement policies such as LRU, LFU, FIFO, Size etc. These replacement policies remove objects from cache based on arrival time of objects and not considering objects hit ratio, bytesent etc. Fuzzy interface system is a technique through which we can delete objects on the basis of their hit ratio, bytesent, latency etc. Fuzzy interface system is a approach which considering parameters for removing objects from cache[3].

Web Consistency is a approach through which we can maintain consistency of cached documents. Cache Consistency is concern to determine that content in the cache should be up to date when object get change on origin server. Cache Consistency is classified into two groups weak consistency and strong consistency. Weak consistency are the consistency model in which stale documents might be return to clients were as strong consistency are the consistency model in which no stale documents return to clients. Different example of weak consistency approaches are TTL(time-to-live), Client polling etc and different example of strong consistency approaches are Invalidation, Polling-every-time, and Leases.

Here weak consistency defined TTL as a time-to-live in which client considers a cache copy up to date if its time-to-live has not expired and client polling is defined in which a client periodically check the freshness of cached copy[1]. In weak consistency user have to be aware that the browser may occasionally display a stale page. And to prevent stale information from being transmitted to users, a proxy must ensure that locally cached document is consistent with that stored on servers. To make sure about requested documents is up-to-date, user has to instruct the browser to reload web page which means contacting a web server to check the freshness of a cached copy.

Here as strong consistency define Invalidation as a approach in which web server keeps tracks of all the client sites that cache a web document and when a document is change sends invalidation messages to clients. And in polling-every-time approach when a client request a document the cache first contact the web server to validate the cache copy and then returns the copy to clients were as in case of Leases, it provide a smooth trade-off between the state space overhead and the number of control message exchanged[2]. Lease is defined as a duration which denotes the interval of time in which the web server agrees to notify the proxy if the document is modified.

2.PROPOSED TECHNIQUES

Web caching is not sufficient for reducing network traffic from origin server, we also have to focus on consistency of cached document. So, here in our proposed techniques we are working on both consistency and caching. Some example of web consistency approaches are TTL, Client polling, polling-every-time, invalidation, leases. In our proposed techniques we have used strong consistency approaches that is invalidation, polling-every-time and leases to minimize stale documents in cache. As we know that cache replacement policies remove objects from cache, based on arrival time of documents, here we have used fuzzy interface system for considering parameters for cache objects to decide cacheable or not. So to maintain bandwidth, latency, hit ratio and staleness of web objects we have proposed a technique "Maintaining strong cache consistency using fuzzy interface system". Here four parameters are define for maintaining consistency and caching, these parameters are frequency, latency, byte ratio and staleness. The three parameters frequency, latency, byte ratio are define for maintaining web objects in cache, were as staleness is define for maintaining consistency of web object. First parameter is frequency which means number of cache hits of the web objects for a period of time in the proxy server cache and frequency value would be given to the web objects according to their hit ratios. The second parameter is Latency which means time to get the requested web content from server and if the web contents required higher Latency, it would be given higher priority. Because, if the high Latency web contents are place into cache of proxy server, the repetitive client requests would be deliver quickly to the users from proxy server thus reducing latency. The third parameter is Bytesent, which means the size of the web content or the byte sent by the server to the Proxy Server for the requested web content. If the content has big size, then it would be given high priority of cache, because higher web content size required more bandwidth for transmission, hence saving bandwidth. The fourth parameter is staleness, which means document in proxy server cache should be up-to-date when document change in origin server. Staleness is a parameters which tells the freshness of document in proxy server cache. This factor is handled based on consistency approaches that is invalidation, polling-every-time and leases.

1-When the user request web pages, all documents are placed in proxy server cache. The latency and bytesents information of the web contents are saved in the proxy server data store. The proxy server maintains cache hits and stale hits of the each web contents for a period of time and it is acting as Frequency and consistency of the web contents. After a period of time, when consistency of web document get expire or cache copy of web object updated in origin server, the consistency policy applied here for maintaining freshness of web objects.

2-The consistency policy; invalidation, polling-every-time and leases is employed here to maintain consistency of web objects. In invalidation method when a document is fetch from a origin server to proxy cache than Web server keeps track of all the client sites that cache a web document and when the document is changed, sends invalidation messages to the clients. In polling-every-time, every time a user request a document and there is a cached copy, the cache first contact the web server to validate the cache copy and then return the copy to user. In lease approach server grant a lease to each request from a proxy. The lease duration denotes the interval of time during which the web server agrees to notify the proxy if the object is modified and after the expiration of lease proxy must sent a message requesting renewal of the lease. After a period of time when cache of proxy server would filled up, than new web object request by users in the cache of proxy server, then at that time replacement policy is used for freeing space.

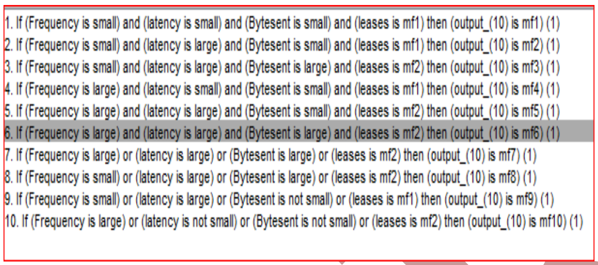
3-The Sugeno type Fuzzy Inference System is employed here to find a rank of web contents. This rank helps in replacement policy. The rank is based on the input values of staleness, Byte Sent, Frequency and Latency and the low valued rank web documents would be removed first from cache of proxy server. If all the web documents consider high rank then, arbitrarily any web contents would removed for new coming web documents into cache memory of Proxy Server.

3.IMPLEMENTATION AND PERFORMANCE ANALYSIS

In this section the performance of three consistency approaches are evaluated individually. All approaches are tested on metlab with same number of parameters and rules.

3.1Leases

Lease is technique through which we can maintain state space overhead and control message overhead. We have developed the rule with fuzzy interface system in matlab. The rules and algorithms used are as follows:



```
1. If (Frequency is small) and (latency is small) and (Bytesent is small) and (leases is mf1) then (output_(10) is mf1) (1)
2. If (Frequency is small) and (latency is large) and (Bytesent is small) and (leases is mf1) then (output_(10) is mf2) (1)
3. If (Frequency is small) and (latency is large) and (Bytesent is large) and (leases is mf2) then (output_(10) is mf3) (1)
4. If (Frequency is large) and (latency is small) and (Bytesent is small) and (leases is mf1) then (output_(10) is mf4) (1)
5. If (Frequency is large) and (latency is large) and (Bytesent is small) and (leases is mf2) then (output_(10) is mf5) (1)
6. If (Frequency is large) and (latency is large) and (Bytesent is large) and (leases is mf2) then (output_(10) is mf6) (1)
7. If (Frequency is large) or (latency is large) or (Bytesent is large) or (leases is mf2) then (output_(10) is mf7) (1)
8. If (Frequency is small) or (latency is large) or (Bytesent is large) or (leases is mf2) then (output_(10) is mf8) (1)
9. If (Frequency is small) or (latency is large) or (Bytesent is not small) or (leases is mf1) then (output_(10) is mf9) (1)
10. If (Frequency is large) or (latency is not small) or (Bytesent is not small) or (leases is mf2) then (output_(10) is mf10) (1)
```

3.1.1Algorithm Leases

```
Setup
  intialize required variables Setup web pages (data) for simulation

Working
Step1.  for i ← 1 to N
Step2.   current_byteset ← 0 to 258
Step3.   current_latency ← 0-2  /* minutes to load page */
Step4.   current_freq ← 0-10 /* no of times page visited */
Step5.   byteset = Max_byteset/current_byteset
Step6.   latency = Max_Latency/current_latency
Step7.  freq = Max_Freq/current_freq
Step8.  stale ← 0-7  /* days allotted to pages */
Step9.  myfuzzy ← read fuzzy
Step10.  mv ←  evaluate rules on the basis of freq;latency;byteset;stale
Step11.   if the probability is more than 0.5 then /* server agrees to notify the proxy if the object is modified */
```

```
Step12.  cache the page
Step13.  else
Step14.  Send a message to renew lease
Step15.  end if
Step16.  Stime ← store time
Step17. end for
```

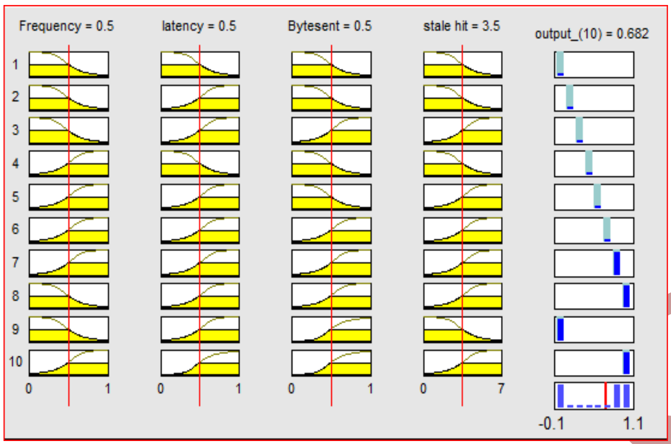
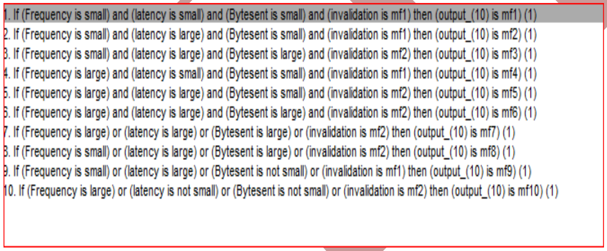


Fig1 Performance of Lease in terms of parameters

3.2Invalidation

Invalidation is a techniques through which we can maintain strong consistency and control message overhead. We have developed the rule with fuzzy interface system in matlab. The rules used are as follows:



1. If (Frequency is small) and (latency is small) and (Bytesent is small) and (invalidation is mf1) then (output_(10) is mf1) (1)
2. If (Frequency is small) and (latency is large) and (Bytesent is small) and (invalidation is mf1) then (output_(10) is mf2) (1)
3. If (Frequency is small) and (latency is large) and (Bytesent is large) and (invalidation is mf2) then (output_(10) is mf3) (1)
4. If (Frequency is large) and (latency is small) and (Bytesent is small) and (invalidation is mf1) then (output_(10) is mf4) (1)
5. If (Frequency is large) and (latency is large) and (Bytesent is small) and (invalidation is mf2) then (output_(10) is mf5) (1)
6. If (Frequency is large) and (latency is large) and (Bytesent is large) and (invalidation is mf2) then (output_(10) is mf6) (1)
7. If (Frequency is large) or (latency is large) or (Bytesent is large) or (invalidation is mf2) then (output_(10) is mf7) (1)
8. If (Frequency is small) or (latency is large) or (Bytesent is large) or (invalidation is mf2) then (output_(10) is mf8) (1)
9. If (Frequency is small) or (latency is large) or (Bytesent is not small) or (invalidation is mf1) then (output_(10) is mf9) (1)
10. If (Frequency is large) or (latency is not small) or (Bytesent is not small) or (invalidation is mf2) then (output_(10) is mf10) (1)

3.2.1Algorithm Invalidation

```
Setup
Initialize required variables
Setup web pages (data) for simulation
Working
Step1.   for i ← 1 to N
Step2.   current_byteset ← 0 to 258
Step3.   current_latency ← 0-2  /* minutes to load page */
```

```
Step4.   current_freq ← 0-10 /* no of times page visited */
Step5.   byteset = Max_byteset/current_byteset
Step6.   latency = Max_Latency/current_latency
Step7.   freq = Max_Freq/current_freq
Step8.   invalid ← 0-1 * document changed -1, 0-no change*/
Step9.   myfuzzy ← read fuzzy
Step10. mv ←  evaluate rules on the basis of freq;latency;byteset;invalid
Step11.   if the probability is more than 0.5 then /* contents are changed  */
Step12.   Send message to proxy server
Server ← send notification that contents have been modified
Step13.   else
Step14.   No action required
Step15.   end if
 Step16.   Stime ← store time
Step17.   end for
```

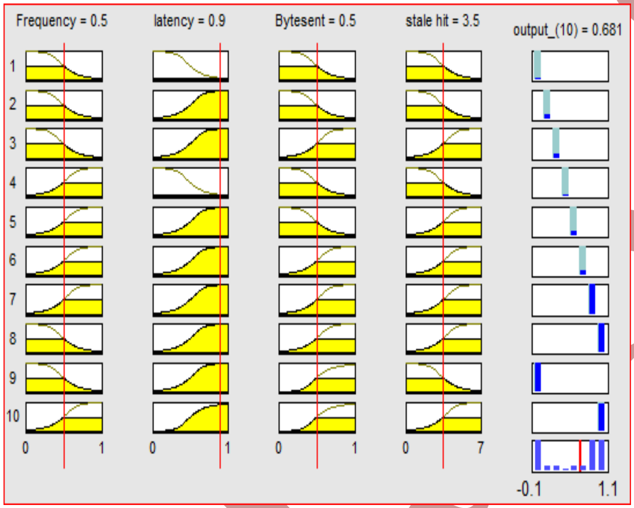
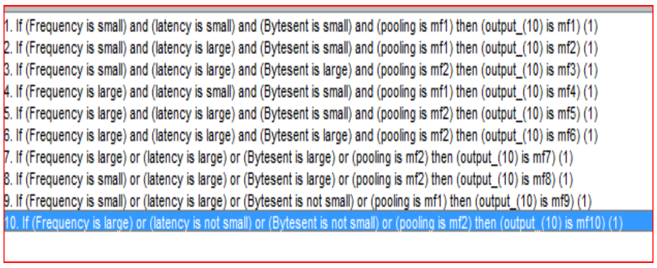


Fig.2Performance of Invalidation in terms of parameters

3.3Polling-every-time

Polling-every-time is a techniques through which we can maintain strong consistency. We have developed the rule with fuzzy interface system in matlab. The rules used are as follows:



```
1. If (Frequency is small) and (latency is small) and (Bytesent is small) and (pooling is mf1) then (output_(10) is mf1) (1)
2. If (Frequency is small) and (latency is large) and (Bytesent is small) and (pooling is mf1) then (output_(10) is mf2) (1)
3. If (Frequency is small) and (latency is large) and (Bytesent is large) and (pooling is mf2) then (output_(10) is mf3) (1)
4. If (Frequency is large) and (latency is small) and (Bytesent is small) and (pooling is mf1) then (output_(10) is mf4) (1)
5. If (Frequency is large) and (latency is large) and (Bytesent is small) and (pooling is mf2) then (output_(10) is mf5) (1)
6. If (Frequency is large) and (latency is large) and (Bytesent is large) and (pooling is mf2) then (output_(10) is mf6) (1)
7. If (Frequency is large) or (latency is large) or (Bytesent is large) or (pooling is mf2) then (output_(10) is mf7) (1)
8. If (Frequency is small) or (latency is large) or (Bytesent is large) or (pooling is mf2) then (output_(10) is mf8) (1)
9. If (Frequency is small) or (latency is large) or (Bytesent is not small) or (pooling is mf1) then (output_(10) is mf9) (1)
10. If (Frequency is large) or (latency is not small) or (Bytesent is not small) or (pooling is mf2) then (output_(10) is mf10) (1)
```

3.3.1Algorithm Polling-Every-Time

```
Setup
Initialize required variables
Setup web pages (data) for simulation

Working
Step1. for i ← 1 to N
Step2.   current_byteset ← 0 to 258
Step3.   current_latency ← 0-2  /* minutes to load page */
Step4.   current_freq ← 0-10 /* no of times page visited */
Step5.    byteset = Max_byteset/current_byteset
Step6     latency = Max_Latency/current_latency
Step7.    freq = Max_Freq/current_freq
Step8.    pooling ← 0-1 /* validty1-Valid 0-not valid */
Step9.    myfuzzy ← read fuzzy
Step10. mv ← evaluate rules on the basis of freq;latency;byteset;pooling
Step11.   if the probability is more than 0.5 then /* contents are valid  */
Step12.    return cache copy of the page
Step13.   else
Step14.   contents not valid; no caching
Step15.  end if
 Step16.  Stime ← store time
Step17. end for
```

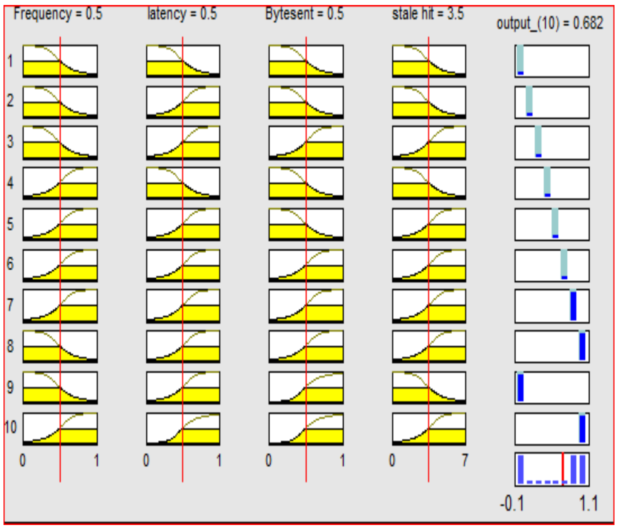


Fig.3Performance of polling-every-time in terms of parameters

4.COMPARISON

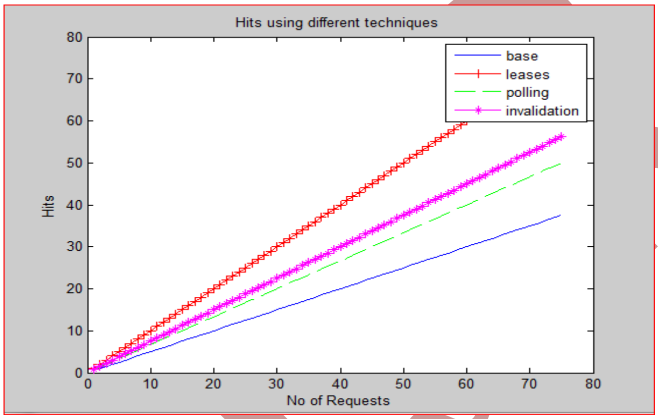


Fig.4 Comparison of both the algorithms in terms of hit ratio

From the above graph, we can infer that:

1)The proposed algorithm web consistency using fuzzy interface system works better than web caching using fuzzy interface system in respect of hit ratio.
2)In case of web consistency using fuzzy interface system we are maintaining both consistency and limited size of cache. We take four parameters as input such as frequency, latency, bytesents and stale hit.

5.CONCLUSION AND FUTURE WORK

The principal of this study is to propose a technique which is efficient in maintaining consistency and limited size of proxy cache and also have been several studies about algorithms for consistency and caching. This chapter summarizes the work of this paper and future work related to the proposed technique. In this paper we have maintain limited size of cache and consistency of cache data with the help of consistency approaches and fuzzy interface system. While previous studies have addressed comparison of consistency approaches and concentrating either on caching or in consistency, we have maintain proxy server cache with the help of caching policy i.e Fuzzy Interface System and consistency approach i.e is Leases, invalidation and polling-every-time. There are a number of future research directions based on the work presented in this thesis. Using the modified version of caching and consistency approaches can reduce proxy server issues and make proxy server more efficient.

6.REFERENCES

[1] Peio Cao and Chengjie Liu Departement of computer science, university Wisconsin-Madison, Madison WI 53706 {cao, chengjie} @cs.wisc.edu , maintaining strong cache consistency in the world wide web.

[2] Venkata Duvvuri, Prashant Shenoy, Member, IEEE, and Renu Tewari Adaptive Leases: A Strong Consistency Mechanism for the World Wide Web july/august 2003

[3] Anish Kumar Saha Assistant Professor CSE Department NIT, Agartala, INDIA Partha Pratim Deb M.Tech CSE Netaji Subhash Engg College West Bengal, India Moutushi Kar, D. Rudrapal Assistant Professor CSE Department NIT, Agartala, INDIA, An Optimization Technique of Web Caching using Fuzzy Inference System International Journal of Computer Applications (0975 – 8887) Volume 43– No.17, April 2012.

[4] Bhuvan Urgaonkar University of Massachusetts – Amherst Anoop George Ninan University of Massachusetts – Amherst Mohammad Salimullah Raunak University of Massachusetts – Amherst Prashant Shenoy University of Massachusetts – Amherst Krithi Ramamritham University of Massachusetts – Amherst for Maintaining mutual consistency for cached web objects, 2001.

[5] Jia Wang, " A Survey of Web Caching Schemes for the Internet" ACM SIGCOMM 2000.

[6] P. Cao, J. Zhang, and K. Beach, "Active Cache: Caching dynamic contents on the web"

[7] Stefan Podlipnig And Laszlo Boszormenyi, "A Survey of Web Cache Replacement Strategies", ACM Computing Surveys, Vol. 35, No. 4, December 2003, pp. 374–398.P. Cao and C. Liu.

Maintaining Strong Cache Consistencyin the World-Wide Web. In Proceedings of the Seventeenth International Conference on Distributed Computing Systems May.

[8]. L. Y. CAO and M. T. OZSU “Evaluation of Strong Consistency Web Caching Techniques” University of Waterloo, School of Computer Science, Waterloo, ON, Canada N2L 3G1

[9].Antonis Sidiropoulos, George Pallis, Dimitrios Katsaros, Konstantinos Stamos,Athena Vakali, Yannis Manolopoulos “Prefetching in Content Distribution Networks via Web Communities Identification and Outsourcing” World wide Web (2008) 11:39–70; Springer.

[10].Hakan Grahn, Per Stenstrom and Michel Dubois. “Implementation and evaluation of update-based cache protocols under relaxed memory consistency models”. Future Generation Computer Systems, 11(3), June 1995

[11]. J.-H. Kim and N. H. Vaidya, “A cost-comparison approach for adaptive distributed shared memory” in Proc. of 1996 International Conference onSupercomputing, pp. 44-51, May.