ANALYSIS OF WEB-PROXY CACHE REPLACEMENT ALGORITHMS UNDER STEADY-STATE CONDITIONS[*]

L. G. Cárdenas, A. Pont, J. Sahuquillo, J. A. Gil
Department of Computer Engineering, Polytechnic University of Valencia, Camino de Vera s/n 46071, Valencia, Spain

Keywords: Performance Evaluation, Simulation, Proxy-Web, Replacement Algorithms.

Abstract: Web-Proxy servers are used to reduce the bandwidth consumption and users' perceived latency while navigating the WWW, by caching the most frequent objects accessed by users. Since they were introduced, most of the evaluations studies related to Web-Proxy caches have focused on the replacement algorithms performance using simulation techniques. But few of them have been done assuring the representativeness of the studies and considering real traces and cache sizes.

This paper describes a methodology that permits fair performance comparison studies of replacement algorithms, that is, the system reaches the steady-state and the results are provided showing narrow confidence intervals. An experimental evaluation study applying this methodology is also presented. The study uses a trace-driven simulation framework, real traces containing more than one hundred million of user's requests, and compares three replacement algorithms implemented in actual Web-Proxy caches.

1 INTRODUCTION

Web-Proxy servers are placed between the users surfing the Web and the original servers with the purpose of caching the most frequently requested objects and saving both latency and bandwidth. Due to proxy disk space limitations, *replacement algorithms* are used by the cache management to improve the disk space utilization. Most of the Web-Proxy performance studies have focused on the evaluation of these replacement techniques using analytical or simulation models.

Simulation is the most common approach used to model Web-Proxy server cache replacement algorithms (Cao and Irani, 1997), (Arlitt et al., 1998), (Arlitt et al., 1999), (Jin and Bestavros, 2000), (Bahn et al., 2002). There is a large amount of Web-Proxy server cache replacement algorithms proposals, however, few of them can be implemented in an actual Web-Proxy system such as Squid (Squid, 2005). This is because some of the parameters used in the eviction mechanism are difficult to estimate as they are strongly trace-dependent.

In any evaluation study, it is important to provide two conditions in order to guarantee the representativeness of the study: i) to reach a steady-state system after an adequate warming phase, and ii) to obtain results afterwards using confidence intervals. This paper presents an evaluation study of Web-Proxy server replacement algorithms matching both conditions, therefore accomplishing a fair comparison. We show that the evaluation must reach a steady-state system so reflecting accurate performance of the replacement algorithm. The confidence intervals show how precise the estimation is for the evaluated metrics.

The study evaluates the most common metrics to quantify the effectiveness of the replacement algorithms: the Hit Ratio (HR) and the Byte Hit Ratio (BHR), and additionally it includes the results of the Delay Saved Ratio (DSR). We use a large set of Web-Proxy cache traces obtained from a Web-Proxy server cache of an University Campus and from the IRCACHE Proxy-Server cache system. The obtained results show a wide confidence interval when mea-

[*]This work has been partially supported by the: i) Spanish Ministry of Education and Science and the European Investment Fund for Regional Development (FEDER) under grant TSI 2005-07876-C03-01, and ii) Mexican Government Grant CONACYT 177846/191901.

G. Cárdenas L., Pont A., Sahuquillo J. and A. Gil J. (2007).
ANALYSIS OF WEB-PROXY CACHE REPLACEMENT ALGORITHMS UNDER STEADY-STATE CONDITIONS.
In *Proceedings of the Third International Conference on Web Information Systems and Technologies - Internet Technology*, pages 253-260
DOI: 10.5220/0001285702530260

suring the Byte Hit Ratio, overlapping the results of different replacement algorithms. As a consequence, we can affirm that there is a group of algorithms that achieves similar performance when considering the Byte Hit Ratio metric. This result is opposite to most of the published conclusions in the literature, where an algorithm achieves the best performance among the evaluated algorithms. On the other hand, the results measuring the Hit Ratio metric have a narrow confidence interval allowing to select the best replacement algorithm for the workload used.

The remainder of this paper is organized as follows. Section 2 describes the related work in Web-Proxy server cache evaluations. Section 3 describes the experimental environment used. Section 4 presents the evaluation of the Web-Proxy server cache replacement algorithms, and analyzes the experimental results. Finally, Section 5 presents some concluding remarks.

2 RELATED WORK

The most commonly evaluation technique used in Web-Proxy cache evaluation studies is the trace-driven simulation using a captured log as a workload (Wooster and Abrams, 1997), (Cao and Irani, 1997), (Arlitt et al., 1998), (Arlitt et al., 1999), (Jin and Bestavros, 2000), (Bahn et al., 2002). Few of them (Arlitt et al., 1998), (Arlitt et al., 1999) include in their evaluation methodology a *warming-phase*, only one of them (Wooster and Abrams, 1997) provides results using confidence intervals, and none of them provides results in both conditions. These studies use the HR and the BHR metrics to compare and to evaluate the performance of the proposed replacement algorithms. The HR indicates how efficient a Web-Proxy server is, and the BHR indicates how much of the total amount of bytes requested are served from the cache (i.e., how much traffic volume can be saved). On the other hand, time-related metrics are seldom used due to the difficulty when modeling accurately the penalty time for the Internet accesses. This section briefly describes these works.

Cao and Irani (Cao and Irani, 1997) propose the GreedyDual-Size (GDS) replacement algorithm using a large set of captured logs with a restriction of two million requests in each simulation. The GDS is an extended version of the GreedyDual algorithm that incorporates the object size, setting a *cost/size* value for each object in the cache. They perform a trace preprocessing discarding the log entries with a 304 HTTP response-code. Therefore, the simulation results could not reflect the current performance of the replacement algorihtms because the log entries discarded represent a large amount of the total of log entries. Their proposal achieves the best performance for the three metrics measured (the HR, the BHR, and the Reduced Latency).

Arlitt *et al.* (Arlitt et al., 1998) (Arlitt et al., 1999) propose and evaluate two formula-based replacement algorithms: Greedy-Dual Size Frequency (GDSF) and the Least Frequently Used with Dynamic Aging (LFUDA) using a single captured log of three months length. The GDSF keeps the smaller size popular requested objects in the cache (a smaller size object has higher probability of being frequently requested). On the other hand, the LFUDA keeps the most popular objects in the cache, regardless of their size. Both algorithms incorporate an aging mechanism to avoid the cache pollution. The HR is calculated considering the un-cacheable log entries as misses and including the 304 status responde-code log entries; and the BHR is calculated using only the logs containing the object size information. The authors provide results for the HR and the BHR metrics using a warming phase of three weeks; but no confidence intervals are shown. Their results show that size-based policies (i.e., the GDSF) achieve a better HR, and that frequency-based policies (i.e, LFUDA) obtain a better BHR. Both algorithms are currently implemented in the Squid Proxy Cache (Squid, 2005).

Jin and Bestravos (Jin and Bestavros, 2000) propose the GreedyDual-Size Popularity (GDSP) algorithm, and evaluate it using two captured logs (with approximately four million requests in each log). The GDSP uses an eviction mechanism similar to the GDSF. They use the same trace prepocessing to discard the log entries of Cao and Irani but they estimate a penalty time for each log entry (simbolizing an object transmission from the Web-Server) using a mathematical formula, which does not consider the benefits of the current HTTP/1.1 protocol (i.e., persistent connections) (Fielding et al., 1999). The parameters of this mathematical formula are estimated using a *least-square fit*. In the evaluation, the proposed GDSP presents the best results for the HR, the BHR and the Latency Saving Ratio metrics.

Bahn *et al.* (Bahn et al., 2002) propose the Least Unified-Value (LUV) replacement algorithm and compare it against a large set of replacement algorithms using two captured logs with a small amount of requests. For this purpose, the experiments consider small cache sizes, unrealistic in the actual Web-Proxy server caches. The main drawback of this study is the trace preprocessing required to adjust a parameter by the eviction mechanism. In spite of this disadvantage, they conclude that the implementation is efficient in both time and space complexities. Their results show the LUV as the best replacement algorithm for all the metrics studied (HR, BHR, and the DSR). There is no information about how the penalty times are estimated to calculate the DSR, or how the traces are prepared to feed the simulator. Finally, they neither use a warming phase, nor estimate confidence intervals.

3 EXPERIMENTAL ENVIRONMENT

This section describes the simulation framework, the workload, the performance metrics, and the replacement algorithms used in this evaluation study.

3.1 Simulation Framework

We present in (Cardenas et al., 2004) an experimental framework for modeling Web-Proxy server caching replacement algorithms, and validate this framework in (Cárdenas et al., 2005) againts several replacement algorithms of a real Web-Proxy server cache. This section summarizes the main features of such environment.

The Multikey Web Cache Simulator (MSE) is an object-oriented simulator, and allows an easy implementation of new cache replacement algorithms. This simulator is composed by three modules: the filter module, the cache-simulator module, and the statistics module. The *filter* module prepares the Web-Proxy server logs to feed the cache-simulator module. The *cache-simulator* module models a Web-Proxy cache replacement algorithm including the cache characteristics of a parameters files. The *statistics* module takes as input: i) the trace obtained in the cache-simulator module, and ii) the number of requests of the warming phase required to reach a steady-state system. The statistics module generates a results file for each metric evaluated (i.e., the HR, the BHR, and the DSR) using confidence intervals with a 95 % confidence level.

3.2 Workload

This section describes the steps taken to prepare the Web-Proxy server logs used in our evaluation study. The traces used were obtained from the Web-Proxy server logs of the Polytechnic University of Valencia (UPV), and the IRCACHE Proxy-Cache system (IRCache, 2005). These Web-Proxy servers use the Squid Proxy Cache software (Squid, 2005), and every trace represents approximately one month of browsing activity. The prepared traces were *LogJun05*, *LogNov05* and *LogMay06* from the UPV, and *LogFeb06-pb* from the IRCACHE Proxy-Cache system. The original total amount of log entries in the UPV are hundreds of millions, while in the IRCACHE are tens of millions.

The trace has a strong influence on the performance of the replacement algorithm, therefore a correct trace preparation is required in order to obtain results reflecting the behavior of the replacement algorithm. The trace preparation was made in two phases: a) the filtering of the un-cacheable content by the Web-Proxy, and b) the replacing of inappropriate information registered in the original Web-Proxy server log. The UPV Web-Proxy logs required a *prior* phase excluding the log entries requesting content from the campus Web-Servers.

The first phase starts eliminating the log entries with an un-cacheable HTTP status response-codes, as stated in the HTTP/1.1 protocol (Fielding et al., 1999). Then, we exclude the log entries requesting dynamic content, since this content is usually not cached by a Web-Proxy server.

Table 1 shows the percentage over the total of the log entries with a 200 and 304 HTTP status response-code in the *LogJun05*, *LogNov05*, *LogMay06* and *LogFeb06-pb* traces after the first phase preparation. The log entries with a 304 HTTP status response-code are a large amount of the total amount of requests in every trace; in the UPV and the IRCACHE trace, this percentage is never less than 33.71% and 15.17%, respectively. Although, these type of requests are not cached by a Web-Proxy, they are Web-Client validation messages of a content already stored in the Web-Proxy server cache, therefore they must be considered in the evaluation experiments.

Table 1: Percentages over the total of the 200 and 304 HTTP status response-code log entries for the traces *LogJun05*, *LogNov05*, *LogMay06.* and *LogFeb06-pb*.

HTTP	LOG			
response-code	Jun05	Nov05	May06	Feb06-pb
200	62.18	60.99	60.73	84.83
304	37.82	39.01	39.27	15.17

In the second phase of the trace preparation we replace log entries data fields. The first step of this phase is to identify which type of log entries require the estimation of new information. There are two cases: a) when the penalty time information does not represent the transmission time of an object requested to the Web-Server from the Web-Proxy server, and b) when the response size information does not symbolize the size of the object requested (like in the

If-not-modified-since log entries) since the registered value is the size of a validation message (approx. 250 bytes). Table 2 shows the duple Squid-HTTP status response-code of the log entries and the cases where a replace is required.

Table 2: Squid-HTTP status response-code of the log entries and the cases where data replacement is required.

Response-code	Object Size	Penalty Time
TCP_HIT/200	No	Yes
TCP_MISS/200	No	No
TCP_MISS/304	Yes	Yes
TCP_REF_MISS/200	No	No
TCP_REF_HIT/200	No	No
TCP_REF_HIT/304	Yes	Yes
TCP_IMS_HIT/200	No	Yes
TCP_IMS_HIT/304	Yes	Yes
TCP_MEM_HIT/200	No	Yes

We use the Log-Logistic probabilistic distribution function when we need to estimate the penalty time. This function approximates the penalty times cumulative distribution function of a Web-Proxy server cache (including the long tail). The reason why we select this probabilistic distribution can be found in (Gil et al., 2005). In the cases where we need to estimate a new object size, we use an estimated mean object size for its corresponding MIME type. The MIME (Multipurpose Internet Mail Extensions) (Freed and Borenstein, 1999b) (Freed and Borenstein, 1999a) groups the type of responses using a well understood ”*di-facto*” standard.

Table 3 shows the total amount of requests, the traffic volume, the total amount of unique objects (Unique O.), the cache space required to store all the unique objects, and the theoretical Hit Ratio (Max Hit Ratio) that will be obtained using a Proxy with an infinite cache size for the *LogJun05*, *LogNov05*, *LogMay06* and *LogFeb06-pb* after the second phase trace preparation.

Table 3: Main Statistics of the Traces *LogJun05*, *LogNov05*, *LogMay06.* and *LogFeb06-pb*.

Statistic	LOG			
	Jun05	Nov05	May06	Feb06-pb
Requests	34.1M	48.3M	37.8M	4.2M
Traffic (GB)	730	834	725	150
Unique O.	5.4M	7.7M	6.4M	2.4M
Unique O.(GB)	225	243	193	92
Max Hit Ratio	84.13	84.14	83.03	42.23

3.3 Performance Metrics

As mentioned above, the performance metrics used in our evaluation are: HR, BHR, and the DSR. The DSR is calculated in our experiments because we can estimate the penalty times for the log entry cases where this information is not available in the log.

The Hit Ratio (equation 1) indicates the percentage of web objects served from the proxy cache. It is a measure of how efficient is the cache management. Although the HR is the most used metric in Web-Proxy cache evaluation studies, there is not evident how this metric quantifies the benefits about the Web-Proxy from the users' point of view.

$$HR = \frac{\sum_{i=1}^{n} Hit_i}{\sum_{j=1}^{m} Requests_j} \qquad (1)$$

The Byte Hit Ratio (equation 2) indicates how much of the total amount of bytes requested have been served from the cache. Some authors (Wessels, 2001) (Rabinovich, 2002) claim that the BHR is directly related to other metrics such as the saved bandwidth.

$$BHR = \frac{\sum_{i=1}^{n} Hit_{i(Bytes)}}{\sum_{j=1}^{m} Requests_{j(Bytes)}} \qquad (2)$$

The Delay Saved Ratio (equation 3) is a time-related metric and has been defined as the sum of penalty times of the hits (as if they were all misses) over the sum of the penalty times from all the requests (again, as if they were all misses). This metric indicates the amount of penalty times saved by the Web-Proxy server, instead of being served by the original Web-Server.

$$DSR = \frac{\sum_{i=1}^{n} Hit_{i(PenaltyTime)}}{\sum_{j=1}^{m} Requests_j(PenaltyTime)} \qquad (3)$$

3.4 Replacement Algorithms

The replacement algorithms select which objects must be evicted from the cache when space is only required for a new incoming object. We evaluate the replacement algorithms implemented in the Squid Proxy Cache (Squid, 2005), a real Web-Proxy system, because many of the theoretical proposals in the literature require a parameter estimation obtained from a preprocessing of the traces in order to perform optimally.

The Least Recently Used (LRU) is the most popular replacement algorithm implemented in a Web-Proxy server. This algorithm exploits the temporal locality of the user's accesses, and it is very simple to implement because the eviction mechanism requires only the access time-stamp. The Greedy Dual Size Frequency (GDSF) and the the Least Frequently Used Dynamic Aging (LFUDA) are formula-based algorithms. The GDSF tries to maximize the HR metric, and the LFUDA tries to maximize the BHR metric. The eviction mechanism of the GDSF uses equation 4 and the LFUDA uses equation 5 to estimate a value, which is used as a key, to select which objects are evicted from the cache (the objects with the smaller key gets evicted first). L is an aging mechanism used to avoid the pollution of the cache, expelling from the cache not-recently referenced objects with a high frequency of access.

$$GDSF = \frac{Frequency}{Size} + L \tag{4}$$

$$LFUDA = Frequency + L \tag{5}$$

4 EVALUATION STUDY

This section presents the methodology of our Web-Proxy Server cache evaluation. As mentioned above two conditions are necessary to guarantee the accuracy of any study: i) to reach a steady-state, and ii) to obtain results with narrow confidence intervals. First, we perform a small set of experiments to analyze the importance of reaching a steady-state system for the evaluation of Web-Proxy server cache replacement algorithms. Then, we explain the steps of the proposed methodology used in this evaluation study. Finally, we present the obtained results of the evaluation.

4.1 Reaching the Steady-State System

The *steady-state* condition is reached when the Hit Ratio of the Web-Proxy cache is stable over time, in spite of the user's accesses variations. To reach this state, a warming phase is needed to avoid the cold misses; this kind of misses could alter the results of any performance evaluation.

Table 4 shows the Hit Ratio (HR), the Byte Hit Ratio (BHR), and the Delay Saved Ratio (DSR) obtained in a simulation experiment of an infinite cache using the *LogMay06* trace: i) without a warming phase, ii) with a warming phase of twenty million requests, and iii) the percentage difference in these two states. An infinite cache size would contain all the objects requested to a Proxy and in our case all the objects included in the trace. Notice that the results obtained with and without a warming phase present significant differences that can alter the conclusions of any study. These differences can be also observed in the values of the Max Hit Ratio numerically calculated (see Table 3) for the infinite cache size when simulations with a warming phase are carried out. Therefore the steady-state is a required condition for any evaluation study.

Table 4: The HR, BHR and DSR with and without a warming phase, and difference with an infinite cache under the trace *LogMay06*.

Metric	Non-warming	Warming	Difference
HR	82.71	85.63	3.53
BHR	69.91	74.88	7.10
DSR	69.35	73.75	4.4

Table 5 shows the HR,the BHR, and the DSR obtained in a simulation experiment of a 8GB cache using the GDSF replacement algorithm without a warming phase, and with a warming phase of: 5, 10, 15, 20, 25, 30, 35 million requests using the *LogMay06* trace. As we can observe, the values obtained using the HR and the DSR metric are higher as the amount of the requests of the warming phase increases; as opposite, the values obtained in the BHR metric decrease. Besides, we can observe the variation of the values of the different metrics (compare to the non-warming experiment) as we increment the amount of log entries used in the warming phase.

Table 5: HR, BHR and DSR without a warming phase, and using different warming phase length of a 8GB cache simulation experiment with the GDSF using the trace *LogMay06*.

Warming	Metric		
Amount	HR	BHR	DSR
No	81.07	38.04	61.77
5M	82.14	37.28	63.04
10M	82.47	36.92	63.25
15M	82.90	36.33	63.53
20M	83.17	36.47	63.74
25M	83.23	35.91	63.78
30M	83.73	35.62	66.15

4.2 Methodology

The evaluation methodology used is organized in the following steps. First, we select the number of requests of the warming phase, allowing to reach a *steady-state* system. Throughout all the experiments we use a *twenty million* requests warming phase for the UPV traces, and two million requests for the IR-CACHE traces. This amount of requests will allow to fill the cache space, and to perform a substancial amount of evictions as training; no results are collected from this period. Then, we estimate the confidence intervals for all the metrics (the HR, the BHR and, the DSR). Our confidence intervals are calculated with a confidence level of 95%.

4.3 Results

The experiments consider cache sizes ranging from 1GB to 32GB typically used in Web-Proxy cache systems (IRCache, 2005). Simulation experiments with a cache size larger than 32GB are worthless, because the results obtained with a cache size of 32GB using the HR metric are very close to the *Infiinite Cache* Hit Ratio. For comparison purposes, we calculate the Infinite Hit Ratio (IHR), the Infinite Byte Hit Ratio (IBHR), and the Infinite Delay Saved Ratio (IDSR) for all the studied traces. These values allow us to compare the simulation results of different cache sizes with the results of an infinite cache.

Figure 1 shows the HR obtained for the traces *LogJun05*, *LogNov05*, *LogMay06* and *LogFeb06-pb* using cache sizes from 1 to 32GB. The GDSF obtains the best HR in all the traces evaluated for all the cache sizes employed. This is indeed an algorithm that maximizes the HR, as in all traces the IHR is almost reached with 32GB cache size. As we observe in the figure, the results measuring the HR have a narrow confidence interval; therefore allows to select the GDSF as the best replacement algorithm as stated in the literature.

Figure 2 shows the BHR obtained for the traces *LogJun05*, *LogNov05*, *LogMay06* and *LogFeb06-pb* using cache sizes from 1 to 32GB. We first observe a wide confidence interval, compared to the confidence intervals obtained measuring the HR, this is the effect of the high size variation in the requested objects. This has a direct consequence: we cannot select an unique best algorithm, but a group of algorithms with similar performance which vary depending on the workload. Therefore, we can affirm that the performance of the LFUDA (the algorithm below the IBHR in the figure), is as good as the LRU (the second below the IBHR) because the confidence intervals overlap. The GDSF algorithm obtains the worst BHR for the used traces.

Figure 3 shows the DSR of the traces *LogJun05*, *LogNov05*, *LogMay06* and *LogFeb06-pb* using cache sizes from 1 to 32GB. Again, we observe a wider confidence interval, as in the results obtained measuring the BHR. The top group of algorithms for this metric is conformed by the LFUDA and the GDSF, while in the IRCACHE trace any of the studied algorithms could be selected as the best one.

The narrow confidence interval obtained in the HR metric is a consequence of the system stability during the simulation experiments. The HR is estimated calculating the ratio of the total amount of hits over the total amount of requests. This ratio is stable after the system reaches the warming phase.

On the other hand, the BHR is estimated using the response size. This value could vary in several orders of magnitude from request to request (from a few KB to some GB). As a consequence, there is a wide confidence interval in the BHR metric caused by the variations of the responses size. Finally, the wide confidence interval obtained in the DSR metric is originated by the variations of the penalty times.

5 CONCLUSIONS

A large amount of research works has focused on the evaluation of Web-Proxy server cache replacement algorithms using simulated systems using the captured logs as workload. Among these works, few comparisons include a warming phase to reach the *steady state*, only one obtain results using *confidence intervals*, and none include both of these conditions. As a consequence, the results obtained and the conclusions reached could have a low representativeness and be questionable.

This paper describes a methodology to fairly compare Web-Proxy server cache replacement algorithms when accomplishing the mentioned conditions. The preliminary experiments results show that an evaluation of Web-Proxy server cache replacement algorithms without reaching the steady-system could alter the obtained results.

We show the experimental results for the most common Web-Proxy server cache replacement algorithms using the proposed evaluation methodology. The comparison shows that when using confidence intervals no replacement algorithm can be identified as the best one, as it is usually presented in the literature, but a group of *best-performing* algorithms. The results obtained using the Hit Ratio metric have narrow confidence intervals generated by the system stability, therefore, we select the GDSF as the replacement algorithm that maximizes this metric (as also mentioned in the literature). But, as the confidence intervals for the Byte Hit Ratio metric are wider (due to the response size variations), we can affirm that the LFUDA and the LRU perform equally good. As a consequence, we can select any of these algorithms to reduce the traffic, and as the LRU presents less CPU overload than the LFUDA, the LRU could be the best choice to reduce the bandwith consumption.

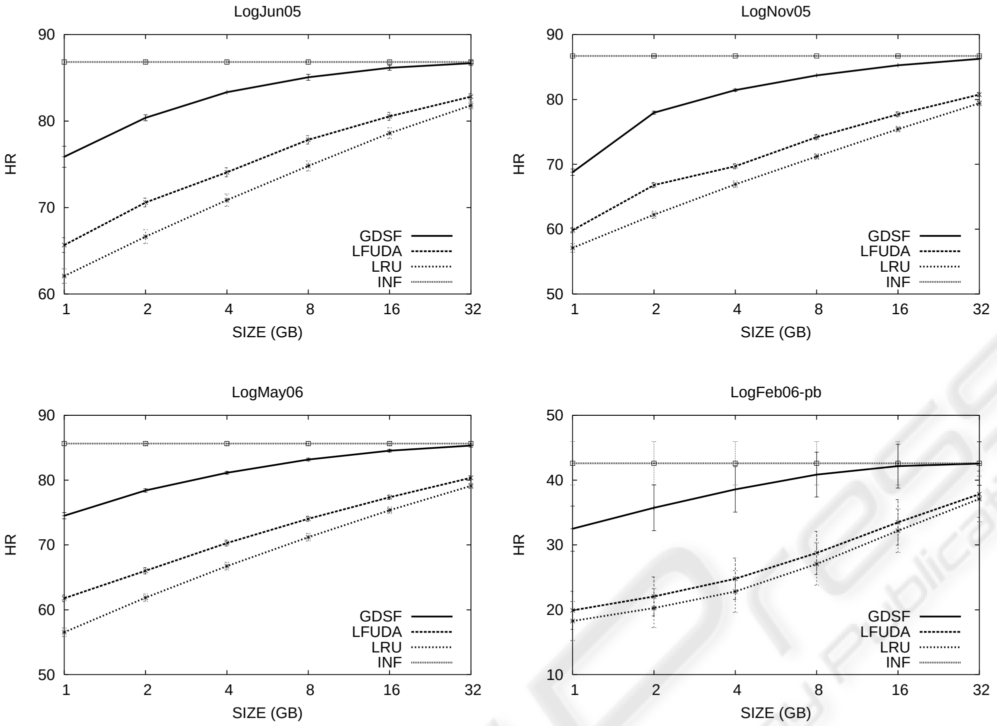

Figure 1: The Hit Ratio of traces *LogJun05*, *LogNov05*, *LogMay06.* and *LogFeb06-pb*.

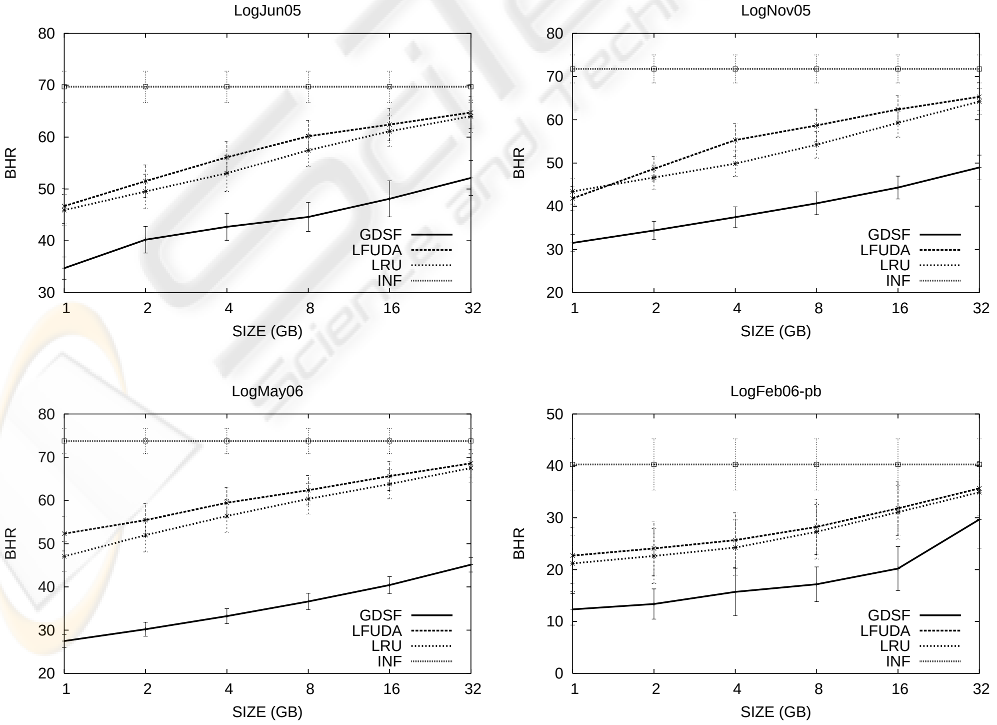

Figure 2: The Byte Hit Ratio of traces *LogJun05*, *LogNov05*, *LogMay06.* and *LogFeb06-pb*.

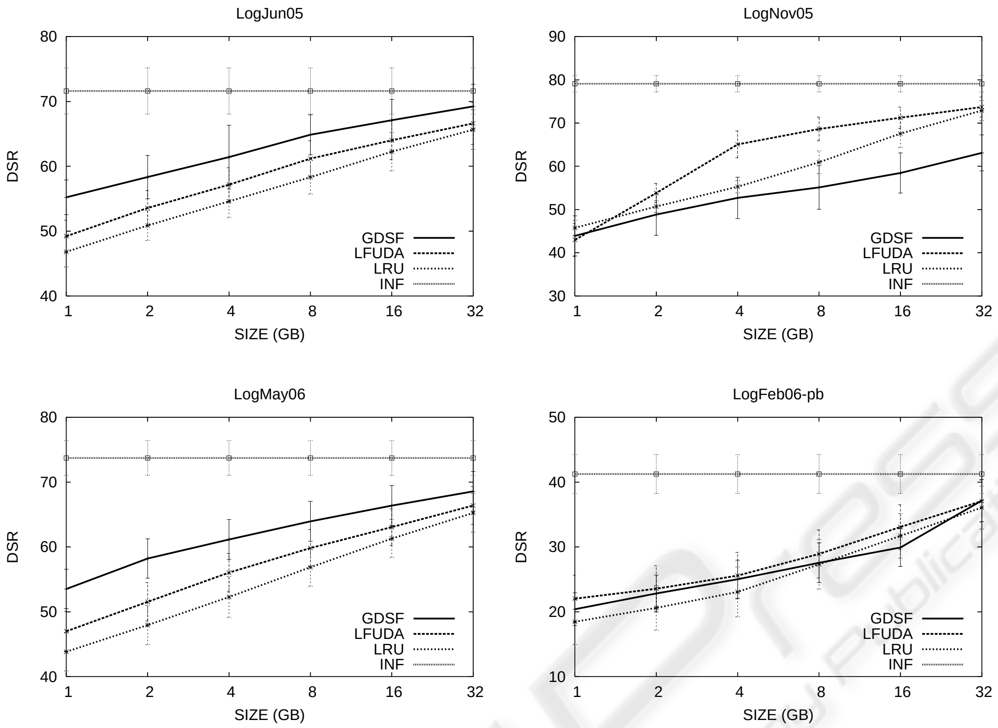

Figure 3: The Delay Saved Ratio of traces *LogJun05*, *LogNov05*, *LogMay06.* and *LogFeb06-pb*.

REFERENCES

Arlitt, M., Cherkasova, L., Dilley, J., Friedrich, R., and Jin, T. (1999). Evaluating content management techniques for Web proxy caches. In *Proceedings of the Workshop on Internet Server Performance (WISP99)*.

Arlitt, M., Friedrich, R., and Jin, T. (1998). Performance evaluation of Web proxy cache replacement policies. *Lecture Notes in Computer Science*, 1469:193+.

Bahn, H., Koh, K., Min, S. L., and Noh, S. H. (2002). Efficient replacement of nonuniform objects in web caches. *IEEE Computer*, 35(6):65–73.

Cao, P. and Irani, S. (1997). Cost-aware www proxy caching algorithms. In *USENIX Symposium on Internet Technologies and Systems*.

Cárdenas, L. G., Gil, J. A., Sahuquillo, J., and Pont, A. (2005). Emulating web cache replacement algorithms versus a real system. In *10th IEEE Symposium on Computers and CommunicationsISCC2005*.

Cardenas, L. G., Sahuquillo, J., Pont, A., and Gil, J. A. (2004). The multikey web cache simulator: a platform for designing proxy cache management techniques. In *12th Euromicro Conference on Parallel, Distributed and Network based Processing PDP2004*.

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and Berners-Lee, T. (1999). Hypertext transfer protocol – http/1.1. RFC 2616 - Network Working Group.

Freed, N. and Borenstein, N. (1999a). Multipurpose internet mail extensions part one: Format of internet message bodies. RFC 2045 - Network Working Group.

Freed, N. and Borenstein, N. (1999b). Multipurpose internet mail extensions part two: Media types. RFC 2046 - Network Working Group.

Gil, J. A., Cárdenas, L. G., Sahuquillo, J., and Pont, A. (2005). Modeling penalty times in proxy-web cache systems. Technical report, Deparment of Computer Engineering (DISCA) UPV.

IRCache (2005). Proxy-web traces. http://www.ircache.net.

Jin, S. and Bestavros, A. (2000). Popularity-aware greedydual-size web proxy caching algorithms. In *Proceedings of the 20th International Conference on Distributed Computing Systems (ICDCS2000)*.

Rabinovich, M. (2002). *Web caching and replication*. Addison-Wesley.

Squid, T. (2005). Squid web proxy cache. http://www.squid-cache.org.

Wessels, D. (2001). *Web caching*. O'Reilly.

Wooster, R. P. and Abrams, M. (1997). Proxy caching that estimates page load delays. In *World Wide Web conference*.