Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 48 (2015) 304 – 312

Procedia Computer Science

International Conference on Intelligent Computing, Communication & Convergence

(ICCC-2015)

Conference Organized by Interscience Institute of Management and Technology,

Bhubaneswar, Odisha, India

Improving the Performance of a Proxy Cache Using Very Fast Decision Tree Classifier

Julian Benadit.P[a,*], Sagayaraj Francis.F[a]

[a]*Department of CSE, Pondicherry Engineering College, Pondicherry, 605014, INDIA*

Abstract

In this paper, we improved the performance of Web proxy cache replacement policies such as LRU and GDSF by adapting a Very Fast Decision Tree learning technique. In the first part, a sliding window method integrated with Very Fast Decision tree classifier (VFDT) to classify the web log data and predict the classes of web objects to be revisited again in future or not. In the second part, a Very Fast Decision Tree classifier is incorporated with proxy caching policies to form novel approaches known as VFDT-LRU and VFDT-GDSF. This proposed approach improves the performances of LRU and GDSF in terms of hit and byte hit ratio respectively.

Peer-review under responsibility of scientific committee of International Conference on Computer, Communication and Convergence (ICCC 2015)

Keywords: Proxy Caching;Cache replacement;Classification;Very Fast Decision Tree classifier

* Corresponding author. Tel.:+ 919629321243.
E-mail address: benaditjulian@gmail.com

Peer-review under responsibility of scientific committee of International Conference on Computer, Communication and Convergence (ICCC 2015)
doi:10.1016/j.procs.2015.04.186

1. Introduction

As the World Wide Web and users are explicating at a very rapid rate, the performance of World Wide Web systems becomes rapidly high. Web caching and prefetching is the one of the best methods for improving the performance of the proxy server. The basic idea in web caching is to retain the most popular web log data in a proxy cache, such that the performance of web proxy cache would have improved, when it is accessed from the user. The main idea behind the web caching concept is a web cache replacement, algorithm and its key parameters in the algorithms.

To ameliorate the functioning of a proxy cache, a lot of research work has been done in their cache replacement policies.Table 1.Presents a summary of the few existing Cache Replacement Policies(CRP)[7,8].Most of these replacement algorithms consider only certain key factors and assign a key value based on the priority for each web document which stored in the cache. But, it is difficult to have a better cache replacement policy that performs well in all situations, because each replacement policy has a different key parameter to optimize web resources. Moreover, various elements can act upon the cache replacement policy to receive a better replacement decision and it is not an easy task for because one parameter is more significant than the other one. Due to this restriction, there is a need for an active method which intelligently manages the web proxy cache by satisfying the objectives of web caching requirement. So this platform promotes, for the integration of the machine learning methods in the web caching replacement algorithms.

In our previous surveys, the intelligent techniques have been applied in web caching algorithm. By extensive use of this prediction pattern, the caching algorithms become more efficacious and adapted for the use of web cache environment, other than the classic web caching algorithms which are already in practice. Moreover, there are multiple users who are fond of the access of web cache algorithms, but it is very necessary to prove a prediction model, which are upgraded often so that web objects can be revisited in the future on a proper standard. In this paper, we proposed the Very Fast Decision Tree Classifier (VFDT)[1,12] to train the web object based on the recurrent sliding window method for their systematic classification, so that the web objects can be predicted in the future or not. Finally, it is incorporated with traditional caching algorithm called VFDT-LRU and VFDT-GDSF to improve its web caching performance on a better channel.

Table 1. Cache Replacement Policies.

CRP	Key Parameter	Cache Replacement Technique
LFU	Number of References.	The least frequently accessed first.
LRU	Time Since last access.	The least recently accessed first.
GDS	Document Size s_d.	Least value first according to value $p_i = C_d / S_d + L$.
	Document cost C_d.	
	An Inflation Value L.	
GDSF	Document Size s_d.	Least value first according to value $p_i = (C_d \times F_d / S_d)^{\beta} + L$.
	Document cost C_d.	
	Number of non-aged references f_d time since last access.	
	Temporal correlation measure β.	
	An Inflation Value L.	

The structures of this paper with the features are processed below. In section 2, we present the related work with an Existing machine learning methods towards web caching policies. The Section 3, proposes the brief introduction of Very Fast Decision Tree classifier model. In Section 4 we introduce the proposed novel web proxy caching approach integrates with the cache replacement algorithm. Experimental results and Performance Evaluations are presented in Section 5 and Section 6. The Section 7 concludes our paper with revealed results.

2. Related Work

Web Caching holds an important role in improving the performance of the web proxy cache. The basic idea of web caching is known for its Replacement Policy, which calculates the most popular web object by storing it in the proxy cache, hence that the popular web documents can be put back with the unpopular ones. Most of the common replacement algorithm assigns new key value computed by factors such as size, frequency and cost. Using this key value, we would rate the top most web documents on their corresponding key-values.

In preceding Paper, it exploits supervised learning methods to cope with the matter[2,4,6,11]. Most of the recent surveys use Back Propagation Neural Network (BPNN) and Decision Tree (ID3) in world-wide caching which is presented in Table 2. Though BPNN training might consume a wide amount of time and need further process overheads. Moreover, the ID3 Decision Tree result in less accuracy in classifying the large web data and it consumes more memory space, So in this paper, we have attempted to increase the performance of web cache replacement, strategies by integrating Very Fast Decision Tree classifier called VFDT[12].

Table 2. Summary of Existing Machine Learning Methods towards web caching policies.

Intelligent methods	Key Parameter	Eviction
ID3[11]	A model based on decision trees consist of a series of simple decision rules, often presented in the form a graph.	Not good for predicting the values of a continuous class attribute. Low prediction accuracy, high variance.
BPNN[6]	Back-propagation uses a supervised learning method at each layer to minimize the error between the layer's response and the actual data. The error at each stage of the hidden layer is an average of the evaluated error.	In the training phase getting of target output is not clear. Storing web object is more complicated.
NNPCR[3]	ANN has been used for making cache replacement decision .An object is selected for replacement based on the rating returned by ANN.	The objects with the same class are removed without any precedence between these objects. Training and Testing should be for longer periods of uptime to achieve high performance rate

In conclusion, we achieved a large scale evaluation compared with other classifier like Back Propagation Neural Network, and Decision Tree, Neural Network Proxy cache Replacement (NNPCR) in term, classification accuracy on different log data sets we collected and the proposed method has improved the performance of the proxy cache in terms of hit and byte hit ratio.

3. Very Fast Decision Tree Model

Domingos and Hulten[12] have developed a decision tree method which is referred to as Very Fast Decision Tree. It is based on the Hoeffding tree, a decision tree learning method. It is one of the most effective and widely used classification methods. It splits the tree using the current best attribute taking into consideration that the number of examples used satisfies the Hoeffding bound. This technique has the property that its output is asymptotically nearly identical to that of main stream learner. VFDT is used to mining the stream of web page requests from these log file which is collected from IRcache.The key idea of using the VFDT[1,12] is to predicts the data as accurately as possible and which host and pages will be requested in the near future, given recent request. We applied decision tree learning to this problem in the following manner. The procedure for VFDT is shown below and the steps involved in it.

Table 1: The VFDT algorithm

Inputs : S is a Web log of example
X is a set of symbolic attributes,
G(.) is a split evaluation function
δ is one minus the desired probability of choosing
the correct attribute at any given node,
τ is a user –supplied tie threshold ,
n_{min} is the # examples between checks for growth
Output :HT is decision tree.

Procedure VFDT (S, X, G, δ, τ)

- Let HT be a tree with a single leaf l_1 (the root)
- Let $X_1 = X \cup \{X_\phi\}$
- Let $\overline{G}_1(X_\phi)$ be the $\overline{G}_1$ obtained by predicting the most frequent class in S
- For each class y_k
- For each value x_{ij} of each attribute $X_i \in X$
- Let $n_{ijk}(l_1) = 0$
- For each example (x,y) in S
- Sort (x,y) into a leaf using HT
- For each x_{ij} in x such that $X_i \in X$
- Increment $n_{ijy}(l)$.
- Label l with the majority class among the examples seen so far at l
- Let n_l be the number of examples seen at l
- If the examples seen so far at l are not all of the same class and $n_l \bmod n_{min}$ is 0, then
- Compute $\overline{G_l}(X_i)$ for each attribute $X_i \in X_l - \{X_\phi\}$ using the counts $n_{ijk}(l)$
- Let X_a be the attribute with highest $\overline{G_l}$.
- Let X_b be the attribute with second - highest $\overline{G_l}$.
- Compute $\in$ using Equation 1
- Let $\Delta\overline{G}_l = \overline{G}_l(X_a) - \overline{G}_l(X_b)$
- If ($(\Delta\overline{G_l} > \in)$ or $(\Delta\overline{G_l} <= \in < \tau)$) and $X_a \neq X_\phi$, then
- Replace l by an internal node that splits on X_a
- For each branch of the split
- Add a new leaf l_m, and let $X_m = X - \{X_a\}$.
- Let $\overline{G_m}(X_\phi)$ be the $\overline{G}$ obtained by predicting the most frequent class at l_m
- For each class y_k and each value of each attribute $X_i \in X_m - \{X_\phi\}$.
- Let $n_{ijk}(l_m) = 0$.
- Return HT.

4. Proposed Novel Web Proxy Caching Approach

The proposed method will present a working flow see Fig. 2. For novel web proxy caching[2,4,5] based on the proposed machine learning technique. In our proposed work we use the Very Fast Decision Tree learner for classifying the datasets that can be revisited again or not in the future[4]. The mining steps consist of two different phases for classifying the datasets in the first phase, we pre-process to remove the irrelevant information in the proxy log data sets, a different technique is applied at the preprocessing stage such as data cleaning, data filtering and data integration. Once this task has been accomplished, the proxy sets have been trained in the second phase by the classifier VFDT, which predicts whether the web objects can be revisited once again in the future or not. In the third phase, the predicted web object[5] has been integrated to the traditional web proxy caching algorithm like LRU and GDSF for replacement strategies.

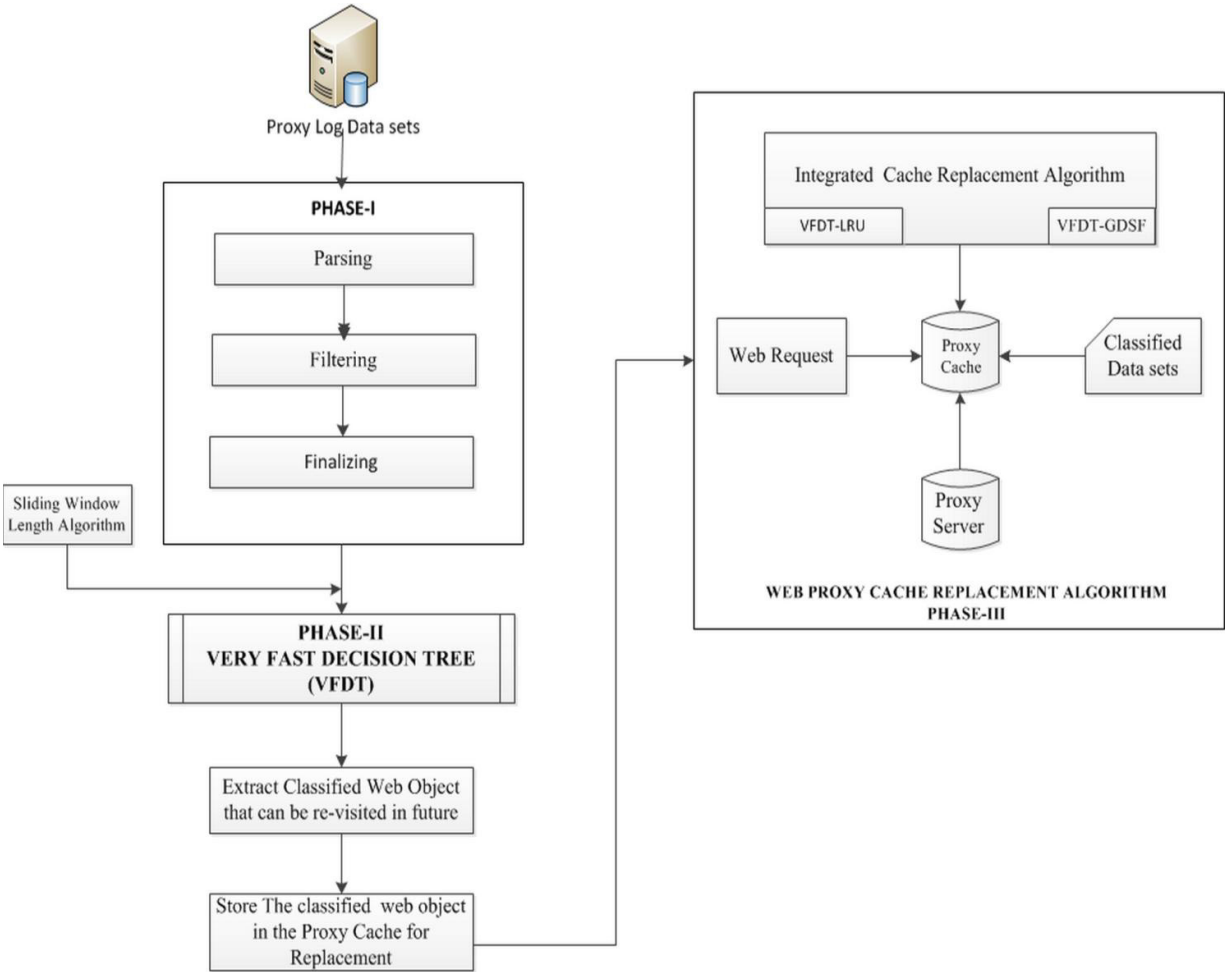

Fig. 2. Working Flow of Web Proxy Caching Approach based On Very Fast Decision Tree Classifier.

4.1. Very Fast Decision Tree Classifier -Input/Output System Based On Sliding Window Mechanism

The input parameters to Very Fast Decision Tree in the Table 3. are labelled as $\{ R_i , F_i , RT_i , Size_i \}$ and the predicted output as $Target_O$. Frequency (F_i) and Recency (R_i) for the objects are estimated based on the sliding window mechanism[5]. The Sliding window method[3,6] is used to obtain the Recency value based on the time before and after the request is made. Otherwise, its recency will have the maximum value among (SWL_i) and therefore the Recency of web object is calculated as follows in Equation 3.

$$\text{Recency} = \begin{cases} \max(SWL_i, \Delta T_i) & \text{if } obj_i \text{ requested before} \\ SWL_i & \text{if } obj_i \text{ request for the first time} \end{cases} \quad (3)$$

Frequency of object, Obj_i is incremented by 1 with respect to an previous frequency value, if the request for Obj_i is within the time interval, or within the boundary of backward-looking SWL, otherwise the frequency value

will re-initialize to 1 see in Equation 4. Target output ($Target_o$) will set to 1, if the object (Obj_i) is re-visited again with the forward sliding window; else $Target_o$ will be 0.

$$Frequency = \begin{cases} F_i + 1 & \text{if } \Delta T_i \leq SWL_i \\ Max\left[\frac{frequency}{\frac{\Delta T_i}{SWL_i}}, 1\right] & \text{if } obj_i \text{ beyond } SWL_i \end{cases} \quad (4)$$

In, Table 3. Presented below uses the various parameters for training the log data sets using the sliding window method. This method trains the data sets using sliding window length, Frequency and Recency of the web object. After the training data sets are prepared, these data sets are then integrated with VFDT classifier. The VFDT classifier takes the input parameters and classifies the data sets. After the data sets are classified, it has been used for web proxy caching algorithm for replacement.

Table 3. Input Parameters for VFDT and Sliding Window Method.

VFDT Parameters	Meaning
R_i	Recency of web object.
F_i	Frequency of web object.
RT_i	Retrieval Time of web object.
$Size_i$	Size of web object.
Recurrent Sliding window Parameters	**Meaning**
Obj_i	Requested web object.
ΔT_i	Time Since Obj_i was last requested.
F_i	Frequency of Obj_i within sliding window.
SWL_i	Sliding Window Length.
$Target_o$	Target output.

4.2. Very Fast Decision Tree -Greedy Dual Size Frequency (VFDT-GDSF)

Very Fast decision Tree learning method integrates with Greedy Dual Size Frequency[3,7,8] to improve the performance of Byte Hit Ratio. In this method, the VFDT uses as inputs the recency and frequency of the web object based on the sliding window mechanism, and as well as Retrieval time, the size of the Web object, and the classifier produces a target output in order to indicate whether the web object can visited in future or not see in Equation 5. Later, this re-visited data can be classified as cacheable data in order to increase the performance of a replacement algorithm in terms of the cache hit ratio.

$$K_i = F_d \times C_d / S_d + L + Target_o \quad (5)$$

4.3. Very Fast Decision Tree-Least Recently Used (VFDT-LRU)

Least Recently Used is the most common algorithm among all the caching algorithms[5,7]. But, this algorithm suffers from cache Contamination, i.e. the unpopular data will remain in the proxy cache for a longer period and it suffers from cache pollution. For reducing cache contamination in LRU, a VFDT classifier is integrated with LRU to form a new approach Called VFDT-LRU in order to increase the performance of the Byte hit ratio.

5. Experimental Results

5.1. Web Log Pre-processing

The Data sets[15] to be used for simulation undergo some pre-processing technique to remove irrelevant requests, extract useful information. The steps involved in web log pre-processing[9] are Data cleaning, Parsing and Filtering which removes irrelevant request and to identify the boundaries between the successive records stored. Finally an output format generated after the pre-processing[9] steps consist of the following fields shown in Table 4.

5.2. Training Phase

In this phase, the Pre-processed log file[9] was trained by the sliding window method SWL[3,6] in order to obtain the Frequency and Recency of the web object as shown in Table 3. In this paper, we have used the Sliding window[4] length of 25 minutes, i.e. (1500sec) as the time period to train the pre-processed data sets for the recency, frequency, value as the object, usually stays in the cache. This generated recency, frequency obtained by sliding window and its timestamp, the size of the object used as the input for the Very Fast Decision tree learning method called VFDT which classify the web object to could be re-visited again or not. In this phase, the trained log file has been cross-validated as testing and training data. Thus the concluded data set is arranged and normalized according to the series [0,1] i.e. Cacheable or un-cacheable data. Here we have used the open source software VFML[14] for the VFDT classifier, finally the classified data set has been integrated into the web proxy caching algorithms LRU, GDSF for Replacement strategies.

Table 4. Sample Training Data Based on Sliding window Mechanism.

URL-ID	Timestamp	Recency	Frequency	Retrieval Time	Size	Predicted Output
1	1168387236.353	1500	1	216	43907	1 (cacheable)
2	1168387236.321	1500	2	214	14179	1 (cacheable)
3	1168387236.621	1500	1	284	890	0 (un-cacheable)
4	1168387238.141	1500	1	263	15679	0 (un-cacheable)
1	1168387238.581	1500	2	71	43097	0 (un-cacheable)
5	1168387240.492	1500	1	203	8592	0 (un-cacheable)
3	1168387240.521	1500	2	231	218	1 (cacheable)
4	1168387241.480	1500	2	875	217	1 (cacheable)

6. Performance Evaluation

6.1. Classifier Evaluation

Accuracy: Classification Accuracy are the performance parameters suited in many machine learning[10,11] applications for measuring how accurately the data sets are classified. In general, Classification accuracy for VFDT is commonly evaluated as the percentage of correctly classified instances over the total number of samples.

From the below given graph in Fig. 3. It is known that the classification accuracy of VFDT is higher than other classifier like ID3 and BPNN.

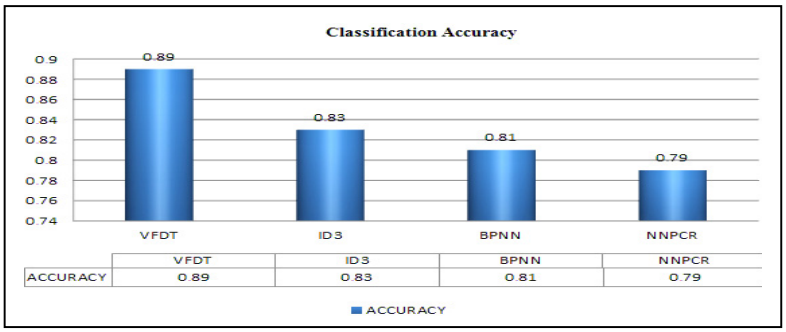

Fig. 3. Comparison of Classification Accuracy

Time : The computation time is the total time taken in seconds for processing a given full sets of data sets .In summation to that the computational time for training VFDT is faster than ID3, BPNN and NNPCR etc. for all Data sets in given below in Table 5.

Table 5. Computational Time for sample log data sets.

Datasets	VFDT	ID3	BPNN	NNPCR
Bo2	0.13	0.86	15.32	20.39
NY	0.08	0.44	15.50	21.60
UC	0.24	1.85	16.10	24.62
SV	0.96	1..60	14.63	16.23
SD	0.55	1.12	16.64	20.01
AVG	0.392	5.87	15.638	20.57

6.2. Evaluation of Web Proxy Caching

6.2.1 Implementation

The Implementation of web proxy log file can be generated by WebTraff[13] simulator which is used to evaluate the performance metrics of the cache replacement algorithm. This tool is written in C++ and TCL scripts for the Generating the Web proxy workload which comes under with two process called ProwGen trace and Web proxy simulation.

6.2.2 Performance Measures

The overall performance of the cache replacement algorithm is measured in terms of cache hit ratio and byte hit ratio given in Table 6.

Table 6. Examples of Performance metrics used in Cache Replacement Policies.

Metric	Description	Formula
Cache Hit Ratio (HR)[8]	Cache Hit Ratio the number of requests satisfied from the proxy cache as a percentage of the total Request.	$\frac{\sum_{d \in D} cr_d}{\sum_{d \in D} r_d}$
Byte Hit Ratio (BHR)[8]	Byte Hit Ratio (weighted ratio) the amount of byte transfer from the proxy cache as a percentage of total number of bytes for the entire request	$\frac{\sum_{d \in D} s_d \cdot cr_d}{\sum_{d \in D} s_d \cdot r_d}$

From, the given graph below in Fig. 4. Specifies that integrated TANB-GDSF increases GDSF performance in terms of cache Hit Ratio and TANB-LRU increases over LRU in terms of Byte Hit Ratio.

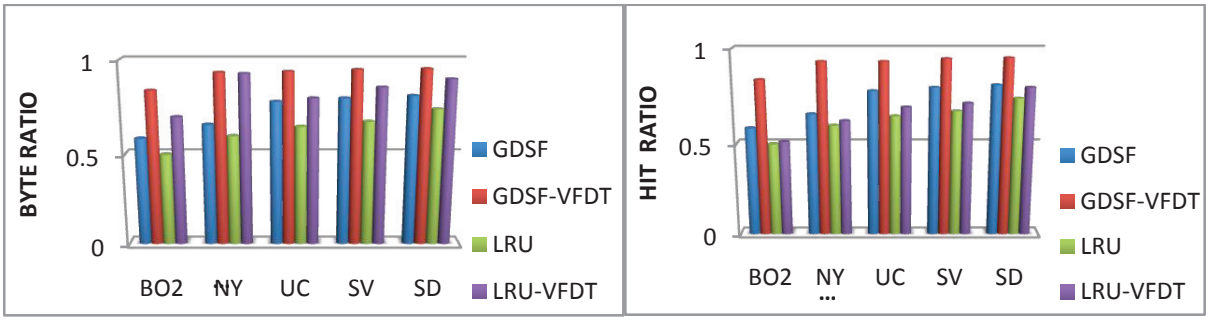

Fig. 4. Cache Hit Ratio and Byte Hit ratio for Different Data Set.

7.Conclusion

This work proposes by implementing a new approach, namely VFDT-LRU and VFDT-GDSF for improving the overall performance of the cache replacement of algorithms. Experimental results have revealed that VFDT achieves much better accuracy compared with other classifiers. In addition, in future we can consider incorporating the classification technique in the surrogate server for improving the performance of the Content Distribution network.

References

1. Mirela Teixeira Cazzolato, Marcela Xavier Ribeiro. Classifying High-Speed Data streams Using statistical Decision Trees . *Journal of Information and Data Management* 2014;0**5**:84-93.
2. Waleed Ali, Siti Mariyam Shamsuddin, Abdul Samad Ismail. Intelligent Naïve Bayes-based approaches for Web proxy Caching. *Knowledge-Based System* 2012;**31**:162-175.
3. Romano S, ElAarag H. Neural network proxy Cache replacement strategy and its implementation in the squid proxy server. *Neural computing and Applications* 2011;**20**:59-78.
4. Ali Ahmed W, Shamsuddin S. Neuro-Fuzzy System in Partitioned client side web cache. *Expert System with Application* 2011;**38**:14715-14725.
5. Kumar C, Norris J.B. A New approach for a proxy-level web caching mechanism. *Decision Support System* 2008;**46**:52-60.
6. Jake Cobb , Hala ElAarag. Web Proxy Cache replacement scheme based on back-propagation neural network. *The Journal of systems and software* 2008;**81**:1539-1558.
7. Podlipnig S, Boszormenyi L. A survey of web cache replacement strategies. *ACM Computing Surveys* 2003;**35**:374-398.
8. Kin-Yeung W. Web Cache replacement policies a pragmatic approach. *IEEE Network* 2006;**20**:28-34.
9. Liu B. *Web Data Mining*. 2nd ed. Springer Heidelberg Dordrecht London New York: Springer; 2011.
10. Han J, Kamber M. *Data Mining concepts and Techniques*. 3rd ed. Morgan Kaufmann; 1979.
11. Mitchell T. *Machine Learning*. 1st ed. McGraw-Hill; 1997.
12. Hulten G, Spencer L, Domingos P. Mining time changing Data streams. Inc: Proc.7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining *„ACM PRESS*, New York 2001.pp.97-106.
13. Markatchev N, Williamson C. Web Traff a GUI for Web Proxy Cache workload Modeling and analysis. Inc:Proc.10th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication systems, *IEEE Computer Society*; 20002. pp. 356-363.
14. VFML Mining High Speed Data Stream Software Tool available in; *http://www.cs.washington.edu/dm/vfml.*
15. NLANR, National Lab of Applied Network Research and Sanitized Access Logs*; http://www.ircache.net/2010*.