Cache Management for Web–Powered Databases

Dimitrios Katsaros
Aristotle University of Thessaloniki, Greece

Yannis Manolopoulos
Aristotle University of Thessaloniki, Greece

INTRODUCTION

In recent years, the World Wide Web, or simply the Web (Berners-Lee, Caililiau, Luotonen, Nielsen, & Secret, 1994), has become the primary means for information dissemination. It is a hypertext-based application and uses the hypertext transfer protocol (HTTP) for file transfers.

During its first years, the Web consisted of static hypertext markup language (HTML) pages stored on the file systems of the connected machines. When new needs arose, e.g., database access, it was realized that we could not afford in terms of storage to replicate the data we want to publish in the Web server's disk in the form of HTML pages. So, instead of static pages, an application program should run on the Web server to receive requests from clients, retrieve the relevant data from the source, and then pack the information into HTML or extensible markup language (XML) format. Even the emerged "semistructured" XML databases that store data directly into the XML format need an application program that will connect to the database management system (DBMS) and retrieve the XML file or fragment. Thus, a new architecture was born: in the traditional couple of a Web client and a Web server, a third part is added, which is the application program that runs on the Web server and serves data from an underlying repository that, in most cases, is a database. This architecture is referred to as *Web-powered database* and is depicted in Figure 1. In this scheme, there are three tiers: the database back-end, the Web/application server, and the Web client.

BACKGROUND

Due to the existence of *temporal locality* in Web request streams, we can exploit the technique of *caching*, that is, temporal storage of data closer to the consumer. Caching can save resources, i.e., network bandwidth, because fewer packets travel in the network, and time, because we have faster response times. Caching can be implemented at various points along the path of the flow of data from the repository to the final consumer. So, we may have caching at the DBMS, at the Web server's memory or disk, at various points in the network (i.e., proxy caches), or at the consumer's endpoint. Web proxies may cooperate so as to have several proxies to serve each other's misses. All the caches present at various points comprise a *memory hierarchy*. The most important part of a cache is the mechanism that determines which data will be accommodated in the cache space and is referred to as the cache *admission/replacement policy*.

Requests for "first-time accessed" data cannot benefit from caching. In these cases, due to the existence of *spatial locality* in request streams, we can exploit the technique of preloading or *prefetching*, which acts complementary to caching. Prefetching may increase the amount of traveling data, but on the other hand, it can significantly reduce the latency associated with every request.

The role of a cache is to store temporally a set of objects that will most probably be requested by its clients. A cache replacement policy assigns a value to every cached object, called *utility value (UV)*, and evicts from cache the object with the least utility value. The aim of the replacement policy is to improve the cache's effectiveness by optimizing two performance measures: the *hit ratio* and the *cost-savings ratio (CSR)*. The former measure is defined as:

$$HR = \sum h_i / \sum r_i$$

and the latter is defined as

$$CSR = \sum c_i * h_i / \sum ci * r_i$$

where h_i is the number of references to object i satisfied by the cache out of the r_i total references to i, and c_i is the cost of fetching object i in cache. The cost can be defined either as the object's size s_i or as the downloading latency c_i. In the former case, the CSR coincides with the byte–hit ratio (BHR); in the latter case, the CSR coincides with the delay-savings ratio (DSR).

Figure 1. Architecture of a typical Web-powered database

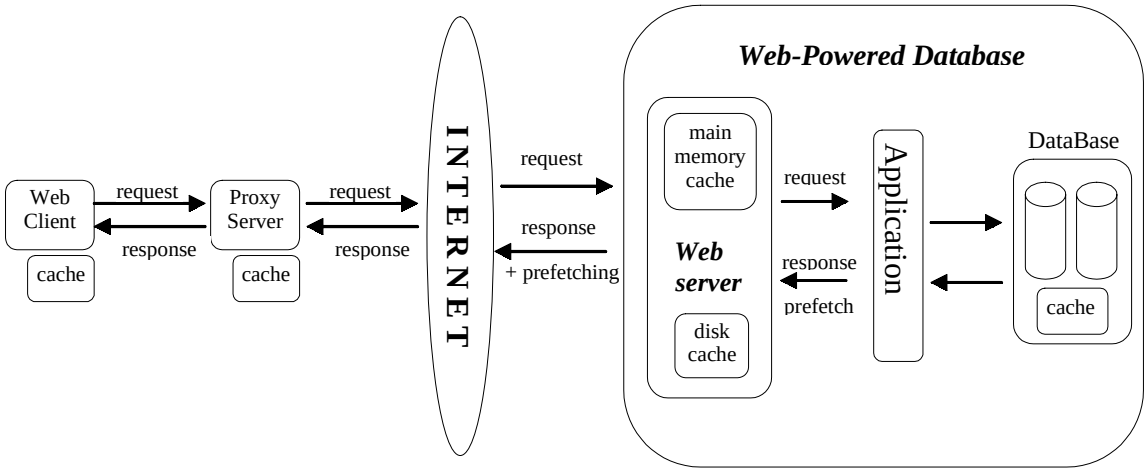

Table 1. A list of factors differentiating Web caching from traditional caching

1. Variable Object Size
The Web object's size varies considerably. It ranges from a few bytes (e.g., small HTML or text files) to several megabytes (e.g., large multimedia files). In contrary, the objects that move through the levels of the caching hierarchy in operating systems or database systems have fixed size, which is equal to the size of a disk block.

2. Variable Fetching Cost
The cost (time penalty) for retrieving a Web object varies significantly. Different objects may have different fetching costs, because they differ in size or in their distance from the client (in terms of network hops). Moreover, the same file may have different fetching costs at different time instances, depending on the server (e.g., heavy load) and network conditions (e.g., congestion).

3. The Depth of the Web Caching Hierarchy
Because caching can happen nearly anywhere, including on the server, on the user's machine, at Internet service protocols (ISPs), at telecommunication companies, at the peering points of national networks, etc., the depth of this hierarchy is significantly larger than the respective depth in computer systems. The large depth of this hierarchy significantly affects the characteristics of the request stream.

4. The Origin of the Web Request Streams
The requests seen by the Web caches, especially by the proxies and the reverse proxies, are not generated by a few programmed processes like the request streams encountered in traditional computer systems. They mainly originate from large human populations with diverse and varying interests.

CACHING IN WEB-POWERED DATABASES

Web cache replacement (Katsaros & Manolopoulos, 2002) is one of the most important areas of Web caching for several reasons. First, studies have shown that the cache HR and BHR grow in a *log-like fashion* as a function of cache size (Breslau, Cao, Fan, Phillips, & Shenker, 1999). Thus, a better algorithm that increases HR by only several percentage points would be equivalent to a several-fold increase in cache size. Second, the growth rate of Web content is much higher than the rate with which memory sizes for Web caches are likely to grow. Finally, the benefit of even a slight improvement in cache performance may have an appreciable effect on network traffic, especially when such gains are compounded through a hierarchy of caches. There are several factors that distinguish Web