Indian Journal of Science and Technology, Vol 9(36), DOI: 10.17485/ijst/2016/v9i36/102159, September 2016
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Intelligent Web Proxy Cache Replacement Algorithm Based on Adaptive Weight Ranking Policy via Dynamic Aging

Rashidah Funke Olanrewaju*, Dua'a Mahmoud Mohammad Al-Qudah, Amelia Wong Azman and MashkuriYaacob

Faculty of Engineering, International Islamic University Malaysia, Jalan Gombak 53100, Kuala Lumpur, Malaysia;
frashidah@iium.edu.my, dm_q66@yahoo.com, amy@iium.edu.my, mashkuri@iium.edu.my

Abstract

Rapid growth in network services and vast usage of Internet worldwide have led to an increase in network traffic and created bottleneck over the internet. Such traffic results in an increase of access time of web documents, server response latency, reduced network bandwidth and slow response time for popular websites. Web cache is an essential optimization technique used to reduce response time and improve performance. However due to its limited size and cost of cache comparable to other storage devices, cache replacement algorithm is used to determine and evict page when the cache is full to create space for new pages. Several algorithms had been introduced and their performances are important in producing high speed web caching. However, their performances are not well optimized. This work proposes a hybrid method that optimize cache replacement algorithm using Naïve Bayes (NB) based approach. Naïve Bayes is an intelligent method that depends on Bayes' probability theory integrated with Adaptive Weight Ranking Policy (AWRP) via dynamic aging factor to improve the response time and network performance. From the pseudocode of the proposed algorithm, it is observed that the complexity of the algorithm is O(n)which is linear, hence the response time is considergood. Performance evaluations based on hit rate and byte hit rate for new the method over conventional methods with real data will be conducted for validation and verification.

Keywords: AWRP, Dynamic Aging, Naïve Bayes, Proxy Cache, Replacement Algorithms

1. Introduction

The large usage of internet leads to an increase in network traffic and page access latency[1]. By having web cache, web objects can be kept closer to the client. A web proxy cache saves web object that can be revisited in the near future closer to the user which leads to a decrease of the number of web object requested from the server, hence, reducing the volume of data that is transmitted over the network as well as the delay while accessing the web page[2,3]. Consequently, the overall performance of web systems can be improved. These web objects are commonly placed in three locations; 1) browser caching, 2) proxy caching and; 3) server side proxy caching[4].

There are three significant factors that play pivotal roles in cache performance. They are the cache replacement algorithms, cache consistency and cache algorithm (passive or active). Cache replacement algorithms, which have the significant effect in the web caching[5], can be classified into five categories, namely: frequency based, recency based, frequency/recency based, function based and randomized based[6]. However, most of the conventional algorithms are not suitable for web caching since the algorithms take into account one or two features while ignoring the others. The algorithms also suffer from many problems like cache pollution in which the cache saved objects that are not requested frequently. There are many algorithms that combine two or more features to

*Author for correspondence

enhance the performance of the replacement algorithms. However, the selection of features is not a straightforward task because an important factor in one environment may be not important in another environment[2].

Adaptive Weight Ranking Policy (AWRP) attempts to give a weight to each page depending on three factors: recency, frequency and total number of access to be made to improve performance of cache and overcome traditional algorithms problem such as least recently used LRU, First Input First Output FIFO and Clock Adaptive Replacement CAR[7]. Intelligent replacement algorithms were proposed to overcome problems of conventional replacement algorithms and to develop the web cache performance. They predict which web objects will be revisited in the future. Machine learning algorithms such as Fuzzy, Artificial Neural Network, Adaptive Neuro Fuzzy, Naïve Bayes, as well as Support Vector Machine have been used for prediction. The next step is to combine the prediction with one or more of the conventional algorithms to improve their performance mainly in terms of hit ratio as well as byte hit ratio[8].

Naïve Bayes classifier is a simple supervised machine learning algorithm which depends on Bayes' probability theory. It can deal with any number of attributes and classes. Moreover, Naïve Bayes technique is considered the fastest machine learning technique and it is very easy to construct. In addition, its performance is very good in a lot of applications and applied well in many domains[2].

Feature selection is a method of extracting subsets of features from feature space. It plays a major role in refining and enhancing the performance of machine learning techniques through reducing the effect of noise and irrelevant features. Moreover, it reduces the computational time and storage required to store features. Feature selection methods are classified in three groups, namely: Filter, Wrapper and embedded methods[10].

This paper proposed an intelligent replacement algorithm based on Naïve Bayes classifier to predict the web objects that will be revisited in the future or otherwise. We propose an embedded feature selection method to extract the best subset feature of Naïve Bayes and later integrate Naïve Bayes classifier with Adaptive Weight Ranking Policy via Dynamic Aging (AWRP-DA) to improve the performance of web systems concerning Hit Ratio (HR) and Byte Hit Ratio (BHR).

The remaining part of the paper is ordered as follow: section two presents related works; section three presents our proposed model, section four presents performance evaluation, section five presents our conclusion and future work.

2. Related Work

2.1 Cache Replacement Algorithms

Weighting Replacement Policy (WRP) ranks the objects in the cache according to three factors: recency, frequency, and reference rate of each object by giving each object a weight as defined in the following equation (1);

$$Wi = Li / (Fi * \Delta Ti) \ldots..(1)$$

where Wi refers to weight, Li refers to recency, Fi refers to object frequency and i is reference rate of each object[11]. Weight is used to determine the evict object when the cache is full by evicting objects starting with the objects with the highest weight value. However, WPR adds time and space overhead to the system due to weight calculation and parameters saving. In[12], authors modified WRP to enhance the hit ratio by adding a special buffer between cache memory and main memory and giving the evicted object another chance. However, this modification again adds storage overhead to the system.

An adaptive replacement algorithm developed by Swain [7] ranks objects in the cache based on weight which is defined in equation (2);

$$Wi= Fi / (N – Ri) \ldots.. (2)$$

where Fi is a frequency index, Ri is recency index and N is the total number of access to be made. It behaves like LRU and LFU algorithms. However, it performs better than LRU, LFU and CAR based on hit ratio, but it assumes all blocks have equality size and it adds time and space overhead.

2.2 Web Cache Replacement Algorithms

In [13], a comparison between eight of the most well-known function based strategies (Greedy Dual (GD)-Size, Greedy Dual-Size with Frequency (GDSF), Greedy Dual* (GD*), Taylor Series Prediction (TSP), MIX, M-Metric, Least Normalized Cache Replacement on the Web (LNC-R-W3), and Least Unified Value (LUV)) using Hit Rate (HR) and Byte Hit Rate (BHR) was made to evaluate the performance of different web cache replacement algorithms. Authors concluded that LRU and GDSF are not well enough nowadays. Moreover, M-Metric or MIX do

well in terms of HR and BHR. However, Taylor Series Prediction (TSP) is recommended for systems that have limitations on CPU usage and disk access. In addition[6], analyzed twenty-seven proxy cache replacement algorithms based on HR, BHR and object removal rate. The author concluded that most of the algorithms are simple and utilizes a relative low CPU and space overhead. The PSS, CSS, M-Metric, MIX and GDSF algorithms give system administrators greater control over their proxy servers. In[3] provided a comparison among different cache replacement algorithms. In addition, the authors modified LFU algorithm based on frequency, popularity and recency of objects using web usage mining. These modifications have proven to produce better results when compared to GDS and LFU algorithms.

In[14] developed the Weighted-Rank Cache Replacement Policy (WRCP) which depends on access frequency, aging and mean access gap ratios and other functions such as size and cost of retrieval parameters to calculate weighting of pages. In[1] proposed two new approaches which called LFUIR and WRPIR. The methods depend on sorting web sites based on internal request factor integrated with LFU and WRP. They concluded that LFUIR and WRPIR occupy cache memory with most benefit pages.

In[4] proposed a web cache replacement algorithm based on Fuzzy Inference System (FIS). FIS ranks web objects depending on frequency, latency and byte-sent and gives each web object value between 0 and 1, so lower web object priority will be evicted first. However, the proposed approach did not have learning ability and added extra computation overhead and implementation complexity. In[5], browser cache was split into two caches namely instant caches managed using LRU algorithm and durable cache that is using fuzzy logic to classify pages into cacheable or uncatchable. Uncatchable page becomes a candidate to be evited when cache is full. The results showed that there was an improvement in terms of HR and LSR over LRU and LFU algorithms. Moreover [15] proposed a multi agent system using fuzzy logic to make an intelligent decision about the clean-up process. They combined LRU and LFU in the child side while LFU, LRU and size in the parent side. The proposed scheme achieved improvement in the terms of HR and BHR in both child and parent sides. However, cache cleaning tasks performed independently and there were no learning algorithms for cache cleaner agent.

In[16] developed an Intelligent Client-side Web Caching Scheme (ICWCS) which depended on splitting client cache side into two caches: short term cache which is managed using LRU algorithm, and long term cache which is managed using neuro-fuzzy system. They improved the performance of web caching in terms of HR and BHR over LRU and LFU algorithms. Be that as it may, the execution in regards to byte hit proportion was not sufficiently commendable in light of the fact that they didn't consider the expense and size of the anticipated items in cache substitution process. Additionally, the preparation procedure required quite a while and additional computational overheads.

In[2], NB-GDS, NB-LRU and NB-DA approaches were developed which combined between Greedy Dual Size (GDS), LRU and DA conventional algorithms and Naïve Bayes classifier. These approaches were done in two stages: offline stage to train NB classifier and predict either web object will be re-visited or otherwise, and online stage which corporate between NB and conventional algorithms to evict pages when proxy cache is full. They performed better than BPNN and ANFIS in terms of HR and BHR, but NB adds additional computational overhead which is needed for target output preparation. In [16] proposed NB-SIZE and C4.5-Hybrid which is an integration between conventional SIZE, Hybrid algorithms, Naïve Bayes and decision tree (C4.5) to significantly improve pure HR, BHR and latency. An integration of LFU-DA and machine learning technique such as SVM, NB and C4.5 designed by[17] to enhance their performance in terms of HR and BHR. However, features are selected manually in training machine learning techniques.

In[18] designed new approaches RFT-LRU and RFT-GDSF that used Random Forest Tree classifier (RFT) to predict the classes of objects that will be revisited or not in future then combined with LRU and GDSF. Beside improvement in performance in terms of HR and BHR, RFT achieved better true positive rate and performance much better than ANFIS. Three phases' system: pre- processing phase, prediction phase that used Tree Augmented Naïve Bayes classifier (TANB) to predict that page revisited in future or otherwise and the last phase integrated TANB with LRU and GDSF to improve proxy web cache in terms of HR and BHR respectively was developed by[19]. Experimental results showed that the system had much better precision compared with other classifiers. However, features that were used in training TANB were chosen manually.

In[20], used automated method to extract best subset of features to improve the prediction method in intelli-

gent systems which were used in the training phase by J48 classifier and NB classifier. Automated prediction scheme caused reduction in the number of features that were selected and a reduction dimensionality of dataset. Moreover, reduction in computation time is achieved, and improved accuracy of NB, C4.5 using the wrapper method compared to manual selection method.

3. Proposed Model

Naïve Bayes Adaptive Weight Ranking Policy via Dynamic Aging (NB-AWRP-DA) incorporates learning capability of NB with AWRP-DA to enhance the performance of web proxy cache in terms of HR and BHR. NB- AWRP -DA composes mainly of three stages namely: raw data collection, pre-processing, training stage, and testing stage.

3.1 Raw Data Collection and Pre-processing

3.1.1 Raw Data Collection

Whenever a user requests a page, the proxy server stores the event into a log file. Log files have information about users' behaviors and requests. Proxy log files can be gathered from different proxies that placed inside organization or universities. Availability of log files was one of the most important motivations to corporate machine learning with conventional replacement algorithms. Most researchers depend on proxy log files that are provided by IRCache network. IRCache trace files are gathered from different proxy servers that are placed around the United State. IRCache trace files provide enough information that can help to indicate cacheable web object or otherwise [6]. Standard features exist in most trace files namely: time elapsed, timestamp, object size, client address, log tag with HTTP script, URL, user identification, request method, content type and hierarchy of data and hostname.

3.1.2 Pre-processing

Proxy log files have many features about user behavior. Some of them do not have impact on proxy cache algorithm performance. In this phase, processing of log files is required to remove irrelevant records, records that have unsuccessful status. We consider the success status that have 200 code and uncatchable request like request that has "?" inside URL. For simplify, each URL will be replaced with unique random integer and will be used it to define object type as follow; HTML type will be defined as 1, image type will be defined as 2, audio will be defined as 3, video will be defined as 4, application will be defined as 5 and others types will be defined as zero. After that, log files will contain the main fields that have impact from which the useful information can be extracted: URL Id, time stamp, elapsed time, object size, type of Web object [20].

3.1.3 Training

Required features will be extracted from prepared log files which have an impact on web proxy performance. They will be used to learn machine learning algorithm. Features that will be used are: URL, time stamp, total elapsed time, object size, frequently, swl-frequently, recency, the type of Web object, time spent and mean. These features will be calculated as described in [20] and internal request frequency as described in[1].

After that, wrapper feature selection method, Incremental Wrapper Feature Subset Selection (IWSS), will be embedded with NB to extract the best sub set of features that has a great impact on classifier performance. IWSS-embedded-NB uses NB classifier inside IWSS feature selection method to speed and enhance IWSS performance. IWSS will be applied to features as default setup defined in WEKA software. More detail about IWSS-embedded-NB discussed at [21].

Finally, an extracted subset of features will be prepared in input and output pattern. The pattern form is < x1, x2,……, xn, y> where x1,…..,xn are features input and y is target output that will be set to 1 if the object visited another time within SWL else it will be set to 0. SWL is set to 30 minute [2]. After that, discretizing features by applying MDL which suggested by [22]. The dataset is divided as 70% for training and 30% for testing. NB classifier predicts that the object will be requested in future or not.

3.1.4 Testing

Intelligent replacement algorithm will be done by integrated training intelligent system with AWRP.AWRP ranking web objects by calculating the weight for each of them depending on three factors as shown in equation 3.

$$Wi = Fi / (N - Ri) \quad (3)$$

where Ri represent recency counter of object i, Fi represents frequency counter of object i, and N is a total number of access to be made. If new object k is stored in the buffer then object's parameters must be set as follow: Rk = clock which means that object k recently used, Fk = 1 which means that object k is used once and Wk = 0. Now, if object j requested and it is found inside buffer a hit occur. Thus, object parameters must be updated in following way:

Fi = Fi+1 for every i.

Ri= corresponding clock access value.

If requested object is not inside the buffer a miss occur. Fetching object to cache is required. If the cache is full then the object with smallest weighting value will be evicted until sufficient space of new object. Thus, evaluation for each object will be done as follow:

Wi evaluated for each block I for N ≠Ri.

Removing page i with lowest weight index.

The weighting value for each object inside the buffer is updated only when misses occur that reduces the overhead in the system. Intelligent system NB integrated with AWRP as follows;

Pri= Apply classifier (common features).

Wi= Fi / (N – Ri)

```
For each object g requested by user
Initialize L =0
Begin
    If g in cache
        Begin
            Cache hit occur
            Update g parameters
            Fi = Fi+1
            Ri= corresponding clock access value
            Pri = apply_intelligent_classifier (common features).
        End
  Else
        Begin
      Cache miss occur
            For each object in cache
            Wi= Fi/ (N-Ri)
            Fi = Wi *Pri +L
     Fetch object from original server
      While no enough space in cache
            Begin
            L = min (F (q)) for each q in cache
            Evict q such that F (q) = L
            End
        End
End
```

Figure 1. NB-AWRPDA pseudo code.

Fi= (Wi*Pri) +l

l is a dynamic aging factor. Initially L is set to 0, but upon the removal of an object i, L is set to Fi (weight of last removed object). Pri represent probability of object to revisited or otherwise and Fi is final weight value for object i.

Figure 1 presents a pseudo code of the proposed algorithm and Figure 2 presents a framework of NB-WRPDA approach.

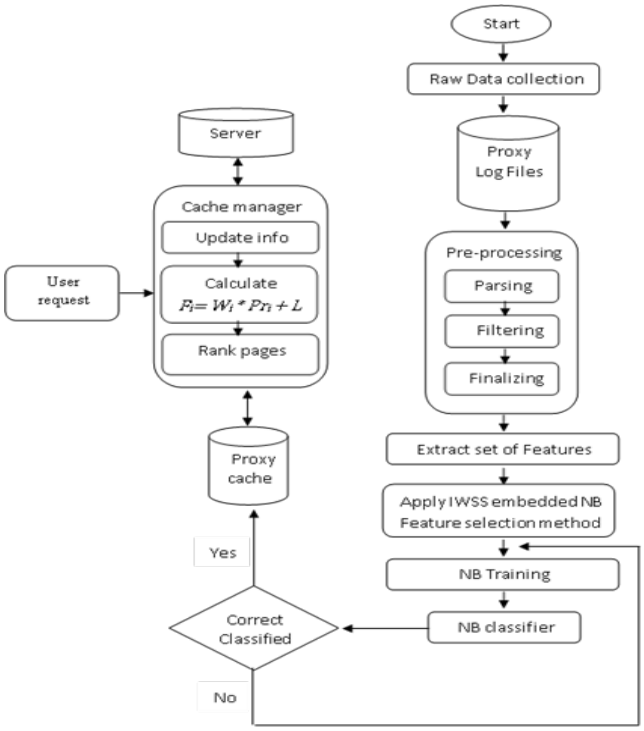

Figure 2. Framework of NB-AWRPDA approach.

4. Performance Evaluation

4.1 Classifier Evaluation

Classifier performance will be evaluated by a set of measures. Most important measures are: Correct Classification Rate (CCR) defined in (4), True Positive Rate (TPR) defined in (5) and True Negative Rate (TNR) defined in (6). Confusion matrix is shown in Table 1[16].

Table 1. Confusion matrix

	Predicted positive	Predicted negative
Actual positive	True Positive (TP)	False negative (FN)
Actual negative	False Positive (FP)	True negative (TN)

$$CCR = (TP+TN)/(TP+FP+FN+TN)\% \quad (4)$$

$$TPR = TP/(TP+FN)\% \quad (5)$$

$$TNR = TP/(TN+FP)\% \quad (6)$$

4.2 Intelligent Algorithm Performance Measures

Hit Ratio (HR) and Byte Hit Ratio (BHR) matrices are two of the most important measures used to evaluate web proxy caching policy. HR and BHR defined in (7) and (8) respectively where N is total number of requests, $\delta i = 1$ if request i in the cache else $\delta i = 0$, and bi is the size of ith request. However, HR and BHR works in opposite way[16].

$$HR = \sum \delta_i / N \quad (7)$$

$$BHR = \sum (\delta_i * b_i) / \sum b_i \quad (8)$$

5. Conclusion and Future Work

Web caching has a significant role in replying web items to end users. However, cache size is limited, which calls for replacement once the cache is full. Furthermore, traditional web cache replacement algorithms such as LFU, LRU, and GDS policies suffer from caching pollution. This study proposed intelligent Web proxy caching approach named NB-AWRPDA, to improve performance of AWRP in terms of HR and BHR. Initially, NB trained from proxy log files to predict if object will be requested in future or otherwise. After that, classifier integrated with AWRP algorithms then with aging factor to enhance performance and solve cache pollution problem.

In future, other intelligent classifiers can be trained and integrated with AWRPDA to improve performance in terms of HR and BHR. Also, others feature selection method can be used for enhancing the performance of intelligent classifier.

6. Acknowledgement

This project is funded by the Ministry of Higher Education Malaysia (November, 2015 to October, 2017):Program for Promoting Fundamental Research Grant Scheme (Project No. FRGS15-254- 0495).

7. References

1. Sarhan A, Elmogy AM, Ali SM. New web cache replacement approaches based on internal requests factor. 9th International Conference on Computer Engineering & Systems, Egypt; 2014.
2. Ali W, Shamsuddin SM, Ismail AS. Intelligent Naïve Bayes-based approaches for web proxy caching, Knowledge-Based Systems. 2012; 31:162–75.
3. Sathiyamoorthi V, Bhaskaran VM. Web caching through modified cache replacement algorithm. International Conference on Recent Trends in Information Technology, Chennai: Tamil Nadu; 2012.
4. KumarSaha A, Dev PP, Kar M, Rudrapal D. An optimization technique of web caching using fuzzy inference system. International Journal of Computer Applications. 2012; 43(17):20–3.
5. Muralidhar K, Geethanjali DN. Improving the performance of the browsers using fuzzy logic. International Journal of Engineering Research and Technology. 2012 Aug; 3(1):1–8.
6. ElAarag H. A quantitative study of web cache replacement strategies using simulation. Web Proxy Cache Replacement Strategies, Springer Briefs in Computer Science; 2013. p. 17–60.
7. Swain D. AWRP: Adaptive Weight Ranking Policy for improving cache performance. Journal Of Computing. 2011; 3(2):209–14.
8. Ali W, Shamsuddin SM, Ismail AS. A survey of Web caching and prefetching. International Journal of Advance Soft Computing Application. 2011 Mar; 3(1):18–44.
9. Chandrashekar G, Sahin F. A survey on feature selection methods. Computers and Electrical Engineering. 2014; 40:16–28.
10. Samiee K. A replacement algorithm based on weighting and ranking cache objects, International Journal of Hybrid Information Technology. 2009 Apr; 2(2):93–104.
11. Bhalgama S, Parmar SS. A novel adaptive cache replacement policy using weighting and ranking parameter. American International Journal of Contemporary Scientific Research. 2015; 2(1):7–16.
12. ElAarag H, Romano S. Comparison of function based web proxy cache replacement strategies. International Symposium on Performance Evaluation of Computer and Telecommunication Systems. 2009; 41(16):252–59.
13. Ponnusamy S, Karthikeyan E. Cache optimization on hot-point proxy caching using weighted-rank cache replacemnt policy. ETRI Journal. 2013; 35(4):687–96.
14. Sirour HAN, Hamad YAM, Eisa AA. An agent-based proxy cache cleanup model using fuzzy logic. International Conference on Computing, Electrical and Electronics Engineering, Africa; 2013.
15. Kumar PV, Reddy VR. Novel web proxy cache replacement algorithms using machine learning techniques for performance enhancement. International Journal of Engineering Sciences and Research Technology. 2014 Jan; 3(1):339–46.

16. Ali W, Shamsuddin SM. Intelligent dynamic aging approaches in web proxy cache replacement. Journal of Intelligent Learning Systems and Applications. 2015; 7:117–27.
17. Benadit JP, Francis SF, Nadhiya M. Enhancement of web proxy caching using random forest machine learning technique. International Journal of Computer Science Issues. 2014; 11(3):83–91.
18. Benadit PJ, Francis FS, Muruganantham U. Improving the performance of a proxy cache using tree augmented naive bayes classifier. Procedia Computer Science Proceedings of the International Conference on Information and Communication Technologies, ICICT 2014, 2014 Dec 3–5, Bolgatty Palace & Island Resort, Kochi: India. 2015; 46:184–93.
19. Abdalla A, Sulaiman S, Ali W. Intelligent web objects prediction approach in web proxy cache using supervised machine learning and feature selection. International Journal of Advances in Soft Computing and Its Applications. 2015 Nov; 7(3):146–64.
20. Bermejo P, Gámez JA, Puerta JM. Speeding up incremental wrapper feature subset selection with Naive Bayes classifier. Knowledge-Based Systems. 2014; 55:140–7.
21. Irani KB. Multi-interval discretization of continuous-valued attributes for classification learning. Machine Learning; 1993. p.1022–7.