IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
e-ISSN: 2278-2834,p-ISSN: 2278-8735
PP 01-06
www.iosrjournals.org

SCPC: A Scalable Co-Operative Proxy Cache for Peer-To-Peer Traffic

Ani.S
Assistant Professor,CSE Department Muslim Association College Of Engineering

Abstract: *Peer to Peer applications are increasing day by day and caching is very important in P2P networks. Many previous papers discuss about the caching in P2P networks, but there are number of constraints in the case of storage management and cooperation among caches. Here proposes a Scalable Co-operative Proxy Cache for Peer-to-Peer Traffic (SCPC), a new cooperative caching system that unites the advantages of both proxy caching and cooperative communications. This paper presents the design of a proxy cache that is completely transparent in the network. The proposed Cache catches and serves traffic from different P2P systems. The storage system implemented in this paper is optimized for storing P2P traffic. Significant improvement in byte hit rate can be achieved using cooperative caching. In the case of inaccessibility of proxy we can't able to access the data in the cache. In such situations co-operative caching mechanism is very important, means then the data can be taken from another cache if the actual proxy is not available.*

Keywords: *P2P network, proxy caching, storage system, cooperative mechanism*

I. Introduction

Peer to Peer traffic uses a chief portion of the Internet, exceeding traditional web surfing traffic, and is expected to increase further. More users are repeatedly joining peer to peer systems and more objects are being made available in the Internet. The congestion due to P2P traffic not only affects users of P2P applications, but also those of other applications such as web. More than 90% P2P traffic navigate transfer links among ISPs, thereby affecting the lower level (i.e. user) of ISPs. The overwhelming bandwidth consumption of peer to peer systems is the supplementary result of this problem. The total volume and expected high intensification of P2P traffic have harmful consequences, including: considerably increased load on the Internet backbone, thus, higher chances of congestion; and large cost on Internet Service Providers (ISPs), hence, higher service charges for all Internet users.

A possible solution for reducing those negative impacts is to cache a portion of the P2P traffic such that upcoming requests for the same objects could be served from a cache in the requestor's autonomous system (AS). Caching of P2P traffic is a good approach, because objects in P2P systems are mostly unchangeable and the traffic is highly recurring. Besides, changing P2P protocols is not required in caching and can be organized transparently from clients. Therefore, ISPs can readily deploy caching systems to reduce their costs. Without affecting the performance of other application on the Internet we have to design a proxy cache that transparently catches and serves traffic from various P2P systems.

Many earlier studies shown that network traffic produced by P2P applications has different characteristics than traffic generated by other Internet applications. For example, object size, object popularity, and connection prototype in P2P systems are relatively different from their other parts in web systems. Most of these characteristics impact the performance of caching systems, and therefore, they should be considered in their design. It is very difficult to design a proxy cache for P2P traffic compared to any other networks such as web or any other traffic. The problem is due to many reasons. The main problem with the P2P traffic is that different P2P system uses different port numbers for connection setup, there is no standard port number for P2P systems. So every P2P system has to take its own mechanism to handling P2P protocols. Consequence of this temporary mechanism causes the repeated modification of the P2P caching system to handle new P2P protocols. In addition extra, probably massive, investment is required for maintaining the entire system. Due to above mentioned problem the procedure to identify the P2P connection becomes complex.

Another problem with the P2P network is the multitude and dissimilar nature of existing P2P systems. Moreover the developers of such systems did not provide any mechanism for the possible caching of P2P traffic. From the above facts we can conclude that fixing whether a request is P2P traffic and if it is a P2P then take out data from that are very complex task. In such systems caching of data is a very complex process.

A new storage management system is needed due to the special features of P2P objects such as large sizes and variable segment lengths. A more advantageous method is to set up cache proxies to cache P2P traffic. Many P2P caching schemes have been proposed [1, 2, 3], and a small number of systems have started caching. But these systems were using mechanism similar to web caching. Web caching is not suitable for P2P systems due to particular characteristics (mentioned above) of its objects [1,4]

The caching possibility of P2P traffic depends on the distribution of the file sizes. The main issue related to caching is what percentage of the traffic would be provided from cache. As mentioned above, Web caching mechanisms are not appropriate to P2P traffic because of the dissimilar properties of this traffic.

In [5] analyzes the possibility of caching in P2P traffic. They analyses two aspects: theoretical caching potential and empirical caching. For theoretical caching potential use a program that reads the traces of the P2P traffic collected by a server, and calculates the optimal, static, set of files that should have been stored on a given size cache, to generate the optimum byte-hit-rate on the traffic defined by the trace. P2P traffic transmitted over an ISP link is highly repetitive and consequently responds well to caching. The analysis of the traffic shown as 67% byte-hit-rate which compares satisfactorily with web caching hit rates (30% to 60%). The disk space required for effective caching of P2P traffic is small. They concluded that the byte-hit-rate computed in their installation with respect to the traffic volume, pointed that a higher byte-hit-rate can be achieved on links with a larger traffic load. From the above we can conclude that caching of P2P traffic is effective for deal with the bandwidth drain caused by the increase of P2P traffic.

Mohamed Hefeeda, Cheng-Hsin Hsu and Kianoosh Mokhtarian [6] who gave a detailed description of design and implementation of proxy caching. However, the lack of cooperative mechanism reduces the cache hit rate. Here proposes the cooperative caching. Caches installed in, different ASs or caches in a large AS, cooperate with each other to serve requests from clients in their networks.

The cooperation between caches works as follows: When a cache receives a request for an object that it does not store locally, it first finds out whether another cache in the AS has the requested object. If any of them does have the object, the object is directly served to the requesting client and is copied to the local cache.[4] gives the idea of cooperative mechanism but the design of proxy cache and storage mechanism mentioned in [6]is more effective in P2P systems.

The main contributions of this paper are as follows:

1. In this paper, present the design and implementation of a proxy cache for P2P traffic, which is called SCPC.
2. Here propose an optimized storage system for P2P proxy caches.
3. Proposes the co-operative caching mechanism. In the case of unavailability of proxy we can't able to retrieve the data from the cache. So if we use co-operative caching mechanism means the data can be taken from another cache if the actual proxy cache is not available.

II. Related Work

Several studies have analyzed various portions of P2P systems. Gummadi [1] studied the P2P traffic in Kazaa and identified the object characteristics of it and show that P2P objects are mostly unchangeable. In P2P systems large objects that are downloaded at most once. To represent the popularity of web objects [2] Zipf distribution is used, the study demonstrates that the popularity of P2P objects does not follow that.

The advantages of caching P2P traffic have been shown in [3] . Object replacement algorithms particularly designed for proxy cache of P2P traffic have also been proposed in [5] and [7]. The above mentioned works do not handle the design and functioning of an actual proxy cache for P2P traffic and does not mention the storage management issues in such caches.

The writers of [8] propose caching mechanisms by using the existing web caches to serve P2P traffic. Web caches are not suitable for P2P caching because of its characteristics such as large object size, request for arbitrary byte range of object etc. To store and serve P2P traffic, proxy caches need to recognize connections belonging to P2P systems and extract required information from them. Several previous works deal with the problem of recognizing P2P traffic, including [9], [10], [11].

Middleman [12], who has studied collection of proxies for streaming media delivery, have considered main feature of streaming media accessing. They identified that continuous media objects such as video or music clips are frequently partially accessed. From this observation partial caching approaches have been proposed there by reduce the memory space required for cache. The basic approach is that only segments of object are cached. Segmentation can be done in different ways. In uniform segmentation, uniform-size segments of objects are cached, whereas in the exponential case, the segment size doubles along the viewing direction. The Rcache [13] use multiple proxies to cope up with limited resources available from a single cache, emphasis on the memory and disk utilization.

Li Xiao [14] proposes a proxy-browser system which mainly used client/server systems for Web caching, where a group of networked clients connects to a proxy cache server and each client has a browser cache. The Web caching system implemented on a proxy-browser system works as: Upon a Web request of a client, the browser first checks if the requested document exists in the local browser cache. If so, the request will be served by its own browser cache. Otherwise, the request will be sent to the proxy cache. If the requested document is not found in the proxy cache, the proxy server will immediately send the request to its cooperative caches, if any, or to an upper level proxy cache or to the Web server, without checking if the document exists in other browsers' caches.

Mohamed Hefeeda, Cheng-Hsin Hsu and Kianoosh Mokhtarian [6],[15], demonstrates that caching P2P traffic is more complex than caching other Internet traffic, and it needs several new algorithms and storage systems. Then, the paper presents the design of a proxy cache for P2P traffic, called pCache that transparently intercepts and serves traffic from different P2P systems. A new storage system is proposed and implemented in pCache. This storage system is optimized for storing P2P traffic. In addition, a new algorithm to infer the information required to store and serve P2P traffic by the cache is proposed. But in this paper cooperation among caches in different AS is not used. Cache byte hit rate can be increased by cooperation among caches.

III. Design Of Proxy Cache

A potential solution for reducing adverse effect of P2P networks in Internet is to cache a portion of the P2P traffic

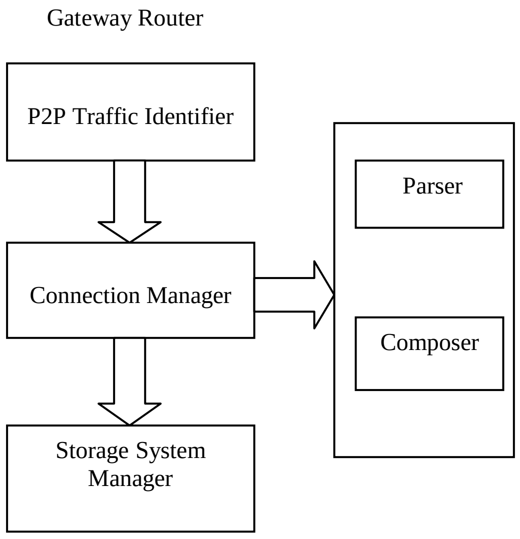


Fig.1 Functional block diagram of Proxy cache.

there by further requests for the same objects could be served from a cache in the requestor's autonomous system (AS).

The block diagram of Proxy cache is shown in fig.1.The user in peer to peer network issues a request to the server. The cache transparently captures the request and check whether it belongs to P2P networks. If it belongs to the peer to peer network then the request routed to connection manager of the cache otherwise request forwarded as normal web traffic. Traffic identifier that is the gateway router checks the address of the requested peer and the request forwarded to the cache. In the Connection manager the request routed to the traffic processor.

Traffic processor consists of two functional units Parser and Composer. Parser extracts the object request messages. Composer constructs accurately formatted messages to be sent to peers. Traffic processor works as follows: parser extract the request. The extracted details consists of the file name, size of the object, date and time of the request, requester address. Then check the requested data within the cache. The storage manager performs the following action.

The storage system contains a hash table in which each object is uniquely identified by object id. This table contains information about cached object such as size of the file and reference counter. Reference counter is used to identify how many connections are currently used this file.

If the requested data is not in the cache the composer constructs the request and sends to the peer. Upon receiving the request the peer send the requested object to the cache. The cache stores the copy of the object there by the following request for the same object can be served from the cache and also sends it to the requested peer. By caching the frequently used object the users such as universities can significantly reduce the cost of Internet usage.

IV. Cooperative Mechanism

As mentioned, the most closely related work is that of Mohamed Hefeeda, Cheng-Hsin Hsu and Kianoosh Mokhtarian [6] who gave a detailed description of design and implementation of proxy caching. However, lack of cooperative mechanism reduces the cache hit rate. Caches installed in, different ASs or caches in a large AS, cooperate with each other to serve requests from clients in their networks. The cooperating ASs may have a close relationship to transmit each other's traffic, or they can be located within the same geographical area.

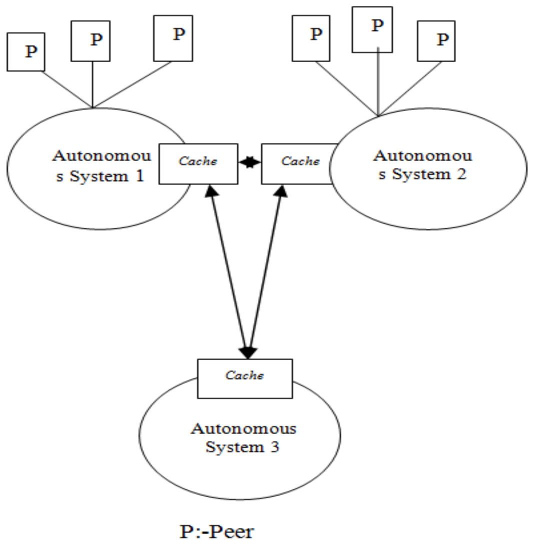
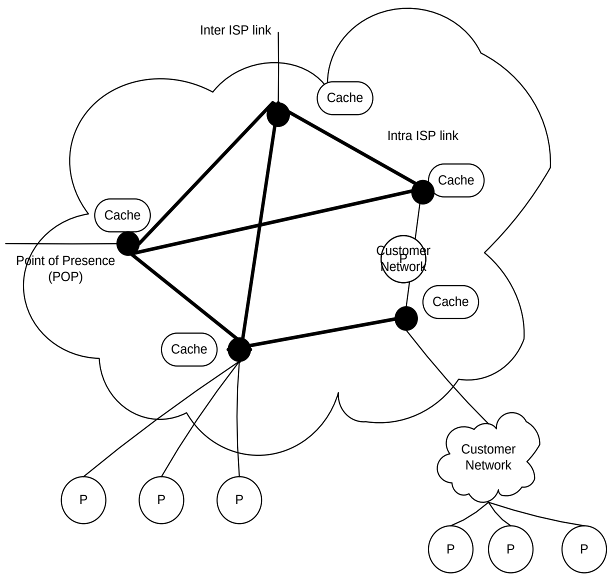


Fig.2 Cooperation among caches in different Autonomous Systems

Fig2 shows the cooperation among caches in different system. Fig 3 shows the cooperation among the caches installed within single large ISP (Internet Service Provider).Cooperation among caches within the same AS would be simpler to implement than cooperation among caches in different ASs. This is because in the first case, all caches are possessed and managed by a single entity, while in the second case, more than one parties are involved.
Without cooperative caching the designed cache is five times efficient than the web proxy implementation. By using cooperative caching mechanism the performance is further increased.
In the proposed system the cache hit rate is higher compared to any other systems. Thereby reduce file transfer through the expensive links (Internet).



P:-Peer
Fig.3 Cooperation among the caches within single large ISP

V. Experimental Results

The objective of this paper is to analyse the performance of cooperative caching. Assume that in an AS with cache that serve K objects then the cost for serving the objects is given by

$$\boldsymbol{C = Cl \sum_{n=1}^{K} g(n) + Ce \sum_{n=q+1}^{K} g(n)} \qquad \textbf{(4.1)}$$

where g (n) is the probability of accessing the object at index n, and costs of downloading an object from the local cache and an external source are Cl and Ce, respectively. Note that objects are indexed based on their popularity g(r)>g(s) for all r < s. The equation (1) denotes the average cost because, for large K, the cache stores the most m popular objects. The second term in (1) is the cost of accessing objects m+1, m+2 ...K from external source because the first m objects are stored in the cache. The first term is the cost of all objects.

Fig.4 shows the comparison between the independent and cooperative caching system installed in different autonomous system. Blue line shows the performance of proxy caching without cooperative mechanism. Fig 5 shows the comparison between the independent and cooperative caching system installed in single autonomous system. Red line shows the performance of cooperative caching system. The graph plots the byte hit rate verses the cache size. In both case we can see the performance efficiency of the cache due to the cooperation.

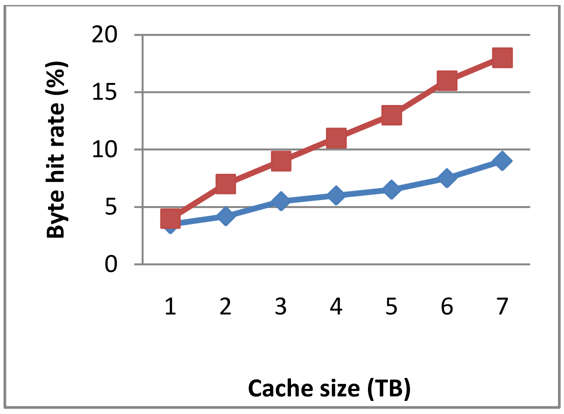


Fig.4 Comparison between independent and cooperative caching installed in different Autonomous Systems

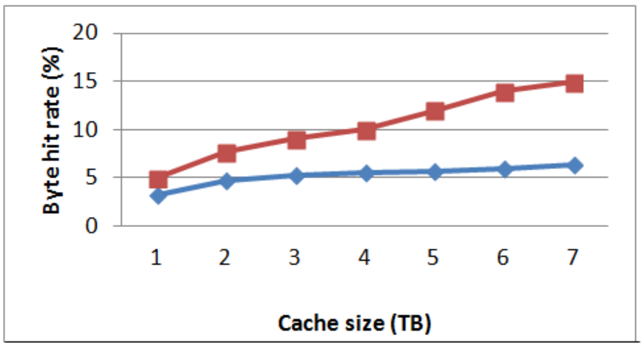


Fig.5 Comparison between independent and cooperative caching installed in single autonomous System

VI. Conclusion And Future Scope

This paper analyzes the importance of cooperative caching in P2P systems. A potential solution for reducing adverse effect of P2P networks in Internet is to cache a portion of the P2P traffic there by further requests for the same objects could be served from a cache in the requestor's autonomous system (AS).

Proxy caching is an important technique to reduce transmission cost in Peer to Peer traffic. However, its effectiveness is restricted by the inadequate storage space and feeble cooperation among proxies and their clients. Here proposes a Scalable Co-operative Proxy Cache for Peer-to-Peer Traffic (SCPC), a new cooperative caching system that unites the advantages of both proxy caching and cooperative communications. This paper presents the design of a proxy cache that is completely transparent in the network. The proposed Cache catches and serves traffic from different P2P systems.

The storage system implemented in this paper is optimized for storing P2P traffic. Significant improvement in byte hit rate can be achieved using cooperative caching. In the case of inaccessibility of proxy we can’t able to access the data in the cache. In such situations co-operative caching mechanism is very important, means then the data can be taken from another cache if the actual proxy is not available. Here no analytical model is used to measure the caching from different P2P systems.

References

[1]. K. Gummadi, R. Dunn, S. Saroiu, S. Gribble, H.Levy, and J. Zahorjan, “Measurement, modeling, and analysis of a peer-to-peer file-sharing workload,” in *Proc. 19th ACM Symp. Operating Systems Principles (SOSP’03)*, Bolton Landing, NY, Oct. 2003, pp. 314–329.

[2]. L. Breslau, P. Cao, L. Fan, G. Phillips, and S.Shenker, “Web caching and Zipf-like distributions: Evidence and implications,” in *Proc. IEEE INFOCOM’99*, New York, NY, Mar. 1999, pp. 126–134.

[3]. N. Leibowitz, A. Bergman, R. Ben-Shaul, and A.Shavit, “Are file swapping networks cacheable? Characterizing P2P traffic,” in *Proc. 7th Int. Workshop on Web Content Caching and Distribution (WCW’02)*, Boulder, CO, Aug. 2002.

[4]. COPACC: An Architecture of Cooperative Proxy-Client Caching System for On-Demand Media Streaming Alan T.S.Ip, Student Member, IEEE, Jiangchuan Liu, Member, IEEE, and John Chi-Shing Lui, Senior Member, IEEE.

[5]. **“**pCache: A Proxy Cache for Peer-to-Peer Traffic” Mohamed Hefeeda, Cheng-Hsin Hsu, and Kianoosh Mokhtarian School of Computing Science, Simon Fraser University, Surrey, BC, Canada

[6]. “Design and Evaluation of a Proxy Cache for Peer-to-Peer Traffic” Mohamed Hefeeda, Senior Member, IEEE, Cheng-Hsin Hsu, Member, IEEE, and Kianoosh Mokhtarian, Student Member, IEEE

[7]. M. Hefeeda and O. Saleh, “Traffic Modeling and Proportional Partial Caching for Peer-to-Peer Systems,” IEEE/ACM Trans. Networking, vol. 16, no. 6, pp. 1447-1460, Dec. 2008.

[8]. G. Shen, Y. Wang, Y. Xiong, B. Zhao, and Z. Zhang, “HPTP: Relieving the Tension between ISPs and P2P,” Proc. Int’l Workshop Peer-to-Peer Systems (IPTPS ’07), Feb. 2007.

[9]. S. Sen, O. Spatscheck, and D. Wang, “Accurate, Scalable In-Network Identification of P2P Traffic Using Application Signatures,”Proc. Int’l World Wide Web Conf. (WWW ’04), pp. 512-521, May 2004.

[10]. T. Karagiannis, A. Broido, M. Faloutsos, and K. Claffy, “Transport Layer Identification of P2P Traffic,” Proc. ACM Conf. Internet Measurement (IMC ’04), pp. 121-134, Oct. 2004.

[11]. A. Spognardi, A. Lucarelli, and R. Di Pietro, “A Methodology for [12]P2P File-Sharing Traffic Detection,” Proc. Int’l Workshop Hot Topics in Peer-to-Peer Systems (HOT-P2P ’05), pp. 52-61, July 2005.

[12]. S. Acharya and B. Smith, “Middleman: A video caching proxy server” in *Proc. ACM NOSSDAV*, Chapel Hill, NC, Jun. 2000.

[13]. Y. Chae, K. Quo, M. Buddhikot, S. Suri, and E. Zegura, “Silo, rainbow, and caching token: Schemes for scalable fault tolerant stream caching,” *IEEE J. Select. Areas Commun.*, vol. 20, no. 7, pp. 1328–1344, Sep.2002.

[14]. Building a Large and Efficient Hybrid Peer-to-Peer Internet Caching System Li Xiao, Member, IEEE, Xiaodong Zhang, Senior Member, IEEE, Artur Andrzejak, and Songqing Chen, Student Member, IEEE Transactions On Knowledge And Data Engineering, Vol. 16, No. 6, June 2004

[15]. “Peer-to-Peer Caching Schemes to Address Flash Crowds” Tyron Stading Petros Maniatis Mary Baker Computer Science Department, Stanford University Stanford, CA 94305, USA