M

Multimedia Proxy Cache Architectures

Mouna Kacimi
University of Bourgogne, France

Richard Chbeir
University of Bourgogne, France

Kokou Yetongnon
University of Bourgogne, France

INTRODUCTION

The Web has become a significant source of various types of data, which require large volumes of disk space and new indexing and retrieval methods. To reduce network load and improve user response delays, various traditional proxy-caching schemes have been proposed (Abonamah, Al-Rawi, & Minhaz, 2003; Armon & Levy, 2003; Chankhunthod, Danzig, Neerdaels, Schwartz, & Worrell, 1996; Chu, Rao, & Zhang, 2000; Fan, Cao, Almeida, & Broder, 2000; Francis, Jamin, Jin, Jin, Raz, Shavitt, & Zhang, 2001; Paul & Fei, 2001; Povey & Harrison, 1997; Squid Web Proxy Cache, 2004; Wang, Sen, Adler, & Towsley, 2002). A proxy is a server that sits between the client and the real server. It intercepts all queries sent to the real server to see if it can fulfill them itself. If not, it forwards the query to the real server. A cache is a disk space used to store the documents loaded from the server for future use. A proxy cache is a proxy having a cache. The characteristics of traditional caching techniques are threefold. First, they regard each cached object as having no dividable data, which must be recovered and stored in their entirety. As multimedia objects like videos are usually too large to be cached in their entirety, the traditional caching architectures cannot be efficient for this kind of object. Second, they do not take into account the data size to manage the space storage. Third, they do not consider in their caching-system design the timing constraints that need moving objects.

The size is the main difference between multimedia and textual data. For instance, if we have a 2-hour-long MPEG movie, we need around 1.5 Gb of disk space. Given a finite storage space, only a few streams could be stored in the cache, thus, it would decrease the efficiency of the caching system. As the traditional techniques are not efficient for media objects, some multimedia caching schemes have been proposed (Guo, Buddhikot, & Chae, 2001; Hofmann, Eugene Ng, Guo, Paul, & Zhang, 1999; Jannotti, Gifford, Johnson, Kaashoek, & O'Toole, 2001; Kangasharju, Hartanto, Reisslein, & Ross Keith, 2001; Rejaie, Handley, Yu, & Estrin, 1999; Rejaie & Kangasharju, 2001). Two main categories of multimedia caching solutions can be distinguished.

- The first category is storage oriented; it defines new storage mechanisms appropriated to data types in order to reduce the required storage space.
- The second category is object-transmission oriented; it gives new transmission techniques providing large cooperation between proxies to transfer requested objects and reduce bandwidth consumption.

In this article, we briefly present a dynamic multimedia proxy scheme based on defining the profile that is used to match the capacities of the proxies to the user demands. Users with the same profile can easily and quickly retrieve the corresponding documents from one or several proxies having the same profile. A key feature of our approach is the routing profile table (RPT). It is an extension of the traditional network routing table used to provide a global network view to the proxies. Another important feature of the approach is the ability to dynamically adapt to evolving network connectivity: When a proxy is connected to (or disconnected from) a group, we define different schemes for updating the routing profile table and the contents of the corresponding caches. Furthermore, our approach stores data and/or metadata in function of storage capacities of each proxy (For instance, if storage capacities are minimum, only textual and metadata are cached.).

The remainder of the article is organized as follows. The next section gives a snapshot of current proxy-caching techniques provided in the literature. The section after that presents our approach, and then we finally conclude the article and give our future work.

BACKGROUND

Two main categories of traditional (or textual-oriented) caching solutions can be distinguished in the literature: hierarchical caching and distributed caching. In a hierarchical caching architecture (Chankhunthod et al., 1996), the caches are organized in several levels. The bottom level contains client caches and the intermediate levels are devoted to proxies and their associated caches. When a query is not satisfied by the local cache, it is redirected to the upper level until there is a hit at a cache. If the requested document is not found in any cache, it is submitted directly to its origin server. The returned document is sent down the cache hierarchy to the initial client cache and a copy is left on all intermediate caches to which the initial user requests were submitted. Hierarchical caching has many advantages; it avoids the redundant retrieval of documents from data servers, reduces network bandwidth demands, and allows the distribution of document accesses over a cache's hierarchy. Despite its advantages, hierarchical caching exhibits several drawbacks. The high-level caches, particularly the root cache, are bottlenecks that can significantly degrade the performance of the system. The failure of a cache can affect the system fault tolerance. Several copies of the same document are stored at different cache levels, which is very storage expensive and restrictive, especially when treating multimedia data. Moreover, there is a lack of direct links between sibling caches of the same level.

The distributed caching approach reduces hierarchical links between caches. Several distributed caching approaches have been proposed to address one or more problems associated with hierarchical caching (Armon & Levy, 2003; Fan et al., 2000; Povey & Harrison, 1997). In Povey and Harrison, the authors propose an extension of hierarchical caching where documents are stored on leaf caches only. The upper level caches are used to index the contents of the lower level caches. When a query cannot be satisfied by the local cache, it is sent to the parent cache that indicates the location of the required documents. In Fan et al., the authors propose a scalable distributed cache approach, called summary cache, in which each proxy stores a summary of its cached-documents directory on every other proxy. When a requested document is not found in the local cache, the proxy checks the summaries in order to determine relevant proxies to which it sends the request to fetch the required documents. Two major problems restrain the scalability of the summary-cache approach. The first problem is the frequency of summary updates, which can significantly increase interproxy traffic and bandwidth usage. The second problem is related to the storage of the summaries, especially when the number of cooperating proxies is important. Armon and Levy investigate the cache satellite distribution system, which comprises *P* proxy caches and a central station. The central station periodically receives from the proxy caches reports containing information about user requests. The central station uses this information to foresee what documents could be required by other proxy caches in the near future. It selects a collection of popular Web documents and broadcasts the selected documents via satellite to all or some of the participating proxies. There are two important advantages of this proposal: (a) It anticipates user requests, and (b) it allows collaboration between proxies independent from the geographical distance of a satellite. However, the central station used leads to a weak fault tolerance.

As for textual-oriented caching solutions, two main categories of multimedia caching solutions can be distinguished: storage oriented and object-transmission oriented. In the first category, basic techniques (Acharya & Smith, 2000; Guo et al., 2001) consist of dividing each media data into small segments and distributing them among different caches. The segment distribution helps to have a virtual storage space; that is, the user can store data even if the local cache does not have sufficient free space. In this manner, a cooperative caching schema is defined. When a client requests a multimedia object, the

Related Content