Characterizing Document Types to Evaluate Web Cache Replacement Policies

F.J. Gonzalez-Cañete, E. Casilari, Alicia Triviño-Cabrera
Dpto. Tecnología Electrónica, Universidad de Málaga, E.T.S.I. Telecomunicación, Campus de Teatinos, 29071 Málaga, Spain
fgc@uma.es

Abstract

In this paper, a study of the performance of six replacement policies has been developed Three of them are classical caching algorithms (LRU, LFU and LFU-DA) while the other three are caching schemes specifically developed for Web documents (GD-SIZE, GDSF and GD). This study has been divided by the main content-types observed in Web traffic (Application, Audio, Images, Text and Video). Using a trace log of a real proxy cache, a characterization of the main properties of the documents has been performed, such as size characteristics, popularity and temporal locality. Finally, a trace driven simulation study of the performance of the replacement policies has been developed for the traffic generated by each considered content-type.*

1. Introduction

Internet and the World Wide Web (the Web) are in a continuous evolution and growth, therefore many efforts to optimize them have been developed. One of the most important optimization techniques is the Web proxy caching that permits to store the documents requested by the users in electronic devices close to them. Since it was proposed in [1], Web proxy caching has been widely utilized to reduce the latency that the users perceive, the HTTP traffic as well as the servers load. After this original proposal of a Web proxy cache, many research activities have aimed to study and develop replacement policies [2], algorithms for cache coherence [3] and cache architectures [4] in order to improve the performance of the caching system.

One of the main research lines is based on differencing the types of documents that are present in the Web (Images, Text, Video,…). Khayari proposed to store in the cache only the most frequently demanded document types (mpeg, gif, jpg, flash, html and plain) although its proposal did not outperform the cache performance [5]. This analysis was performed using the LRU replacement policy. However, different result could be achieved applying others replacement policies. In this paper we analyze the best replacement policy for each document type by means of simulations.

This paper is organized as follows. Section 2 summarizes the trace processing and the statistical characterization of the workload based on the content-type. Section 3 lists and explains the replacement policies considered in Section 4 to evaluate the performance of a proxy cache that takes into account only one content-type of the downloaded document. Finally, Section 5 presents the main conclusions of this paper.

2. Trace Processing and Characterization

To evaluate the performance of a cache that only considers one type of document content type at a time, a common workload trace that contains HTTP requests from a proxy of the IRCache project [6] has been utilised in the different simulations. This proxy is located in the Research Triangle Park (North Carolina, USA). The traces include requests from the 7th to the 11th of June 2004 generated by the Squid Web proxy cache software [7]. The traces include information for each HTTP request processed by the proxy, such as the time the request was initiated, the document size, the URL, the document type (content-type), the request method (GET, POST, …) and the response code from the server.

This trace has been preprocessed to purge those requests that have been generated dynamically by CGI (Common Gateway Interface) because the documents returned by these kind of requests are unique for each request and therefore they should not be cached [8]. Because of this fact, the requests that contain the strings ‘cgi’, ‘cgi-bin’ or ‘?’ have been discarded. Those requests that contain the string ‘:3128’ have been filtered as this is the port that IRCache utilises to interchange information between collaborating caches.

As cacheable response codes, 200 (OK), 203 (Partial), 206 (Partial Content), 300 (Multiple Choices), 301 (Moved) and 302 (Redirects), have been taken into account. For 304 (Not Modified) response code, the size shown in the traces corresponds to the size of the response and not to the real document size (it informs that the document has not been modified since it was requested last time). Consequently these documents have been requested again to the original server to obtain the real size. On the other hand, the documents whose content-type was unknown have been requested again. Table 1 summarises the basic characteristics of the trace after the above mentioned process.

The next step is to divide the requests by the content-type of each document according to those ones defined in [9]. These content-types are: Applications, Audio, Images, Text and Video. Table 2 shows the characteristics of the documents according to its type on the analysed workload.

The Image and Text documents account for the 91 % of requests and the 59 % of bytes transferred, so they are the most influent content-types. There are major differences between the document size of the different types as it can be observed if we compare the mean and median of each type. These differences are better shown in Figure 1 (left), where the cumulative distribution function of the document sizes has been plotted.

Because of the fact that the standard deviation of the sizes is greater than the mean, a possible heavy-tailed behaviour could be presented, so, the complementary cumulative distribution function of the sizes is shown in Fig. 1 (right). According to [10], if straight lines are presented in this representation for at least three orders of magnitude, the heavy-tailed behaviour can be assumed. This behaviour is presented for all content-types. The results presented previously are coherent with those deduced in [11].

To characterize the popularity of each type of document Zipf law has been utilized [12] [13]. This law asserts that the probability *P(i)* for the *i-th* most popular document to be requested is inversely proportional to its popularity ranking as shown in equation (1).

The parameter α is the slope of the log/log representation of the number of references to the

Table 1. Main trace characteristics

Number of Requests	4,040,036
Size (GB)	40.4
Distinct documents	1,713,903
One timers	1,299,217

Table 2. Trace characteristics by content-type of documents

	App.	Audio	Images	Text	Video
Mean (Kbytes)	41	123	4	14	260
Median (Kbytes)	3	7	1	3	573
Std. Dev. (Kbytes)	761	948	21	104	974
Requests (%)	8.08	0.28	75.54	15.77	0.12
Bytes (%)	33.51	3.49	36.33	23.15	3.19
Distinct documents (%)	26.90	38.00	45.98	33.92	75.02

$$P(i) \propto \frac{1}{i^{\alpha}} \quad \text{with } \alpha \text{ close to 1} \tag{1}$$

documents as a function of its popularity rank. The popularity is a good estimator of the cacheability of documents because a document that is very popular is more probable to be referenced again in the near future, so the probability of cache hit increases. Popularity has the handicap that it does not completely model the temporal locality of references to the same document. So, a second parameter is needed to model the temporal correlation between two successive references to the same document.

In [14] a method to model the temporal correlation is proposed drawing in a log/log scale the probability distribution of distances between references to documents with the same frequency of access k in the whole trace and calculating the slope β of the distribution. The slope has been calculated using a minimum square regression approximating the equation (2).

$$T \propto \frac{1}{t^{\beta}} \quad \text{for documents with frequency } k \tag{2}$$

A value of $k = 8$ has been selected for this analysis, although similar values of β are obtained for other values of k. Table 4 summarizes the values that we have obtained for α and β for each content type.

Another characteristic to study is the relationship between document size and popularity since some replacement algorithms, such as the algorithms of the Greedy-Dual family, give more preference to small documents than to big ones because they suppose the fact that small documents are more accessed than bigger ones. If we represent the popularity as a function of the document size, we obtain the results shown in Figure 2. These figures demonstrate that this relationship is valid for all content-types except for Video documents where the distribution is sparse.

Table 3. α and β parameters for the aproximantions of the distributions of popularity of documents and temporal locality by content-type

Content-Type	α	β
All	0.64	0.46
Application	0.78	0.51
Audio	0.45	0.40
Image	0.62	0.46
Text	0.73	0.54
Video	0.50	0.95

3. Replacement policies

Many replacement policies have been proposed for Web caching [15], but only six of them have been considered in this study. LRU, LFU and LFU-DA have been selected because they represent classical replacement algorithms, while GD-SIZE, GDSF and GD* represent algorithms specifically developed for Web caching as they take into account the document size. In this section we explain how these replacement policy algorithms work and their advantages and drawbacks.

- LRU (Least Recently Used): It replaces the document that was least recently referenced. It has the advantage that it is a very simple algorithm which can be implemented to work very efficiently. On the other hand it has the handicap that it does not take into account the frequency and size of documents.

- LFU (Least Frequently Used): This algorithm evicts the document that has been least referenced. If there are some documents with the same reference count, LRU is used. It has the disadvantage that documents that have been referenced very often in the past but they are not popular any more are maintained in cache and not evicted. This effect is called "cache pollution". It does not take into account the document size either.

- LFU-DA (LFU-Dynamic Aging): It was proposed in [16] as a more practical implementation of the LFU-Aging algorithm [17]. This algorithm solves the "cache pollution" problem of the LFU algorithm using a variable that contains the "age" of the cache, i.e. the number of references to the least frequently used document. When a new document is inserted or the referenced document is already in cache, the reference count of the document is added to the aging variable.

- GD-SIZE (Greedy Dual-Size) [2]: It utilizes a cost function to evaluate the documents. The value of a document is calculated as shown in Eq.3:

$$V(p) = \frac{Cost(p)}{Size(p)} \quad (3)$$

where *Cost(p)* is the transmission cost of document p and *Size(p)* is the size of *p* in bytes. The document evicted will be the one with the lowest evaluation function value. As functions costs Eq. 4 and Eq. 5 have been proposed among others. Eq. 4 considers that every document has the same transmission cost, while Eq. 5 takes into account the number of packets needed to transmit the document as the transmission cost. This replacement policy does not take into account the frequency of the document.

$$Cost(p) = 1 \quad (4)$$

$$Cost(p) = 2 + \frac{Size(p)}{1460} \quad (5)$$

- GDSF (Greedy-Dual Size with Frequency) [18]: It is a modification of the Greedy-Dual-Size algorithm which considers the reference count of documents. The function which weights each document is shown in Eq. 6:

$$V(p) = Freq(p)\frac{Cost(p)}{Size(p)} \quad (6)$$

where *Freq(p)* is the number of references to document p, *Cost(p)* is the cost and *Size(p)* is the size of *p*. As cost functions Eq. 4 and Eq 5 have been utilised.

- GD* (Greedy-Dual*) [19]: This algorithm is a modification of the Greedy-Dual-Size algorithm taking into account the frequency count of documents and the temporal correlation of references using a parameter β in the value function (Eq. 7), where the only difference to the GDSF replacement policy is the parameter β. This parameter is a number between zero and one that models the temporal correlation between two successive accesses to the same document and it is a characteristic of each workload and it was calculated for each content-type in section 2.

$$V(p) = \left(Freq(p)\frac{Cost(p)}{Size(p)} \right)^{\frac{1}{\beta}} \quad (7)$$

4. Evaluation of Replacement Policies

Two metrics are conventionally utilized to evaluate and compare the efficiency of the replacement policies:

- HR (Hit ratio): It is defined as the total number of requests that cause a hit in the cache (i.e. the document is already present in the cache when the request is performed) divided by the total number of requests.
- BHR (Byte Hit Ratio): It is defined as the summation of the document sizes that cause a

hit in the cache divided by the size of the documents processed.

To evaluate and compare the efficiency of the replacement policies for each content-type a proxy simulator has been developed. This simulator implements the policies explained in section 3 and can be configured to simulate different sizes of cache. The simulator processes the trace files explained in section 2 returning a result file that contains parameters such as the HR and BHR, the total size of the cache, the number of documents evicted, the total number of documents at the end of the simulation, etc. 10% of the trace has been used to "warm up" the cache and avoid cold start influences. To distinguish the modification of a document from the interruption of a transfer we compare the difference between sizes of successive requests to the same document. If the difference is less than 5% of the document size, we consider that the document has been modified and it has to be treated as a new document; otherwise a cancel is considered [20].

We first simulated a cache with infinite size to determine the total size filled in the cache for each content-type. The next simulations were performed using the 50%, 30%, 10%, 5% and 2% of the maximum sizes obtained for a each type.

Figure 3 shows the evaluation of the replacement policies for the Application content-type. If we use the constant cost model, the cost based replacement policies clearly outperform the classical algorithms for the HR metric, with a performance close to 60% for a small cache size. This difference is more evident as the cache size decreases. This behaviour is due to the LRU is the worst option to maximize the HR. For the BHR metric the behaviour is similar, although the difference between policies is only about 4%. Under the packet cost model, GD* and GDSF are the best options to maximize the HR, but they are less efficient than with the constant cost model. GD* and GDSF outperform the other policies for the BHR too, obtaining slightly better results than with the constant cost model. In summary, for Application content-types, GD* and GDSF are the best choices, obtaining high HR but low BHR, as could be expected due to the relationship between size and number of references shown in section 2. Small documents are more accessed than bigger ones and the Zipf law slope for Applications is high, therefore a huge HR is obtained even for a small cache. On the other hand, the heavy-tailed behaviour for the Application content-type is the most noticeable, therefore the documents of the tail are responsible of a great percentage of the traffic, but these big documents are not referenced many times. This is the reason why little BHR is obtained.

The simulation results for the Audio content-type are shown in Figure 4. GD*, GDSF and GD-Size obtain the same performance for all cache sizes in the constant cost model and outperform the other algorithms for more than 20% for the HR. For the BHR, LFU and LFU-DA are the best choices for a very small cache, although the performance is similar for all policies while the cache size increases. Under the packet cost model, GD* is the best choice to maximize the HR, while all policies, except LRU and GD-Size, provide the same good performance for the BHR. For this content-type, both metrics can not be optimized simultaneously for cache sizes less than 10%. For a bigger cache size the cost replacement policies outperform HR and BHR.

The Image content-type performance study is represented in Figure 5. This figure shows that the cost replacement policies maximize the HR in both cost models, although the performance is slightly better for the constant cost model. If we consider the BHR, there are tiny differences between policies, but LFU is the best choice for cache size greater than 10% and GDSF is the best for smaller cache sizes.

Figure 6 illustrates the performance evaluation for the Text content-type and it shows similar characteristics to the Image content-type for the HR. The GD* and GDSF algorithms obtain the best performance for all cache sizes and both cost models. The LFU-DA algorithm seems to be the best choice to maximize the BHR, although GDSF(packets) obtains similar results. For the Text content-type, the GD-SIZE algorithm is the worst for both metrics.

Finally, Figure 7 depicts the performance evaluation for the Video content-type. Only the constant cost model results have been depicted because both models obtain the same performance. The figure shows that, except for the LFU algorithm, the other policies present the same behavior for both metrics and cost models, so any of them could be a good choice. There is little differences between replacement policies due to the fact that the percentage of references is only 0.12% and the document sizes versus number of references is very sparse, therefore cost policies can not obtain optimal results.

5. Conclusions

In this paper, an exhaustive study of the different content-types of HTTP traffic for Web documents has been developed. To perform this study a workload trace from a real proxy cache has been characterized obtaining its main statistic properties. For each content-type, a deep study of document size

characteristics, popularity and temporal correlation has been presented. Six replacement policies have been presented: three classical schemes (LRU, LFU, LFU-DA) and three size-based schemes specifically developed for Web caches (GD-SIZE, GDSF, GD*) using two cost models (constant cost and packet cost). Then, a simulation driven study of the performance of these replacement policies for each content-type of Web documents has been performed. From this study some conclusions can be derived:

- Applications: The three cost algorithms with constant cost model maximizes the HR, but GDSF(p) and GD*(p) maximizes the BHR.
- Audio: For the HR, GD-SIZE(1), GDSF(1) and GD*(1) are good choices. To maximize the BHR, all policies give the same performance, except LFU that works better for a small cache size.
- Images: GDSF(1) and GD*(1) outperform the other policies for the HR. To maximize the BHR GDSF(p) and GD*(p) are the best choice for small caches (less than 10%) and LFU is better for bigger ones.
- Text: GDSF(p) maximizes both the HR and the BHR.
- Video: All replacement policies considered except LFU obtain the same results for HR and BHR.

As it can be observed in the previous summary, there is no a replacement policy that outperforms the others for all content-types, so to develop a proxy cache that distinguishes the content-types of documents, the best algorithm for each content-type should be applied. Another consideration to take into account is the metric that we need to maximize, since, except for the Text and Video content-types, the best algorithm for each metric differs.

6. Acknowledgements

We would like to thank Duane Wessels for the access to the workload traces.

This work was partially supported by the public Project N^{er} TEL2003-07953-C02-01

7 References

1. Luotonen, A., Altis, K.: World-Wide Web Proxies, Proceedings First International Conference on the WWW (1994)
2. Cao, P.: Cost-Aware WWW Proxy Caching Algorithms, Proceedings USENIX Symposium on Internet Technologies and Systems (1997)
3. Krishnamurthy, B., Wills, C.E.: Proxy Cache Coherency and Replacement – Towards a More Complete Picture, Proceedings of the IEEE International Conference on Distributed Computing Systems (1999)
4. Busari, M.: Simulation of Web Caching Hierarchies, Pd. Thesis (2000)
5. Khayari, R.A., Best , M.., Lehmann, A.: Impact of Document Types on the Performance of Caching Algorithms in WWW Proxies: A Trace Driven Simulation Study, 19th IEEE International Conference on Advanced Information Networking and Applications (2005)
6. http://www.ircache.net
7. http://www.squid-cache.org
8. Zhang , X.: Cachability of Web Objects, Technical Report 2000-19, (2000)
9. http://www.iana.org/assignments/media-types
10. Crovella, M.E., Bestavros, A.: Self Similarity in the World Wide Web Traffic: Evidence and Possible Causes, Proceedings IEEE/ACM Transactions on Networking, Vol. 5, Nº 6, (1997) 835-846
11. Arlitt, M., Williamson, C.: Web server workload characterization: The search for invariants, Proceedings of the ACM SIGMETRICS'96 Conference (1998)
12. Breslau, L., Cao, P., Fan, L., Phillips, G.: On the Implications of Zipf's Law for Web Caching, 3rd International WWW Caching Workshop (1998)
13. Breslau, L., Cao, P., Phillips, P., Shenker, S.: Web Caching and Zipf-like Distributions: Evidence and Implications, IEEE Infocom, Vol XX (1999)
14. Jin, S., Bestavros, A.: Temporal Locality in Web Request Streams. Sources, Characteristics and Caching Implications, Poster proceedings ACM SIGMETRICS 2000 Conference (2000)
15. Nagaraj, S.V., Web Caching and its Implications, Kluwer Academic Publishers (2004)
16. Arlitt, M., Williamson, C.: Internet Web Servers: Workload Characterization and Performance Implications, IEEE/ACM Transactions on Networking (1997)
17. Robinson, J.T., Devarakonda, M.V.: Data Cache Management Using Frequency-Based Replacement, Proceedings 1990 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems, (1990) 134-142
18. Cherkasova, L.: Improving WWW Proxies Performance with Greedy-Dual-Size Frequency Caching Policy, Technical Report HP Labs HPL-98-69 (1998)
19. Jin, S., Bestabros, A.: GreedyDual* Web Caching Algorithm: Exploiting the Two Sources of Temporal Locality in Web Request Streams, Intl' Journal of Computer Communications, Vol. 24, No. 2, (2001) 174-183
20. Arlitt, M., Friedrich, R., Jin. T.: Workload Characterization of a Web Proxy in a Cable Modem Environment, Technical Report HPL-1999-48, Hewlet-Packard Laboratories (1999)

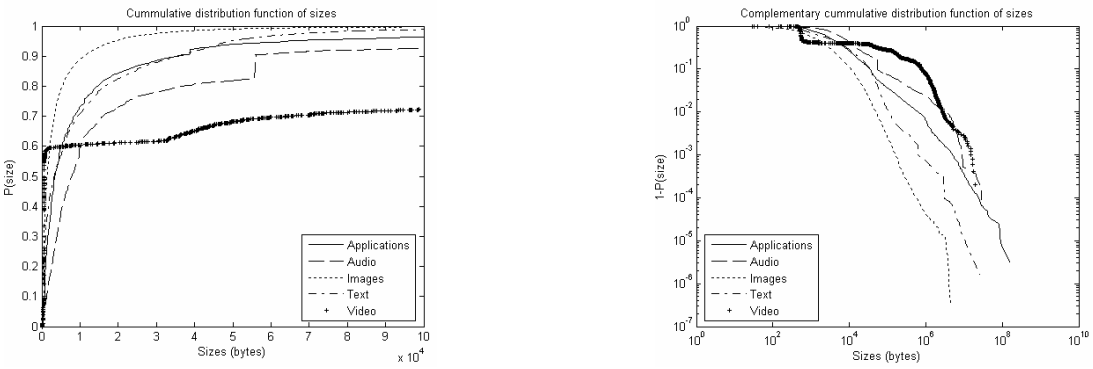

Figure 1. Cumulative distribution function of document size (left) and complementary cumulative distribution function of document size by content-type (right), as a function of the content-type.

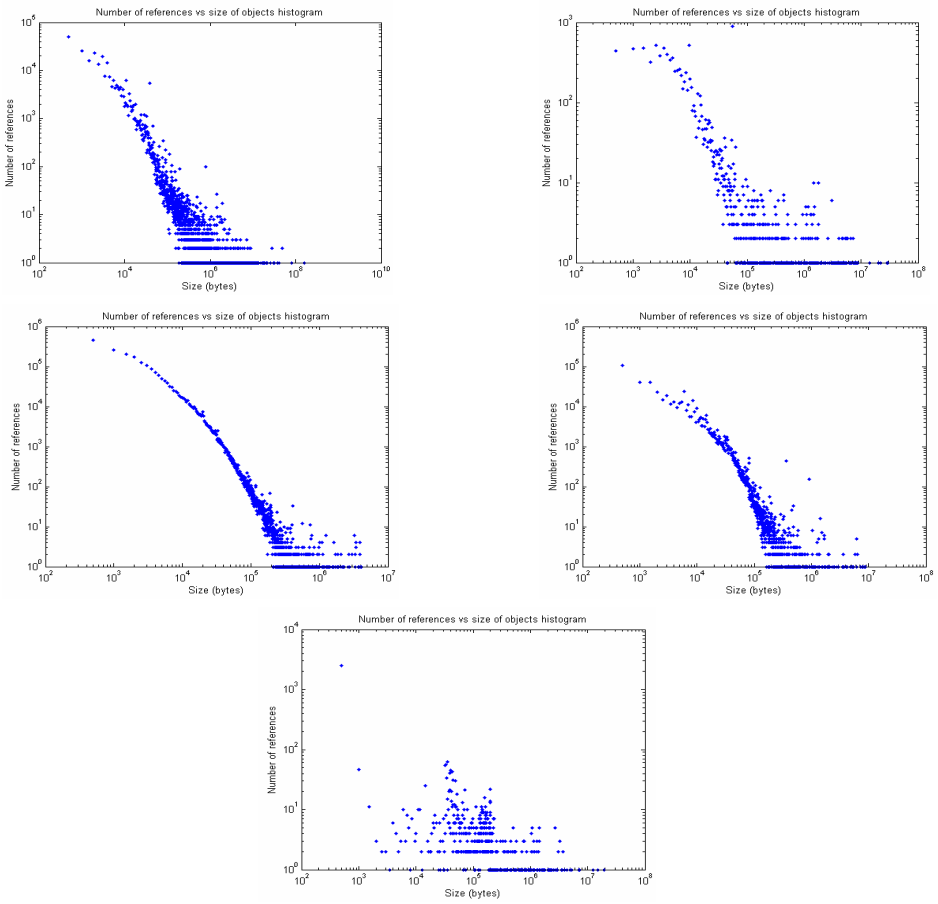

Figure 2. Popularity vs. document size. Application (top left), Audio (top right), Images (center left), Text (center right) and Video (bottom)

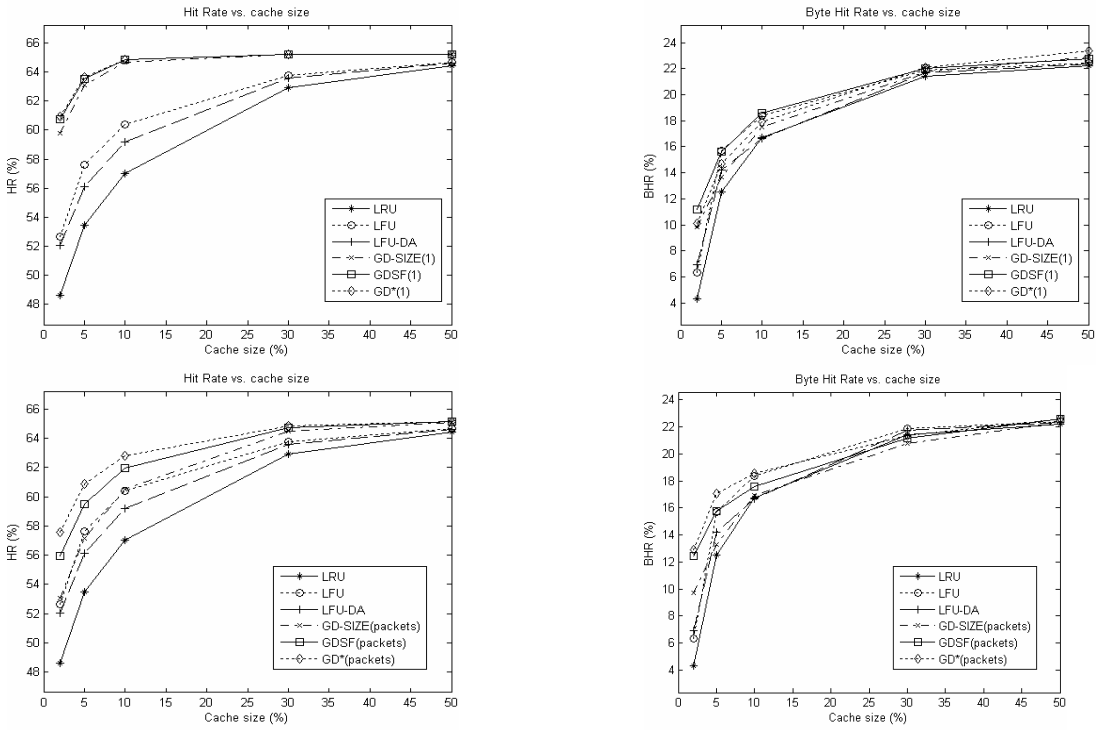

Figure. 3. Evaluation of cache replacement policies for Application content-type. Evaluation under constant cost function (top) and packets cost function (bottom).

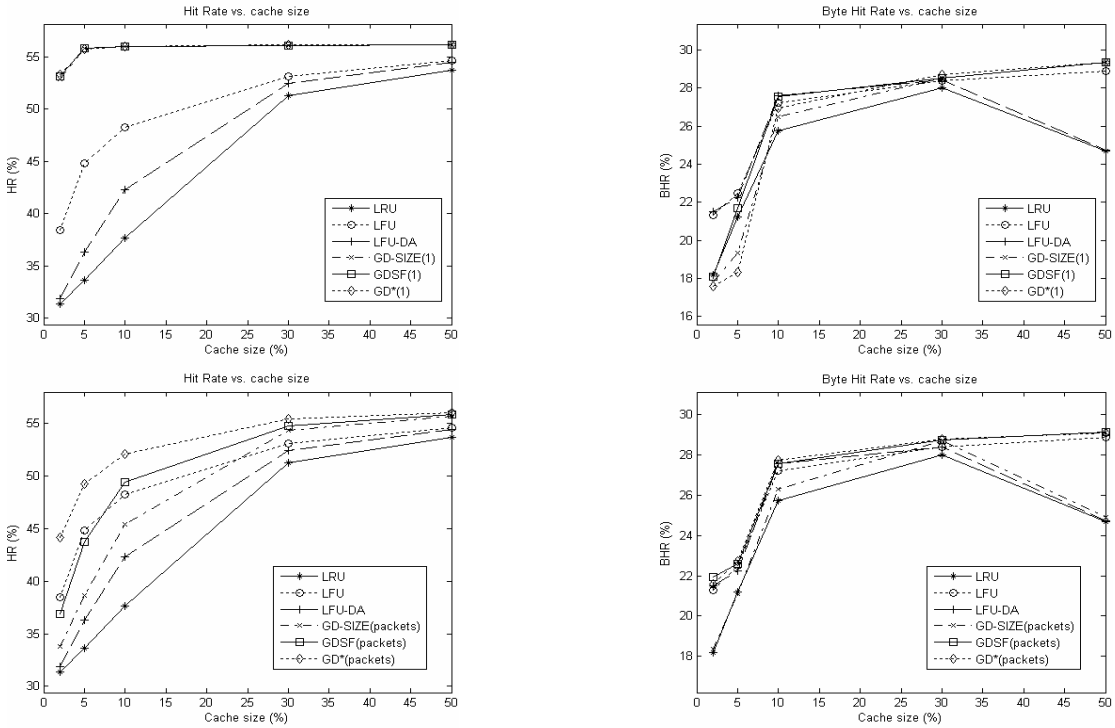

Figure. 4. Evaluation of cache replacement policies for Audio content-type. Evaluation under constant cost function (top) and packets cost function (bottom).

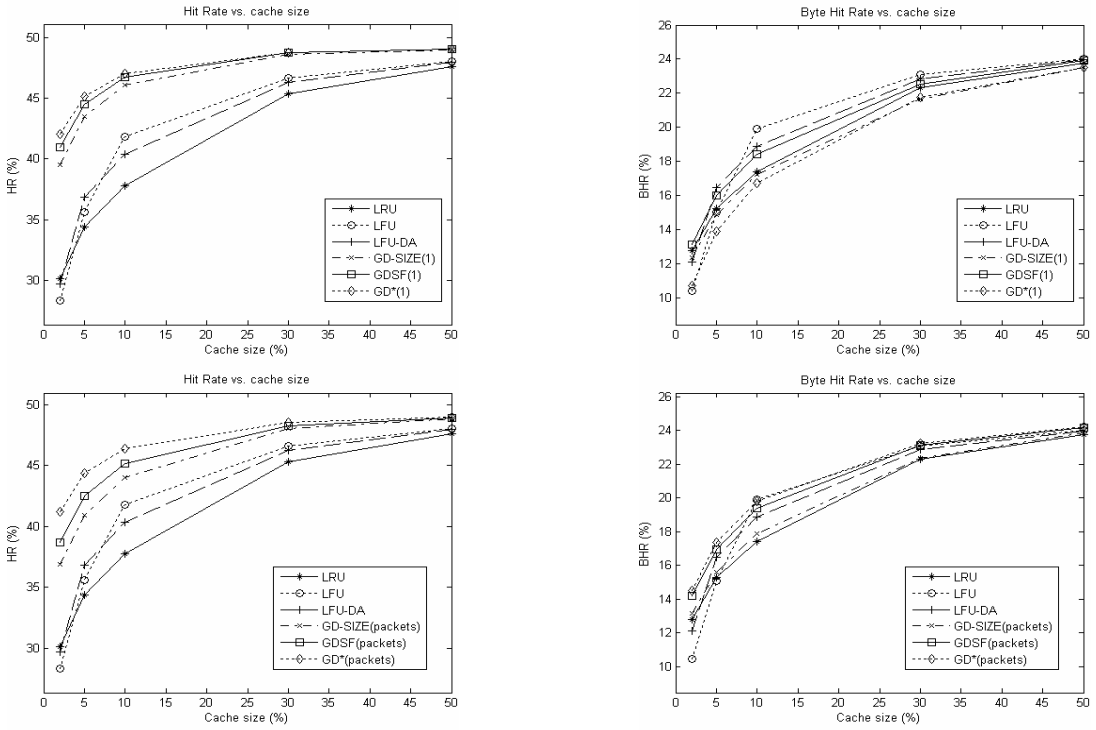

Figure 5. Evaluation of cache replacement policies for Image content-type. Evaluation under constant cost function (top) and packets cost function (bottom).

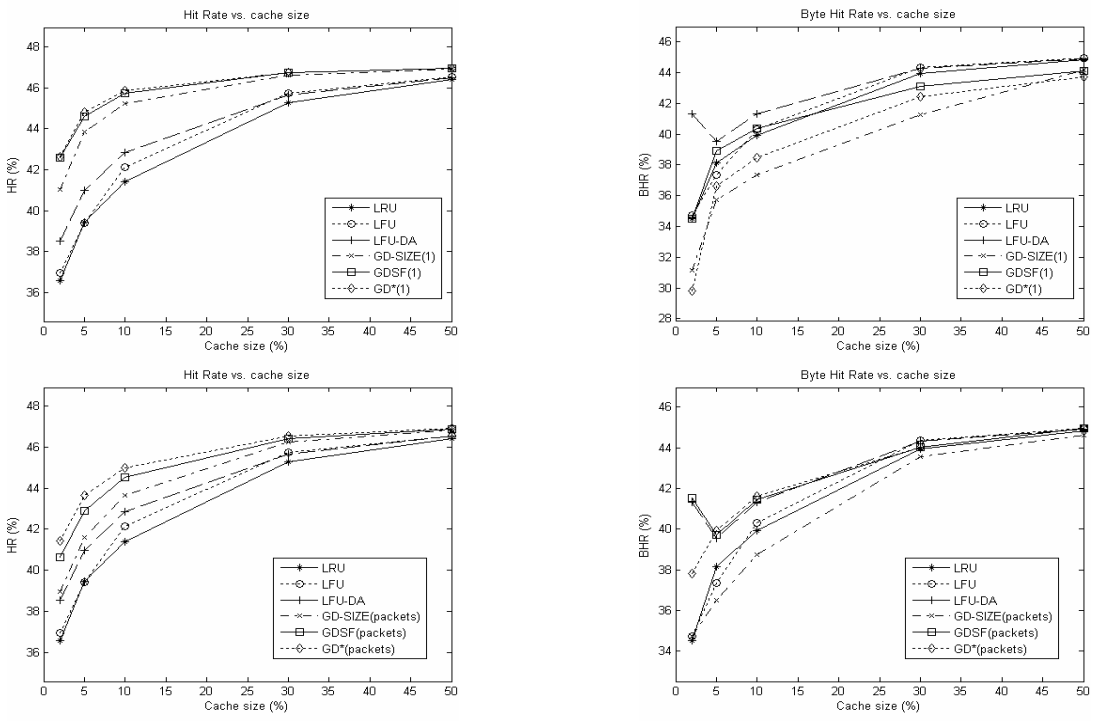

Figure 6. Evaluation of cache replacement policies for Text content-type. Evaluation under constant cost function (top) and packets cost function (bottom).

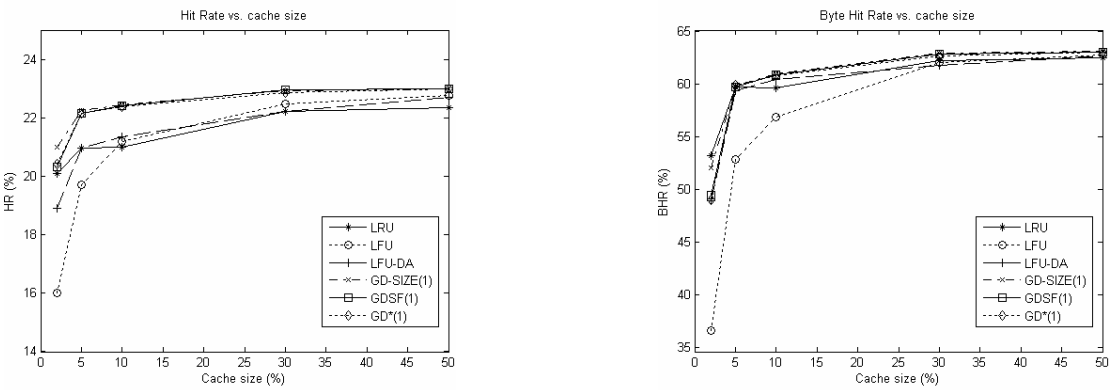

Figure. 7. Evaluation of cache replacement policies for Video content-type. Evaluation under constant cost function.