Efficient Caching Policies for the P2P Web Caching

Kyungbaek Kim[1] and Daeyeon Park[1]

Department of Electrical Engineering & Computer Science,
Division of Electrical Engineering,
Korea Advanced Institute of Science and Technology (KAIST),
373-1 Guseong-dong, Yuseong-gu, Daejeon 305-701, Republic of Korea
kbkim@sslab.kaist.ac.kr and daeyeon@ee.kaist.ac.kr

Abstract. Web proxy caching is widely deployed technique but the performance of the proxy cache is limited by the local storage with which it can not accommodate the increasing size of user community. Some researches address this limitation by using the resource of clients with p2p method and achieve the very high hit rate. However, they treat the web objects as homogeneous objects and there is no concern about the various web characteristics, especially size. Even if the total storage is enough to achieve the high hit rate, this overlook limits the byte hit rate of the system and wastes more external bandwidth.

This paper suggests the efficient policies for the web caching system to exploit the characteristics of web objects efficiently. Basically, we concentrate on storing large sized objects efficiently. If the number of clients which want to use the cache system increases, the total storage of the whole cache system increases too. This feature makes enough storage to store the requested objects, especially small sized objects and we can get the high hit rate. However, because of the limited client storage and low priority of large sized objects, we can not exploit the effect of hit for the large sized objects which reduces the internet traffic enormously. If we store large sized objects efficiently, we can achieve not only the high hit rate and the high byte hit rate. To store large sized objects efficiently, we suggest two policies : *storage policy* and *replacement policy*.

For the storage policy, the small sized objects are stored by itself, but the large sized objects are stored by dividing into many small blocks because each client does not have enough storage for the whole of the object. Each client which stores large sized objects just obtains the header objects for them and data blocks are distributed in other clients. According to this, the storage overhead for each client reduces and is balanced. However, the proxy cache which is used in the central server based system stores only small sized objects and the header objects for the large sized objects to achieve the high hit rate and reduce the response delay.

When a cache needs space for new objects, we apply our replacement policy to evict the other less useful cached objects. In this case, we evict small sized objects first. That is, we give the large sized objects higher priority than the small sized objects to increase the availability of the large sized objects. Moreover, to prevent that the missing just one block of the large sized object spoils the whole of the object, we apply the *n*-chance replacement algorithm to the clients. This policy permits the chance of moving blocks for *n* times before evicting the blocks and makes the availability of the object higher.

We examine the performance of the efficient policies via a trace driven simulation and demonstrate effective enhancement of the web cache performance.

keywords : Web caching, Peer-to-Peer, Replacement Policy