International Journal of Engineering Research and Development
***e*-ISSN: 2278-067X, *p*-ISSN: 2278-800X, www.ijerd.com**
Volume 3, Issue 1 (August 2012), PP. 01-08

Improving the Performance of Browsers Using Fuzzy Logic

K Muralidhar[1], Dr. N Geethanjali[2]
[1]*Research Scholar, Sri Krishnadevaraya University, Anantapur*
[1]*Assistant Professor, Department of Computer Applications, Sri Sai College of IT and Management, Kadapa*
[2]*Research Supervisor, Associate Professor and Head,*
Department Of Computer Science and Technology, Sri Krishnadevaraya University, Anantapur

Abstract— The large popularity of web services and applications makes their performance very critical. Reducing the latency of retrieve web pages has become a real challenge. Caching is widely used for this purpose. Web caching is a well-known strategy for improving performance of Web-based system by keeping web objects that are likely to be used in the near future close to the client. Most of the current Web browsers still employ traditional caching policies that are not efficient in web caching. This research proposes a splitting browser cache to two caches, instant cache and durable cache. Initially, a web object is stored in instant cache, and the web objects that are visited more than the pre-specified threshold value will be moved to durable cache. Other objects are removed by Least Recently Used (LRU) algorithm as instant cache is full. More significantly, when the durable cache saturates, a fuzzy system is employed in classifying each object stored in durable cache into either cacheable or uncacheable object. The old uncacheable objects are candidate for removing from the durable cache. By implementing this mechanism, the cache pollution can be mitigated and the cache space can be utilized effectively. Experimental results have revealed that the proposed approach can improve the performance in terms of hit ratio (HR) and* Latency Saving Ratio (LSR) *when compared to LRU and Least Frequently Used (LFU).

Keywords—FRA, Fuzzy logic, Fuzzy inference, LFU, LRA, page replacement algorithm.

I. INTRODUCTION

With the ever-widening speed gap between computing elements and memory units in today's computing environment one of the important means to improve the performance of Web service is to employ web caching mechanism. Web caching is a well-known strategy for improving performance of Web based system. The web caching caches popular objects at location close to the clients, so it is considered one of the effective solutions to avoid Web service bottleneck, reduce traffic over the Internet and improve scalability of the Web system[1]. The web caching is implemented at client, proxy server and original server [2]. However, the client-side caching (browser caching) is economical and effective way to improve the performance of the World Wide Web due to the nature of browser cache that is closer to the user [3,4].

Three important issues have profound impact on caching management namely: cache algorithm (passive caching and active caching), cache replacement and cache consistency. However, the cache replacement is the core or heart of the web caching; hence, the design of efficient cache replacement algorithms is crucial for caching mechanisms achievement [5]. In general, cache replacement algorithms are also called web caching algorithms [6].

Since the apportioned space to the client-side cache is limited, the space must be utilized judiciously [3]. The term "cache pollution" means that a cache contains objects that are not frequently used in the near future. This causes a reduction of the effective cache size and affects negatively on performance of the Web caching. Even if we can locate large space for the cache, this will be not helpful since the searching for object in large cache needs long response time and extra processing overhead. Therefore, not all Web objects are equally important or preferable to store in cache. The setback in Web caching consists of what Web objects should be cached and what Web objects should be replaced to make the best use of available cache space, improve Hit Rates, reduce network traffic, and alleviate loads on the original server.

Most web browsers still concern traditional caching policies [3, 4] that are not efficient in web caching [6]. These policies suffer from cache pollution problem either cold cache pollution like the least recently used (LRU) policy or hot cache pollution like the least frequently used (LFU) and SIZE policies [7] because these policies consider just one factor and ignore other factors that influence the efficiency the web caching. Consequently, designing a better-suited caching policy that would improve the performance of the web cache is still an incessant research [6, 8].

Many web cache replacement policies have been proposed attempting to get good performance [2, 9, 10]. However, combination of the factors that can influence the replacement process to get wise replacement decision is not easy task because one factor in a particular situation or environment is more important than others in other environments [2, 9]. In recent years, some researchers have been developed intelligent approaches that are smart and adaptive to web caching environment [2]. These include adoption of back-propagation neural network, fuzzy systems, evolutionary algorithms, etc. in web caching, especially in web cache replacement.

In this paper, the proposed approach grounds instant cache that receives the web objects from the Internet directly, while durable cache receives the web objects from the instant cache as these web objects visited more than pre-specified threshold value. Moreover, fuzzy system is employed to predict web objects that can be re-accessed later. Hence, unwanted objects are removed efficiently to make space of the new web objects.

II. LITERATURE REVIEW

Related Works on Web Caching

Web caching can be implemented at different levels, i.e., client, server, network. The web browser and the web server are responsible for caching at the client and at the server side, respectively. Proxy servers are used for caching at network level.

The efficiency and performance of proxy caches mainly depend on their design and management. Replacement policies play a key role for the effectiveness of caching. The goal of replacement policies is to make the best use of the available resources by dynamically selecting the pages to be cached or evicted.

Replacement policies have been extensively studied in the literature. A few policies, e.g., Least Recently Used (LRU), Least Frequently Used (LFU), are direct extensions of the traditional replacement algorithms typical of the operating system domain. Other policies, e.g., Greedy Dual Size (GDS), have been explicitly designed for web environments.

Many generalizations of both traditional and web specific policies have been proposed. In [11], the LRU algorithm is generalized such as to take into account access costs and expiration times. A generalization of the GDS algorithm that incorporates short term temporal locality and long term popularity of web request streams is presented in [12]. In [13], randomized algorithms have been applied for approximating any existing web cache scheme. In [14], the replacement strategies are addressed in the framework of a model for optimizing the content of the web cache. The model is based on genetic algorithm or an evolutionary programming scheme.

Although there are many studies in web caching, but research on web caching is still fresh. This section presents some existing web caching techniques based on ANN or fuzzy logic.

In [15], ANN has been used for making cache replacement decision. An object is selected for replacement based on the rating returned by ANN. This method ignored latency time in replacement decision. Moreover, the objects with the same class are removed without any precedence between these objects. An integrated solution of ANN as caching decision policy and LRU technique as replacement policy for script data object has been proposed in [16]. However, the most important factor in web caching, i.e., recency factor, was ignored in caching decision. Both pre fetching policy and web cache replacement decision has been used in [17]. The most significant factors (recency and frequency) were ignored in web cache replacement decision. Moreover, applying ANN in all policies may cause extra overhead on server. ANN has also been used in [6] depending on syntactic features from HTML structure of the document and the HTTP responses of the server as inputs. However, this method ignored frequency factor in web cache replacement decision. On other hand, it hinged on some factors that do not affect on performance of the web caching.

Although the previous studies have shown that the ANNs can give good results with web caching, the ANNs have the following disadvantage: ANNs lack explanatory capabilities, the performance of ANNs relies on the optimal selection of the network topology and its parameters, ANNs learning process can be time consuming, and ANNs are also too dependent on the quality and amount of data available [18, 19, 20].

The last fifteen years have seen development of a number of novel caching algorithms that have attempted to combine recency and frequency. There is an association between recency and frequency; if a recently used page is likely to be used soon, then such a page will be used frequently. The least recently frequently used (LRFU) policy is one of them [21, 22].

The LFFU policy associates a value with each block. This value is called CRF (Combined recency and frequency) and quantifies the likelihood that the block will be referenced in the near future. This value is calculated according to a mathematical equation that is going to mix recency and frequency and comes to a single decision parameter.

Although there is a correlation between recency and frequency but, as experiences show, this correlation is not the same for all kind of workloads. So, describing this relation with an exact mathematical formula is impossible. In real world situations, it would often be more realistic to find viable compromises between these parameters. For many problems, it makes sense to partially consider each of them. One especially straightforward method to achieve this is the modeling of these parameters through fuzzy logic. Using fuzzy rules we can combine these parameters as they are connected in real worlds. This research shares consideration of frequency, recency, and size in replacement decision.

Browser Web Caching

Caches are found in browsers and in any of the web intermediate between the user agent and the original server. Typically, a cache is located in the browser and the proxy [18]. A browser cache (client-side cache) is located in client. If we examine the preferences dialog of any modern web browser (like Internet Explorer, Safari or Mozilla), we will probably notice a cache setting. Since most users visit the same web site often, it is beneficial for a browser to cache the most recent set of pages downloaded. In the presence of web browser cache, the users can interact not only with the web pages but also with the web browser itself via the use of the special buttons such as back, forward, refresh or via URL rewriting. On other hand, a proxy cache is located in proxy. It works on the same principle, but in a larger scale. The proxies serve hundreds or thousands of users in the same way.

As cache size is limited, a cache replacement policy is needed to handle the cache content. If the cache is full when an object needs to be stored, then the replacement policy will determine which object is to be evicted to allow space for the new object. The goal of the replacement policy is to make the best use of available cache space, to improve Hit Ratios, and to reduce loads on the original server. The simplest and most common cache management approach is LRU algorithm, which removes the least recently accessed objects until there is sufficient space for the new object. LRU is easy to implement and proficient for uniform size objects such as in the memory cache. However, since it does not consider the size or the download latency of objects, it does not perform well in web caching [6].

Most web browsers still concern traditional replacement policies [3, 4] that are not efficient in web caching [6]. In fact, there are few important factors of web objects that can influence the replacement policy [2, 9, 10]: recency, i.e., time of

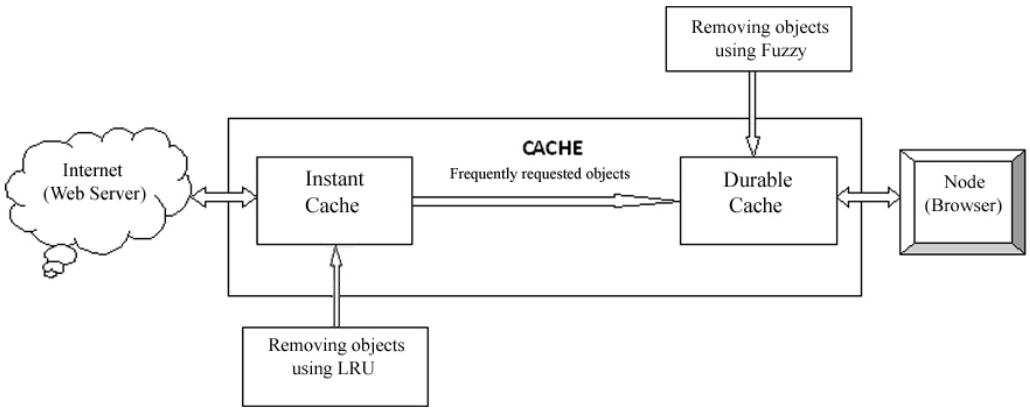
(since) the last reference to the object, frequency, i.e., number of the previous requests to the object, and size of the web object. These factors can be incorporated into the replacement decision. Most of the proposals in the literature use one or more of these factors. However, combination of these factors to get wise replacement decision for improving performance of web caching is not easy task because one factor in a particular situation or environment is more important than others in other environments [2, 9].

III. FRAMEWORK OF SMART BROWSER WEB CACHING SCHEME

Framework

This section, presents a framework of Smart Browser Web Caching Scheme. As shown in Fig. 1, the web cache is divided into instant cache that receives the web objects from the Internet directly, and durable cache that receives the web objects from the instant cache.

FIG. 1. **A framework of Smart Browser Web Caching Scheme.**



The concept of Fuzzy Logic (FL) was conceived by Lotfi Zadeh, a professor at the University of California at Berkley, and presented not as a control methodology, but as a way of processing data by allowing partial set membership rather than crisp set membership or non- membership. This approach to set theory was not applied to control systems until the 70's due to insufficient small-computer capability prior to that time. Professor Zadeh reasoned that people do not require precise, numerical information input, and yet they are capable of highly adaptive control. If feedback controllers could be programmed to accept noisy, imprecise input, they would be much more effective and perhaps easier to implement. Unfortunately, U.S. manufacturers have not been so quick to embrace this technology while the Europeans and Japanese have been aggressively building real products around it.

When the user navigates specific web page, all web objects embedded in the page are stored in instant cache primarily. The web objects that visited more than once will be relocated to durable cache for longer caching but the other objects will be removed using LRU policy that removes the oldest object firstly. This will ensure that the preferred web objects are cached for longer time, while the bad objects are removed early to alleviate cache pollution and maximize the hit ratio. On the contrary, when the durable cache saturates, the Fuzzy system is employed in replacement process by classifying each object stored in durable cache to cacheable or uncacheable object. The old uncacheable objects are removed initially from the durable cache to make space for the incoming objects if all objects are classified as cacheable objects, then our approach will work like LRU policy.

The main feature of the proposed system is to be able to store ideal objects and remove unwanted objects early, which may alleviate cache pollution. Thus, cache space is used properly. The second feature of the proposed system is to be able to classify objects to either cacheable or uncacheable objects. Hence, the uncacheable objects are removed wisely when web cache is full. The proposed system is also adaptive and adjusts itself to a new environment. Lastly, the proposed system is very flexible and can be converted from a client cache to a proxy cache using minimum effort. The difference lies mainly in the data size at the server which is much bigger than the data size at the client.

Fuzzy Inference System

Fuzzy logic is an extension of Boolean logic dealing with the concept of partial truth which denotes the extent to which a proposition is true. Whereas classical logic holds that everything can be expressed in binary terms (0 or 1, black or white, yes or no), fuzzy logic replaces Boolean truth values with a degree of truth. Degree of truth is often employed to capture the imprecise modes of reasoning that play an essential role in the human ability to make decisions in an environment of uncertainty and imprecision. Fuzzy Inference Systems (FIS) are conceptually very simple. They consist of an input, a processing and an output stage. The input stage maps the inputs, such as frequency of reference, recency of access, and so on, to the appropriate membership functions and truth values. The processing stage invokes each appropriate rule and generates a corresponding result. It then combines the results. Finally, the output stage converts the combined result back into a specific output value [23].

As discussed earlier the processing stage which is called inference engine is based on a collection of logic rules in the form of IF-THEN statements where IF part is called the “antecedent” and the THEN part is called the “consequent”. Typical fuzzy inference systems have dozens of rules. These rules are stored in a knowledgebase.

An inference engine tries to process the given inputs and produce an output by consulting an existing knowledgebase. The five steps toward a fuzzy inference are as follows:

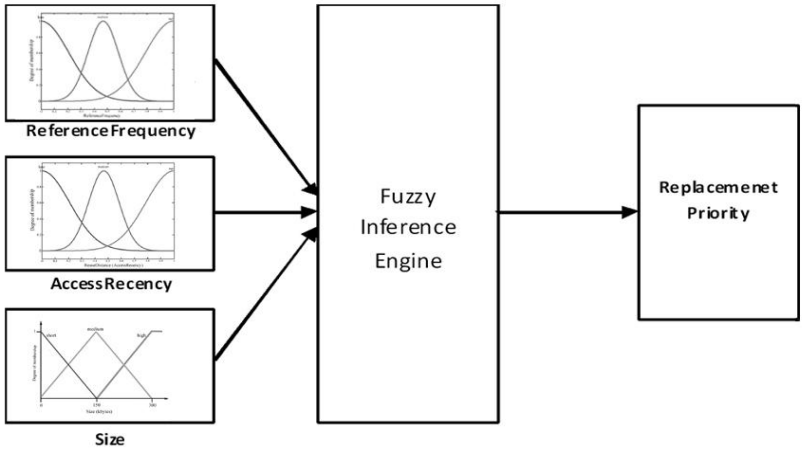
- Fuzzifying Inputs
- Applying Fuzzy Operators
- Applying Implification Methods
- Aggregating All Outputs
- Defuzzifying outputs

Fuzzifying the inputs is the act of determining the degree to which they belong to each of the appropriate fuzzy sets to which they belong to each of the appropriate fuzzy sets via membership functions. Once the inputs have been fuzzified, the degree to which each part of the antecedent has been satisfied for each rule is known. If the antecedent of a given rule has more than one part, the fuzzy operator is applied to obtain one value that represents the result of the antecedent for that rule. The implification function then modifies that output fuzzy set to the degree specified by the antecedent. Since decisions are based on the testing of all of the rules in an FIS, the results from each rule must be combined in order to make a decision. Aggregation is the process by which the fuzzy sets that represent the outputs of each rule are combined into a single fuzzy set. The input of the defuzzification process is the aggregated output fuzzy set and the output is a single value. This can be summarized as follows: mapping input characteristics to input membership functions, input membership function to rules, rules to a set of output characteristics, output characteristics to output membership functions, and the output membership function to a single-valued output.

IV. THE PROPOSED MODEL FOR FUZZY INFERENCE SYSTEM

The block diagram of Fuzzy Inference system is presented in Figure 2.

FIG. 2. **Inference system block diagram.**



In the proposed model, the input stage consists of three linguistic variables. The first one is the frequency of references. This parameter exploits the temporal locality of references. The second input variable is the recency of access i.e., time elapsed since last access. The last input variable is the size of the object. With the help of these three parameters this system is going to find out the replacement priority, which determines which page should be replaced.
The input variables mapped into the fuzzy sets as illustrated in Figures 3, 4, 5.

FIG. 3. Fuzzy sets corresponding to frequency of references

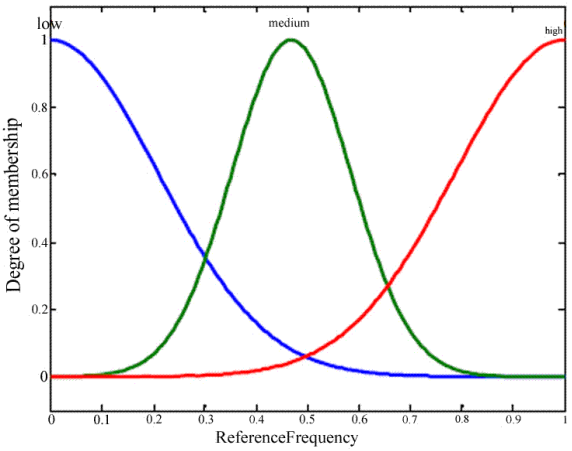


Fig 4. Fuzzy sets corresponding to recency of access

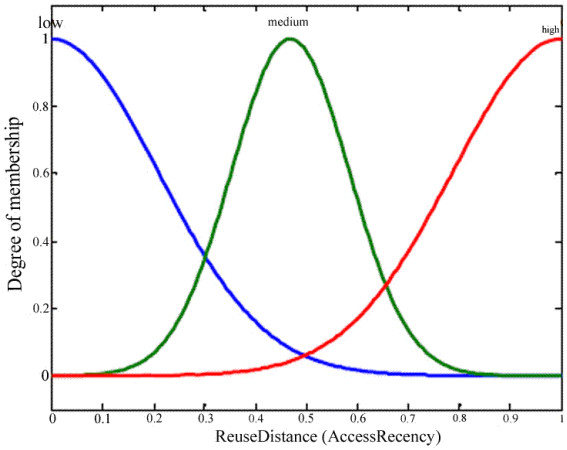
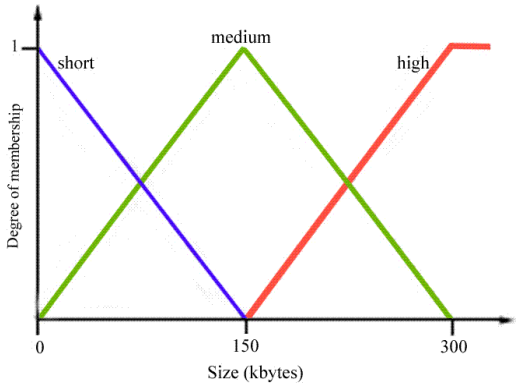


FIG. 5. Fuzzy sets corresponding to size



Fuzzy rules try to combine these parameters as they are connected in real world. Some of these rules are mentioned here.

Fuzzy Rules:

If (size of the incoming object > size of free space in durable cache)
{
If (ReferenceFrequency is High) and
If (AccessRecency is Low) and
If (Size is Short) then
ReplacementPriority is Low

If (ReferenceFrequency is Low) and
If (AccessRecency is High) and
If (Size is High) then
ReplacementPriority is High

If (ReferenceFrequency is Medium) and
If (AccessRecency is Medium) and
If (Size is Medium) then
ReplacementPriority is Medium
}

In fuzzy inference system, the number of rules has a direct effect on its time complexity. So, having fewer rules may result in a better system performance.

The Proposed Algorithm (FRA)

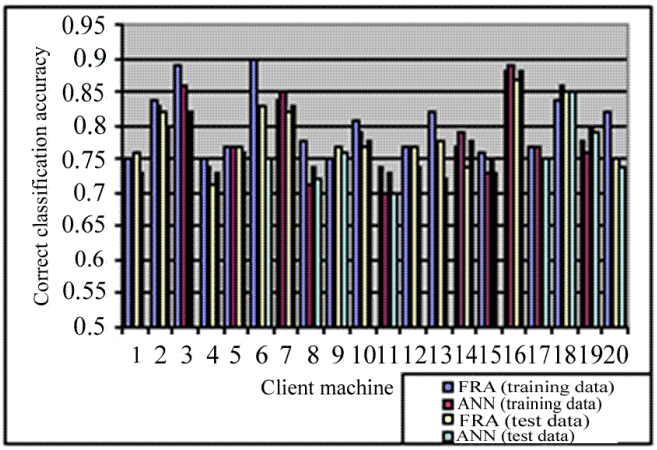
1. For each used page P in Durable cache, feed its reference frequency, access recency and size. Consider the output as replacement priority of the page P.
2. Swap out the page with highest replacement priority.
3. If more than one page is having the highest priority then the page which is older is replaced.

Performance Evaluation

To evaluate the performance of the proposed method, trace driven simulations is developed in Java with various types of workloads. In the simulations, we compare our fuzzy algorithm with the LFU, LRU, and GDS replacement algorithms, as they represent a reference in the framework of web caching.

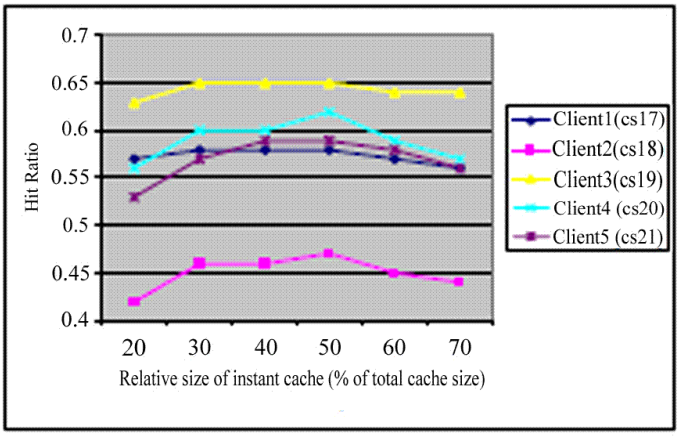
Two performance metrics, Hit Ratio (HR) and Latency Saving Ratio (LSR) are used to evaluate the efficiency of the replacement algorithms. The following figure shows comparison of the proposed method (FRA) and ANN for 20 clients in both training and test data. As we can see FRA produce god classification accuracy. Both training and test data in most clients compared to ANN.

FIG. 6. **Comparison of the correct classification accuracy for FRA and ANN**



In the proposed method, an obvious question would be the size of each cache. Many experiments were done to show the best size of each cache to ensure better performance. The simulation results of Hit Ratio of five clients with various sizes of instant cache are shown in figure 7.

Fig 7. **Hit Ratio for different instant cache sizes**

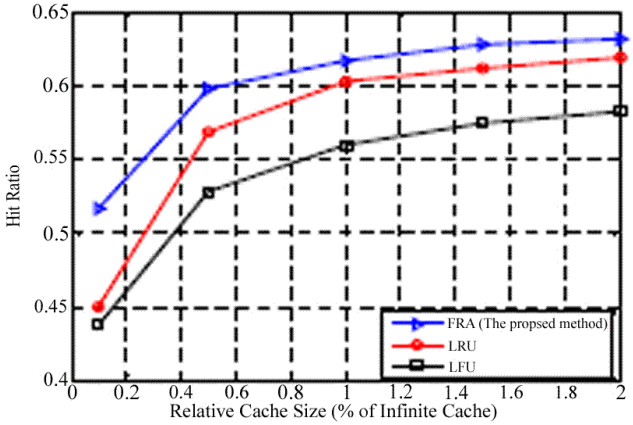


We can conclude that instant cache with size 40% and 50% of total cache size performed the best performance. Here the assumption is that the instant cache is 50% of the total cache size.

For each client, the maximum HR and LSR are calculated for a cache of infinite size. Then, the measures are calculated for a cache size 0.1%, 0.5%, 1%, 1.5% and 2% of the infinite cache size on the performance measures accordingly. The observed values of the measures are steady and close to maximum values after 2% of the infinite cache size in several policies.

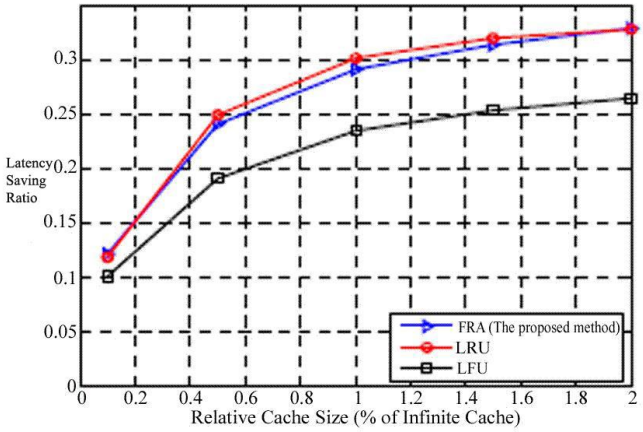
The performance of the proposed approach is compared to LRU and LFU policies that are the most common policies and form the basis for other web cache replacement algorithms [9]. Fig. 8, 9 show the comparisons of the average values of HR and LSR for twenty clients for the different policies with varying cache size. For the proposed method (FRA), HR and LSR include HR and LSR in both instant cache and durable cache.

FIG. 8. **Impact of cache size on Hit Ratio**



As we can observe that the proposed method (FRA) has superior performance for HR compared to other policies in all conditions. This is mainly due to the capability of the proposed method in storing ideal objects that are important or preferable to the user. On the other hand LSR was the same or slightly worse that LRU.

FIG. 9. **Impact of cache size on Latency Saving Ratio**



V. CONCLUSION AND FUTURE WORKS:

Replacement policies play a key role for the effectiveness of web caching. This study proposed a new approach by splitting web browser cache to two caches, instant and durable cache, on a client computer for storing the ideal web objects and removing the unwanted objects in the cache for more effective usage. This study also described the use of fuzzy logic to improve durable cache replacement decisions. The choice of the pages to be evicted from the cache is based on qualitative reasoning that takes into account the page characteristics.

The experimental results show that this approach has better performance compared to the most common policies. More over, the results of simulations have shown that the fuzzy algorithm achieves good performance even for small cache size. This means that the fuzzy approach allows a dramatic savings of the disk space and improves the performance of Browsers.

The future work will include more characteristics of pages such as freshness and consistency.

REFERENCES

[1]. Wessels, L.D.: Web Caching. O'Reilly, USA (2001)

[2]. Chen, H.T.: Pre-fetching and Re-fetching in Web Caching systems: Algorithms and Simulation. Master Thesis, TRENT UNIVESITY, Peterborough, Ontario, Canada (2008)

[3]. Mookerjee, V.S., Tan, Y.: Analysis of a Least Recently used Cache Management Policy for Web Browsers. Operations Research, Linthicum 50(2), 345–357 (2002)

[4]. Chen, T.: Obtaining the Optimal Cache Document Replacement Policy for the Caching System of an EC Website. European Journal of Operational Research 181(2), 828 (2007)

[5]. Koskela, T., Heikkonen, J., Kaski, K.: Web Cache Optimization with Nonlinear Model using Object Feature. Computer Networks Journal 43(6) (2003)

[6]. Ayani, R., Teo, Y.M., Ng, Y.S.: Cache pollution in Web Proxy Servers. In: International Parallel and Distributed Processing Symposium (IPDPS 2003), p. 248a. ipdps (2003)

[7]. Wong, A.K.Y.: Web Cache Replacement Policies: A Pragmatic Approach. IEEE Network magazine 20(1), 28–34 (2006)

[8]. Podlipnig, S., Böszörmenyi, L.: A Survey of Web Cache Replacement Strategies. ACM Computing Surveys 35(4), 374–398 (2003)

[9]. Cobb, J., ElAarag, H.: Web Proxy Cache Replacement Scheme based on Back-propagation Neural Network. Journal of System and Software (2007)

[10]. Farhan: Intelligent Web Caching Architecture, Master thesis, Faculty of Computer Science and Information System, UTM university, Johor, Malaysia (2007)

[11]. C. Aggarwal, J. L. Wolf, and P. S. Yu. Caching on the World Wide Web. IEEE Trans. On *Knowledge and Data Engineering*, 11(1):94-107, 1999.

[12]. S. Jin and A. Bestavros. GreedyDual* Web Caching Algorithm: Exploiting the Two Sources of Temporal Locality in Web Request Streams. *Computer Communications*, 2(24):174-183, 2000.

[13]. K. Psounis and B. Prabhakar. A Randomized Web-Cache Replacement Scheme. *In Proc. IEEE INFOCOM 2001*, pages 1407-1415, 2001.

[14]. A. Vakali. Evolutionary Techniques for Web Caching. *Distributed and Parallel Databases – An International Journal*, 11(1):93-116, 2002.

[15]. Acharjee, U.: Personalized and Artificial Intelligence Web Caching and Prefetching. Master thesis. University of Ottawa, Canada (2006)

[16]. Li, X.X., Huang, H., Liu, C.H.: The Application of an ANFIS and BP Neural Network Method in Vehicle Shift Decision. In: 12th IFToMM World Congress, Besançon France, M.C (2007)

[17]. Calzarossa, V.G.: A Fuzzy Algorithm for Web Caching. Simulation Series Journal 35(4), 630–636 (2003)

[18]. Krishnamurthy, B., Rexforrd, J.: Web Protocols and Practice: HTTP/1.1, Networking Protocols, Caching and Traffic Measurement. Addison-Wesley, Reading (2001)

[19]. Muñoz-Expósito, J.E., García-Galán, S., Ruiz-Reyes, N., Vera-Candeas, P.: Adaptive Network-based Fuzzy Inference System vs. Other Classification Algorithms for Warped LPC-based Speech/music Discrimination. Engineering Applications of Artificial Intelligence 20(6), 783–793 (2007)

[20]. Jang: ANFIS: Adaptive-network-based Fuzzy Inference System. IEEE Trans. Syst. Man Cybern. 23(3), 665 (1993)

[21]. Lee D., Choi J., Kim J.-H., Min S.L., Cho Y., Kim C.S., Noh S.H., On the Existence of a Spectrum of Policies That Subsumes the Least Recently Used (LRU) and Least Frequently Used (LFU) Policies, Proceedings ACM SIGMETRICS, 1999.

[22]. Lee D., Choi D., Kim J.-H., Noh S. H., Min S. L., Cho Y., Kim C. S., LRFU: A spectrum of policies that subsumes the least recently used and least frequently used policies, IEEE Trans. Computers, vol. 50, no. 12, 2001.

[23]. Wang Lie-Xin, A course in fuzzy systems and control, Prentice Hall, Paperback, Published August 1996.