

Concurrent Evaluation of Web Cache Replacement and Coherence Strategies

A.S.Z. Belloum

University of Amsterdam
Kruislaan 403
1098 SJ Amsterdam, the Netherlands
adam@science.uva.nl

L. O. Hertzberger

University of Amsterdam
Kruislaan 403
1098 SJ Amsterdam, the Netherlands

When studying Web cache replacement strategies, it is often assumed that documents are static. Such an assumption may not be realistic, especially when large-size caches are considered. Because of the strong correlation between the efficiency of the cache replacement strategy and the real state of the cached documents, one may expect dramatic changes of the cache performance when there is a possibility of hits on out-of-date documents. The authors' aim in this study is to outline using simulation the impact of some cache coherence techniques such as the invalidation protocol, the TTL_based strategies, and prefetching on the efficiency of the cache replacement strategies. Using a simple cache model, they will show that the cache hits recorded when using some replacement strategies are more likely to be performed on out-of-date documents. They will show also the impact of the workload used on the outcome of the simulation.

Keywords: Web caching, cache replacement, cache coherence, simulation modeling

1. Introduction

The problems in Web caching result from the fact that we can never dispose of an infinite memory space and thus the cache manager has to start removing sooner or later documents from its local storage to make room for newly requested ones. Although the problem of selecting the next document to be removed has been widely discussed in the literature [1–3], it is still not clear yet which one of the most commonly discussed strategies leads to the best performance. In a previous study, we pointed out that if the documents are considered static, whatever the replacement strategy used, we get the same performance when assuming a relatively small Web cache size (512 MB) [4]. But as one may expect, this is a very optimistic result; in real situations, the documents are not static—they could be modified at any time and without any warning by their owners. This means that part of the documents forwarded from the cache to the end-users are out of date. Regarded from this point of view, caching does not only imply storing the maximum number of documents in the cache but also the documents that are more likely to be up to date.

The document lifetime, which seems to have a major importance in the evaluation of the cache performance, is unfortunately impossible to predict with precision. Often, estimates of this parameter are used, and the accuracy of these estimates is yet under discussion. In some studies, the uniform distribution is assumed good enough to approximate accurately the document-updating process as it was suggested by Liu and Cao [5], while in other studies a multimodal distribution is proposed as a better approximation [6].

Several techniques have been proposed to optimize the document-replacement process within simple and proxy caches [5, 7–9]. In most of these studies, the cache size has been identified as having always a positive impact on the cache performance. The common characteristic of all these studies is the fact that they all assume that cached documents are static. For the studies that focus on the Web cache coherence, the number of stale hits performed by these replacement strategies could be unacceptably high [6, 10]. These studies propose different techniques to solve the problems of the stale hits. Two categories of methods have emerged as a potential solution for the problem of Web cache coherence—the *strong cache coherence* [11] and the *weak cache coherence* [10]. The cost and the efficiency of the two categories are still a controversial issue. The strong coherence strategies guarantee that no stale hits are performed by the cache server, while the weak strategies minimize the probability of these stale hits. The replacement

Submission Date:

Accepted Date:

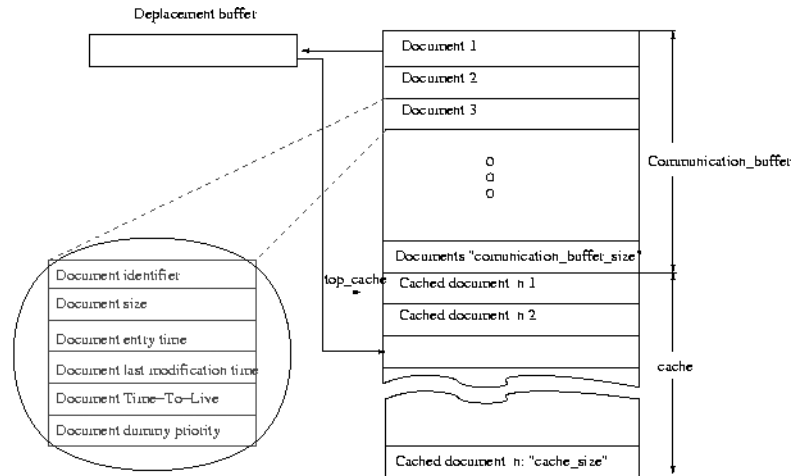


Figure 1. Information stored in the cache model

strategies used with a weak cache coherence while increasing the hit ratios of the cache server may also increase the stale hits. If weak cache coherence is considered, the replacement strategies should be compared not only on the increase of the hit ratios but also on the quality of these hits.

In general, Web caching is initiated by the end-user requests, some proposed methods suggest that Web caches should be also active in the process of the dissemination of the documents. The motivation behind this approach is the fact that cache servers have access to both the access history and the state of the cached documents; the cache can thus perform some statistics to predict network traffic. Methods such as push-caching [12] or document refreshing (prefetching) [11] are simple examples of server-initiated strategies.

Using a simple simulation model and real workloads, we compare in this article five replacement strategies, namely, the LRU (the least recently used document is removed first), the LFU (the least frequently used document is removed first), the SIZE-based (the large-size document is removed first) [7], the WM (the weighted method combines several parameters to choose the next document to be removed) [1], and the NNC (a method derived from the nearest neighbor classifier that uses the similarity between two documents to select the document to be removed) [4].

The rest of the article is organized as follows: In Section 2, we present the model of the Web cache we are proposing to approximate the cache behavior. In Section 3, we discuss the impact of both the cache size and the cache coherence on different document replacement strategies. Section 4 deals with the impact of prefetching on the cache performance. Finally, Section 5 concludes this study.

2. The Simulation Model

To compare the document replacement strategies discussed in the previous section, we have built a simple model of a

Web cache. The model uses a heap as the priority data structure to store the cached documents. When a document is cached, it is assigned a priority number, which allows it to be inserted at the right position in the heap. The priority number assigned to each document depends on the cache replacement strategy being used. This approach has allowed us to use the same implementation of the cache model for all the cache-replacement strategies. For performance reasons, we do not store the real documents in the cache, only the size of the cached documents is stored. For each document, the following header is kept in the heap: the identifier, the size, the entry time, the last modification time, and the time-to-live (Fig. 1). Before entering the heap, the documents are kept in a buffer where the header is created. The three first fields of the header are extracted from real access log files, while the two last fields, if not defined, are assigned values from predefined tables for the time-to-live or generated using the probability distributions discussed in Section 2.2. The warm-up phase is identified either when a cache becomes full for the first time or when the hit rate remains within a predefined interval as it was suggested by Arlit and Williamson in [7].

2.1 The Main Components of the Model

Figure 2 shows the simulation platform we have implemented to study the behavior of the cache. One part of this model, called *Web cache server*, implements the main functions of a cache server, mainly the replacement and the cache coherence strategies. The other part, called *trace generator*, generates the requests forwarded to the cache and receives messages from the cache to check the state of the documents. It keeps track of all the requests it has generated in the past. Besides generating the requests to be sent to the Web cache server, the trace generator is in charge of the document-updating process. It includes a periodic process that marks each document as modified to

simulate document updates. The document remains in this state until a “get-if-modified” is received from the cache; at this time, both the trace generator and the Web cache server have the same version of the document and then the invalidation flag is turned off. The trace generator plays the role of the origin server for each document included in the log-file. The trace generator introduces two important factors that could indeed impact the results of the simulations—the distribution used to update the documents and the frequency at which they are updated.

2.2 Simulating Document Updates

The distribution that approximates the document-updating process is quite difficult to define. Some Web traffic analyses stated that documents may remain unchanged for a long time and then start being modified frequently within a short period of time [13]. It has also been shown in other studies that popular documents have small size [9]. Therefore, we can assume that the age and the size of the documents can be used to approximate the document-updating time.

Heuristic for the document updating process. If large-size documents are not frequently modified (most of these documents are images, audio files, or video streams), and if very small documents are likely to be static in regard to the statistics [9, 14], we can assume that the document-updating process could be approximated by a normal distribution on the size of the documents. Thus, only documents belonging to the interval defined by the standard deviation are more probable to be modified.

We have added to the trace generator a process that periodically changes the last modification time of one of the documents contained in the access log-file. The new process, called the *document-updating process*, has two main parameters: the frequency of polling the list of documents and the probability distribution used to select the documents from the list. In the experiments presented in this article, we have used two probability distributions, namely, the uniform and the normal distributions.

2.3 The Frequency of Document Updates

The second parameter of the document-updating process is the frequency at which the documents are updated. It can also have a great impact on the accuracy of the simulations. This parameter should be tuned to reflect a typical document update frequency on the Internet. In our model, the trace generator is simulating updates of documents on behalf of a large number of Web servers. The document-updating process is performed in a separate process executed in parallel with the process generating requests sent to the cache. The relationship between the two processes could be fixed before starting the simulation. For instance, if the same frequency is used for both the incoming requests and the document-updating process, we have found out that it implies that a document is modified every 4 seconds, whereas polling the updating list at a frequency equal to 1/100 of the incoming requests leads to one docu-

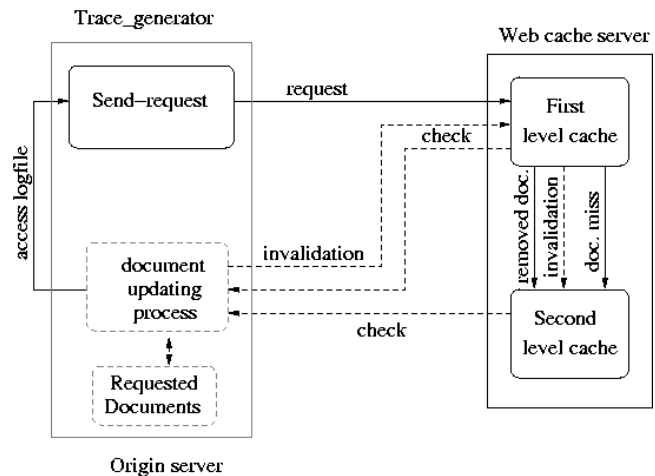


Figure 2. Web cache model

ment update every 100 seconds. The probability of selecting a document when using the first polling frequency is equal to 3.3%, while this probability decreases to 0.1% with the second one. This range of frequencies overlaps with the one presented in Web traffic analysis performed by Bestavros [9], where it is stated that the average document update probability is between 2.5% and 0.5%.

2.4 Performance Metrics

The performance metrics used in this Web cache analysis focus on the document hit-ratio and the byte hit-ratio. These two metrics are the most commonly used in the literature. Because this article does not aim at studying the cache replacement strategies, only the document hit-ratio “DHR” is used. In this article, we focus mainly on the quality of the cache hits, that is, the percentage of hits performed on up-to-date documents. This DHR records the hit ratio obtained for a predefined cache configuration and it reports the percentage of hits over the total received requests.

2.5 The Workloads

The workloads used in this study are part of access log-files provided by the Web server of the Computer Science Department at the University of Amsterdam WINS (wins.uva.nl), and the proxy server NLANR (ircache.nlanr.net). These traces, of which some characteristics are shown in Table 1, present two types of workloads. The WINS workload contains external requests to documents provided by the wins.uva.nl server. This workload exhibits a strong document locality of references. The NLANR workload involves requests coming from different Web servers that use NLANR as a proxy.

3. The Experiments

In this section, we present some of the experiments we have performed to point out the relationship between the

Table 1. Workload characteristics

Workload	Duration	Transferred Data	Number of Requests
WINS	1 month	4.2 GB	737,750
NLANR	1 day	2.5 GB	261,135

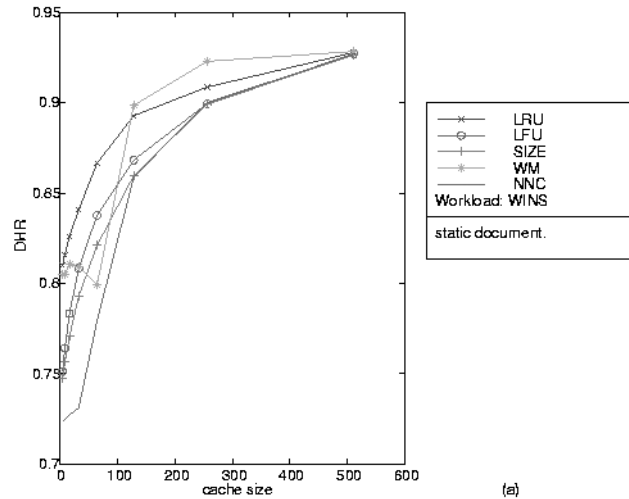
cache replacement strategies and the methods used to maintain the cache coherence. The experiments compare the cache hits recorded with five cache replacement strategies when both the weak and the strong cache coherences are considered. The parameters modified in these experiments are the cache size, the document updating distribution, and the input workload.

3.1 Maintaining Cache Coherence Using TTL

The TTL-based strategy is one of the methods that aims only for a weak cache coherence. Each document is assigned a TTL_value (time-to-live), and unless this time elapses, the document is considered up to date [11]. Even if the HTTP protocol has provided a new header field where it is possible to specify the expiration date for each document, the usage of the optional field is still very low. We have thus used the setting of the TTL_values defined by the Harvest Cache where the TTL_values are fixed according to the document type [6].

Figure 3 represents the hit ratio recorded for all the cache replacement strategies when the documents are considered static. In this case, the increase of the size of the cache can only have a positive impact on the hits. When more space is provided, more documents can be stored and thus more hits are recorded.

A completely different behavior is shown when the documents are not static. Figure 4 shows a dramatic change in the outcome of the simulation results. When considering the possibility of caching out-of-date documents (Fig. 4(a)), increasing the size of the cache does not lead to higher cache hits. A large percentage of the recorded hits are in fact hits on out-of-date documents. When increasing the cache size, the hit ratios recorded for the five replacement strategies show a transient phase for small-size caches. For these caches, the hits either drop dramatically to reach a minimum value (LRU, SIZE-base, and WM) or increase then drop (LFU and NNC). Depending on the replacement strategy, the important transitions of the hits are recorded for cache values varying between 32 MB and 256 MB. Even if the document-replacement strategies presented in this study are based on different parameters that represent different characteristics of the cached documents, similar behavior of the cache has been recorded—small-size caches perform, in general, higher hit ratios. The uniform approximation of the time the documents are updated has increased the percentage of hits on out-of-date documents, which is likely to happen on large-size caches. The document-updating process can have an important impact

**Figure 3.** The document hit ratio when considering static documents

on the outcome of the simulation. To show this impact, we have repeated the same experiments using the normal distribution (Fig. 4(b)).

This time, only the SIZE-based strategy, the WM, and the NNC have significantly decreased the hits for large-size caches, but still the highest hits are recorded for small-size caches. The normal distribution, which updates more frequently relatively small-size cached documents, has affected only the replacement strategies that use the size when removing the documents. In the current setting of these strategies, large-size documents are removed first, thus more documents are stored in the cache and for a longer period of time, which has increased the hits on long-term cached documents.

We have performed the same experiments using another workload with a very low locality of references. This workload contains a large number of documents that are requested only once; most of the cache-replacement strategies cannot prevent the one-timer documents to enter the cache. These documents are often stored instead of more popular documents [15]. With the new workload, higher hits rates are shown for large-size caches (Fig. 5). Due to the large number of one-timer documents, around 78% of the workload, the document-updating process had only a minor impact on the cache hits.

In this experiment, the replacement strategies that have considered the time the documents entered the cache (the LRU and the NNC) show higher hits. With the LFU strategy, the few popular documents, around 20% of the workload, have been kept for a long time in the cache, which has reduced the hits for large-size caches. The SIZE-based and the WM strategies, which focus more on the size of the documents, are the only strategies that allow the cache hits to increase for large-size caches.

The experiments presented in this section have shown that when considering hits on out-of-date documents, the

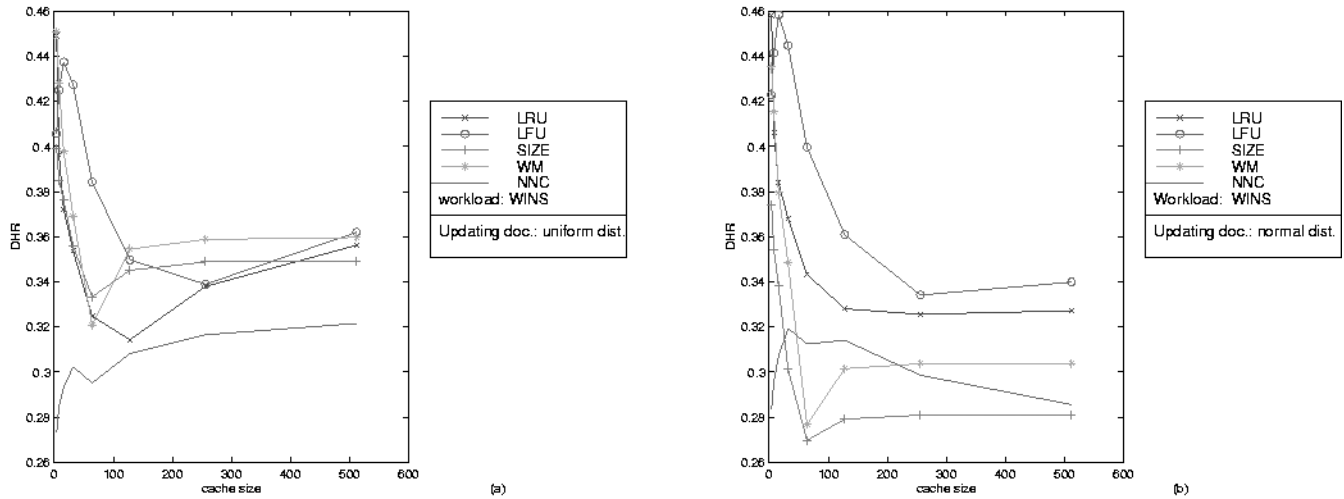


Figure 4. The DHR recorded a weak cache coherence (WINS workload)

number of hits performed by the cache depends very much on the nature of the traffic traversing the cache. One outcome of these experiments is that caches submitted to a traffic having a strong locality of reference do not need to be very large, while the caches submitted to a traffic having a weak locality of reference can still produce better hit ratios when their local storage increases. It is also shown in these experiments that some cache-replacement strategies are more likely to perform hits on long-term cached documents, and these strategies do not use the time the documents have been stored as a parameter when removing documents.

3.2 Maintaining Cache Coherence Using Invalidation Protocol

The invalidation protocol allows for maintaining a strong cache coherence; it is initiated by the origin servers (the servers that contain the original copies of the cached documents). This method has the disadvantage of overloading the origin servers and generating a large amount of useless traffics; these side effects are not addressed in this study (see [16] for more information on this topic). Our interest in this article is to measure the impact of the invalidation protocol on the cache-replacement strategies for different values of the cache size. By reproducing the same conditions of the experiments discussed in Section 3.1, we will be able to measure the percentage of false hits performed using the TTL_values to maintain the cache coherence. This is possible because the invalidation protocol does not allow any false hit. What one can expect is an increase in hit ratios, especially for large sizes of cache. The invalidation protocol updates the content of the large caches where documents can remain cached for a long period of time. One cache miss on a specific document introduced by the invalidation protocol can cause a number of correct hits on the subsequent access to that document.

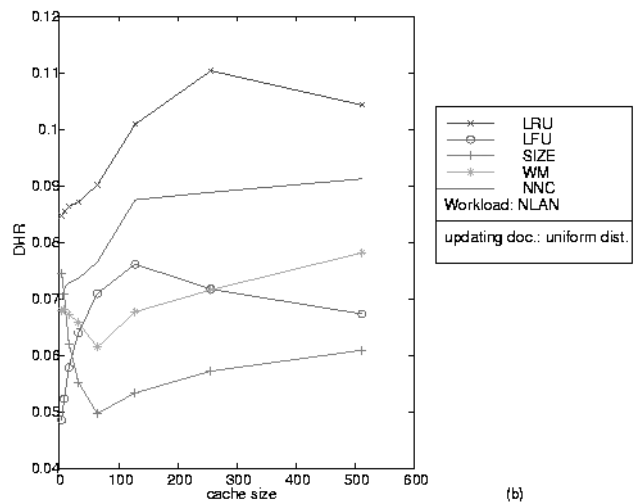


Figure 5. DHR recorded using a weak cache coherence (NLAN workload)

The cache hit ratios shown in Figure 6(a) confirm these facts. When these ratios are compared to the results obtained without invalidation (Fig. 4(a)), a clear increase in the hit ratios of large cache size is recorded. When we use the normal distribution, two types of hits are outlined (Fig. 7); the LRU and LFU replacement strategies allow the cache to record the maximum hit ratios when small-size caches are considered, while the other replacement strategies show higher hit ratios for large-size caches. For most of the replacement strategies, a cache of 64 MB seems to be the point where the number of hits is likely to change dramatically. The invalidation protocol can also be less efficient on large-size cache configurations, a phenomenon we could not clearly see with the uniform distribution.

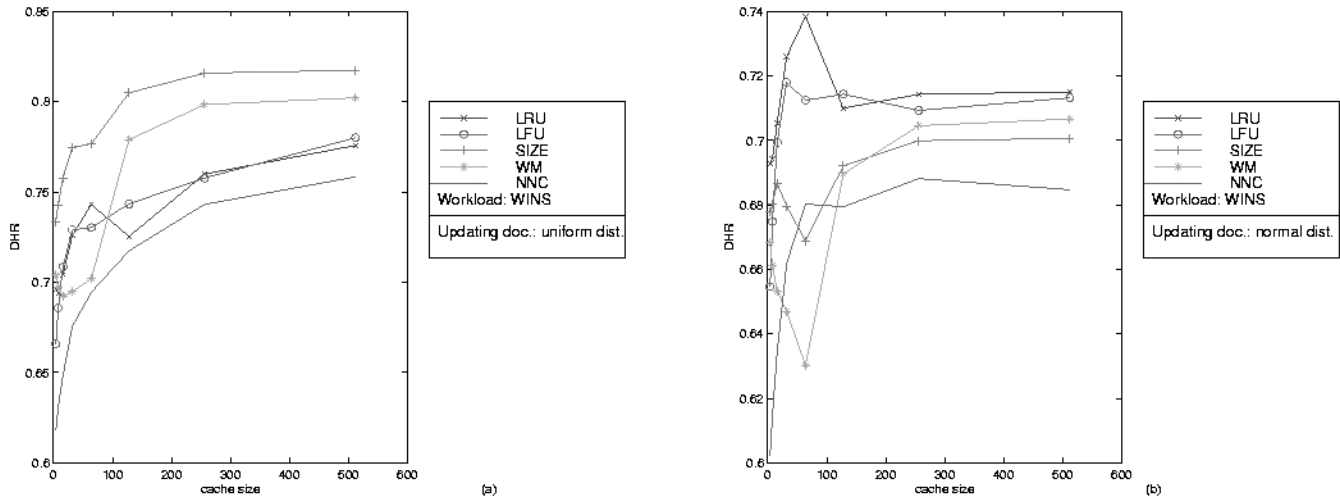


Figure 6. The DHR recorded using a strong cache coherence (WINS workload)

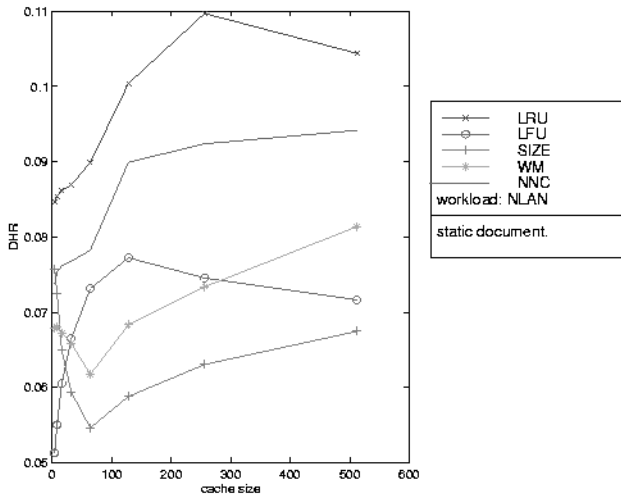


Figure 7. The DHR recorded using a strong cache coherence (NLAN workload)

If a workload with a weak locality of reference is used, the invalidation protocol does not have a significant impact on the cache hits. The evolution of the hit ratios is quite similar to the one shown in the previous experiment. The rate of documents successfully invalidated is less than 10% for large-size caches [16]. This means that a large number of invalidation messages are sent to the cache while the documents have been already removed. Besides that, because of the large number of one-timer documents contained in the workload, most of the invalidation messages could not produce the expected subsequent hits. A large number of the prefetched documents fall into the category of one-timer documents.

The efficiency of the invalidation protocol is tightly related to the workload used. The experiments presented here show an increase of the cache hits when using the invalidation protocol, but only when the workload has a strong locality of reference. This improvement in the cache hits was recorded for the five cache-replacement strategies and for all values of the cache size. However, applying the invalidation protocol when the traffic exhibits a weak locality of reference did not provide any improvement in the cache hits. Using the invalidation protocol when the cache is submitted to the traffic with a weak locality of reference will be equivalent in terms of hits to the weak document coherence presented in the previous section.

4. Improving the Performance Using Prefetching

In the previous section, we pointed out that increasing the number of hits does not always imply better performance. A hit on a recently received document is more likely to be a better one than a hit on a long-term cached document. One method to avoid false long-term hits is to prefetch some cached documents before they are really requested by the end-users. To avoid overloading the cache and the network traffic, prefetching can be done during the nights when the Internet traffic is low [1]. Documents that are out of date or will very soon become out of date are checked and if necessary prefetched. Prefetching allows for the increase in the probability of a good hit on long-term cached documents.

Figure 8 shows the impact of documents prefetching on the hit ratios. Better performances are achieved for all the replacement strategies, especially for large-size caches for which the hit ratios are again very high. We recorded almost the same hit ratios as if we were to consider static documents. The maximum cache hits are recorded for large-size caches; the monotonic increase of the number of

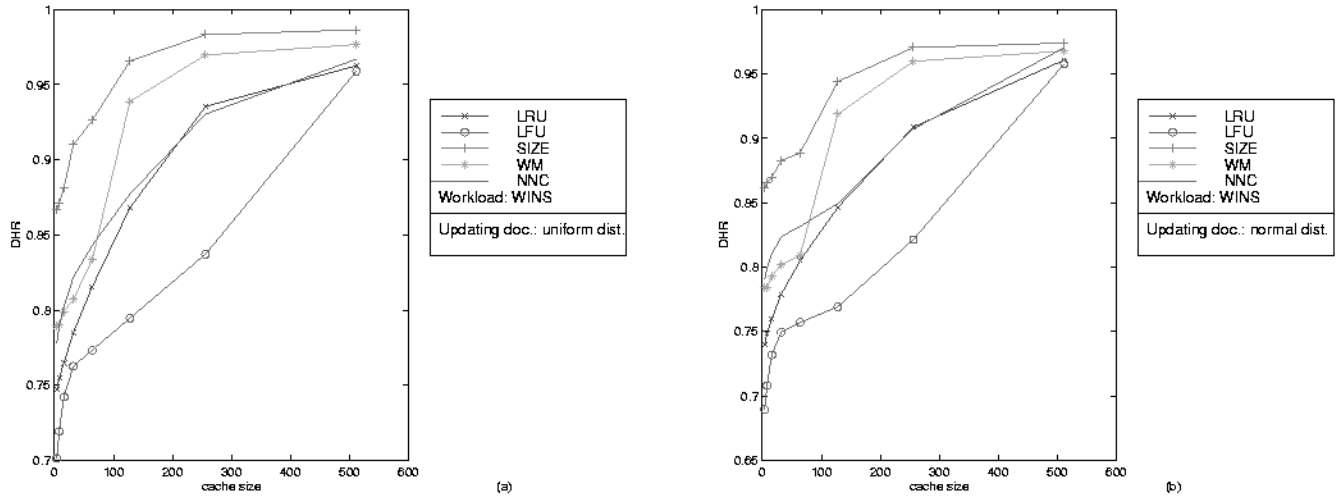


Figure 8. The DHR when prefetching is used (WINS workload)

hits with the increase of the size of the cache suggests that hits on long-term cache are no longer false hits, which proves that the pre-fetching mechanisms succeeded in keeping this category of documents up to date. The document-updating process does not have any more significant impact on the cache hits; similar hit ratios have been recorded for both of the two distributions (uniform and normal). Prefetching has improved dramatically the hits when the SIZE-based and the WM strategies are used that outperform all the other strategies; this suggests that with the SIZE-based strategy and the WM, the cache performs most of its hits on long-term cached documents.

The use of the workload with a weak locality of reference did not reduce the impact of the prefetching method. Figure 9 shows an important increase of the cache hits. Once again, the SIZE-based and the WM strategies have benefited the most from prefetching. Actually, more documents have been prefetched when the SIZE-based and the WM strategies are used, because both remove large-size documents first. However, the number of successfully prefetched documents remains still quite high.

5. Conclusions

Improving the Web cache performance does not imply only increasing the hit ratios, but it also involves improving the quality of the hits. Studying the document-replacement strategies without considering the cache coherence yields the same quality for all the cache hits when different replacement strategies are used. We have shown in this study the existence of two categories of replacement strategies—one that performs its hits on recently requested documents such as the LRU and others that perform the major fraction of the hits on long-term cached documents. For the first category, large cache size will contain a large number of documents on which only few hits are recorded. For the

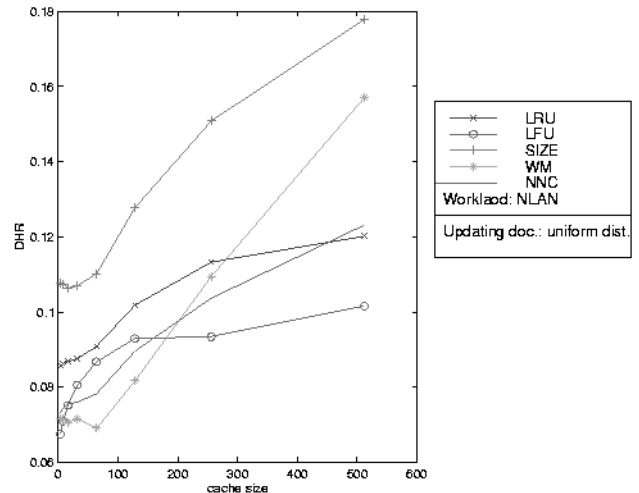


Figure 9. The DHR when prefetching is used (NLAN workload)

second category, the larger the cache, the better the hit ratios. However, it is likely that a large number of the hits are performed on out-of-date documents. In some of the experiments presented in this article, it is shown that having large-size caches does not lead to the highest hit ratios. Under certain conditions, small-size cache behaves a lot better. We have also shown that the outcome of the simulation shows different behavior if the workload presents different characteristics; for instance, a workload with a strong locality of reference can be better handled by small cache size, while a workload with a weak locality of reference can still benefit from large-size caches.

The quality of the hits can be improved using a very simple method that consists of prefetching the cached

documents when they become out of date before they are requested by the end-users. The results of these experiments show a dramatic increase in the hit ratios when large-size caches are considered. These experiments have confirmed the difference in quality of the hit ratios we pointed out in the first set of experiments. In the last experiments, the replacement strategies that have benefited the most from the documents prefetching are those that were reported as recording a large number of hits on long-term cached documents.

6. Acknowledgments

We thank Ercin Kaletas, Mourad Boulkroune, and Tarek Belloum, for their many comments and suggestions on this article, and Henk Muller of the University of Bristol for his constructive criticism throughout the project. This work was supported by the HPCN Foundation through the project JERA grant 96150. We thank also the National Science Foundation (grants NCR-9616602 and NCR-9521745), and the National Laboratory for Applied Network Research for providing access to data used in the experiments.

7. References

- [1] Bolot, J., and P. Hoschka. 1996. Performance engineering of the World Wide Web: Application to dimensioning and cache design. Paper presented at the Proceedings of the Conference on Computer Networks and ISDN, pp. 1397–405, May, Paris.
- [2] Abrams, M., C. R. Standridge, G. Abdullah, E. A. Fox, and S. Williams. 1997. Removal policies in network caches for World-wide Web documents. Paper presented at the Proceedings of the ACM SIGCOMM, pp. 293–305, August, Stanford, CA.
- [3] Cao, P., and S. Irani. 1997. Cost-aware www proxy caching algorithms. Paper presented at the Proceedings of the USENIX Symposium on Internet Technologies at Systems, pp. 193–206, December, Monterey, CA.
- [4] Belloum, A., A. Peddemors, and L. Hertzberger. 1998. Jera: A scalable Web server. Paper presented at the Proceedings of the Conference PDPTA 98, pp. 167–74, July, Las Vegas, NV.
- [5] Liu, C., and P. Cao. 1997. Maintaining strong cache consistency in the World-wide Web. *IEEE Transaction on Computers* 47(4):447–57.
- [6] Gwertzman, J., and M. Seltzer. 1996. World-Wide Web cache consistency. Paper presented at the Proceedings of the USENIX Technical, January, San Diego, CA.
- [7] Arlitt, M. F., and C. L. Williamson. 1997. Trace driven simulation of document caching strategies for Internet Web servers. *SIMULATION: Transactions of The Society for Modeling and Simulation International* 68(1):23–33.
- [8] Belloum, A., and L. Hertzberger. 1998. Simulation of a two level cache serve. Technical Report CS-98-01, Computer Science Department of the University of Amsterdam, Amsterdam, the Netherlands.
- [9] Bestavros, A. 1995. Demand-based document dissemination to reduce the traffic and balance load in distributed information system. Paper presented at the Proceedings of the IEEE Symposium on Parallel and Distributed Processing, October, pp. 338–45, San Antonio, TX.
- [10] Liu, Z. 1997. Static caching of Web servers. Paper presented at the Proceedings of the ACM SIGCOMM, pp. 293–305, September, Cannes, France.
- [11] Dingle, A., and T. Partl. 1996. Web cache coherence. Paper presented at the Proceedings of the Conference on Computer Networks and ISDN, pp. 907–20, May, Paris.
- [12] Gwertzman, J., and M. Seltzer. 1995. The case for geographical push-caching. Paper presented at the Proceedings of the Workshop on Hot Operating Systems, pp. 51–5, May, Orcas Island, WA.
- [13] Gwertzman, J. 1995. Autonomous replication in wide-area distributed information. Technical Report TR-95-17, Harvard University Division of Applied Sciences, Center for Research in Computer Technology, Cambridge, MA.
- [14] Rousskov, A. 1999. A performance study of the squid proxy on http/1.0. *World Wide Web Journal* 2(1):47–67.
- [15] Belloum, A., and L. Hertzberger. 1998. Replacement strategies dedicated to Web caching. Paper presented at the Proceedings of the IEEE Conference ISIC/CIRA/ISAS 98, pp. 576–81, September, Gaithersburg, MD.
- [16] Belloum, A., and L. Hertzberger. 2000. Maintaining document coherence in a www environment. *Information Research Journal Special Issue on Web Research* 6(1). Available: <http://informationr.net/ir/6-1/paper91a.html>.

Adam S. Z. Belloum is an Assistant Professor at the Department of Computer Science at the University of Amsterdam. He received the MSc and PhD degrees from the Universite de Technologie de Compiègne, France. His interests include Web caching and distributed/collaborative modeling using grid technology.

Louis O. Hertzberger is a Professor in the Department of Computer Science at the University of Amsterdam. His research interests are in the field of parallel computing, embedded systems, intelligent autonomous robotics, and their application in industrial automation. He received a PhD in experimental physics from the University of Amsterdam. He may be reached at bob@science.uva.nl.