Document Replacement Policies dedicated to Web Caching

A. Belloum and L.O. Hertzberger
Computer Architecture and Parallel Systems Group
Dept. of Computer Science
Universiteit van Amsterdam
Kruislaan 403 NL 1098 SJ Amsterdam
Email: {adam,bob}@wins.uva.nl

ABSTRACT

Web caching has been considered as a powerful solution to deal with the growth of web traffic. Several studies have shown that caching documents throughout the Internet can save network bandwidth and reduce document access latency [9, 8, 10]. However, this technique has introduced new problems such as maintaining the document coherency and selecting the next document to be removed. With the continuous increase in demand for documents, the web cache servers are becoming the new bottleneck. A need for better resource management is becoming urgent in order to reduce the overhead sustained by web cache servers. In this paper, a number of web replacement policies are discussed and compared on the basis of trace-driven simulations. The impact of the web cache server configuration is pointed out through a set of experiments that use the cache size as a tuning parameter.

KEYWORDS: Web caching, simulation, modeling

1. INTRODUCTION

Web caching is widely used nowadays, almost all web browsers allow the end-users to chose their own web cache or *proxy* sever. At each request, the web browser checks first whether the configured proxy server has a valid copy of the requested document. In turn, the proxy server forwards the user request either to another proxy server or directly to the original remote server. A three level proxy servers hierarchy has been investigated in [4], the latency introduced by such a topology is relatively low. When a proxy server receives the requested document, it sends a copy to the end-user and keeps another copy in its local storage. Upon the next request to this document, the proxy server returns the copy to the end-user without connecting the remote server. Due to the limited storage space at the proxy servers, it is often necessary to remove previously cached documents before storing any newly requested documents. In this paper, the focus will be on the replacement policies used in web caching.

Document replacement strategies have a great impact on web cache performance [13, 2]. The replacement process is tightly related to the cache size: if an infinite cache exists, there is no need for such a document removal process since all the documents that enter the cache are kept for ever. In real caches documents have to be replaced because of a lack of memory space, and the choice of the next document to be removed is quite important.

The removal process is started once the web cache is full or the remaining amount of memory is not enough to store the new incoming document. Usually, this strategy is referred to as the "on-demand replacement strategy" to distinguish it from the so called "periodic replacement strategy" where documents are removed after a certain amount of time [12]. The argument for using periodic removal is that if a cache is nearly full, then running the removal policy only on-demand (when no more space is available) will invoke the removal policy on nearly all document requests. If removal is time consuming, it might generate a significant overhead. On the other hand, periodic removal reduces the hit-ratio because documents are removed earlier than required.

The rest of the report is organized as follows: In Section 2, we discuss some of the document replacement strategies propose in the literature for web caching. In Section 3, we show the importance of different document replacement parameters using Principal-Component-Analysis. In Section 4, we present a new replacement strategy that is derived from the Nearest Neighbor Classifier (NNC), which is commonly used in pattern recognition and classification problems. In Section 5, a set of experiments is presented in order to show the impact of the NNC strategy on the web cache performances. Section 6 concludes this paper.

2. REPLACEMENT STRATEGIES

The document replacement strategy can be split in two phases. First, the documents are sorted in order to determine the removal ordering. Second, one or more documents are removed from the head of the removal list. The algorithms currently used in practice to replace documents have not been designed specifically for the web but, instead, have been taken straight form earlier work on processor caches. Most of these methods are not well adapted to the web environment because they focus only on one or a part of the parameters that could impact the web cache performances and ignore the rest.

Several strategies have been proposed such as the least frequently used (LFU), first in first out (FIFO) and many others. Only few of them have however, been implemented and used in real cache servers. Most of the time, documents are sorted in the cache according to the least recently used (LRU) strategy. The efficiency of the LRU strategy has been widely discussed in literature [13, 1]. The fact that LRU focuses only on the time in selecting documents has been the main reason for criticism [7]. Many web traffic studies show that time is not the unique factor that impacts the web cache performance. Other factors such as the number of references to the document, the document size, the document time-to-live and the document retrieve time were identified as a potential key for the document selecting process. What is not clear yet, is how these parameters impact the web cache performance. It becomes more and more clear that an appropriate key for selecting the documents should consider several factors. However, the appropriate way to combine these factors has not been defined yet. In some methods, factors are combined two by two such as the LRU-MIN a variant of the LRU that considers the document size [12], or the LFU_Aging that combines the number of document references and the time the documents entered the cache [1]. These methods show a slight improvement of the cache performances over the LRU strategy.

New methods have been investigated by different research groups all over the world to deal with the document replacement problem. The most interesting results were published by Mark Abram et al. from Virginia University [13] and Bolot and Hoschka from INRIA [7]. In their methods, each document is assigned a priority according to which the removal process will select the next document to be replaced. The priority is computed according to a certain mathematical model [7, 13]. In each model, the different factors are assigned a weight which is calculated using weight optimization methods [7], or simulation driven techniques [13].

The weighted function proposed by Bolot-Hoschka combines several factors related to each document according to the following mathematical model:

$$w(t_i, s_i, rtt_i, ttl_i) = \frac{w_1 * rttl_i + w_2 * s_i}{ttl_i} + \frac{w_3 + w_4 * s_i}{t_i} \quad (1)$$

Where

t_i: the time since the doc i was last referred.

s_i: the document size.

rtt_i: the time it takes to retrieve the document.

ttl_i: the time to live of the document.

w_i: are fixed using Lagrange multiplier technique.

In this model, the time plays an important role. The two terms on the right-hand side capture respectively the cost to retrieve a document and the document locality of reference. The second model, which was proposed by Wooster-Abrams (the HYB algorithm), focuses on the number of references to a document rather than the time. Besides this, the estimated quantities are based on a *per-server* and not on a *per-document* basis. HYB replaces the document with the lowest value of the following expression:

$$w(n_{ref}, CBW, clat) = \frac{clat_{ser(i)} + W_B}{CBW_{ser(i)}} * \frac{n^{W_N}}{s_i} \quad (2)$$

Where

$ser(i)$: The server on which the document i resides.

$clat_j$: The latency to open a connection to the server j.

CBW_j: The bandwidth of the connection to the server j.

s_i: The size of the cached document.

n: the number of references to the document i.

W_B and W_N: are constant that set the importance of the variables.

3. WORKLOAD ANALYSIS

When combining several parameters to remove documents from the cache, the first question that comes into mind is which parameter is the most important. The mathematical model proposed in the weighted methods described in the previous section were not clearly justified. It is clear that these models are a new formulation of the classical replacement strategies such the LRU or the LFU. New parameters have been added either to the time (Bolot-Hoschka) or the number of references (HYB strategy). These parameters express the cost to retrieve documents such as the retrieve-time ($rttl$) or the estimated bandwidth of the connection to the server (CBW_j). In this section, we will try to underline the most relevant parameters in describing the behavior of workloads used for web cache simulations.

3.1. Workload traces

The workloads used in this study are part of access log-files provided by the web server of the Computer Science Department of 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 reference locality. The NLANR workload involves requests coming from different web servers that use NLANR as a proxy server. Usually, this kind of servers are more busy than normal cache servers. These access log-files are quite large and it is very hard to use more than few days of workload duration.