SB-10-13

Replication Algorithms for Scalable Streaming Media in Content Delivery Networks

Zhou SU Jiro KATTO Yasuhiko YASUDA

School of Science and Engineering, Waseda University

1. Introduction

CDN (Content Delivery Networks) improves end-user performance by replicating web contents on a group of geographically distributed servers. However, current replica strategies in CDN are to simply and repeatedly keep the complete replica of the original object on many content servers. Disadvantages of this method are as follows: to repeatedly store the same large-sized object into different content servers consumes too much server space. Also because some of the content servers are not always requested by clients, to keep replicas on these servers waste storage cost. It is more serious for replicating some large-sized objects such as streaming media, e.g. high quality video, which are being distributed over the Internet more and more.

There have been many other published studies on streaming media. However, almost all researches are about how to replace or cache streaming in a single proxy cache. How to efficiently replicate and distribute streaming media in a group of servers such as CDN has not been mentioned.

Recently researchers found that scalable (layered) streaming media is appropriate for the Internet because of its better flexibility and functionality. In this paper, we therefore discuss about a scheme to replicate scalable streaming contents in CDN. Our work focuses on how to replicate different layers of different streams into different content servers to save the system resources and improve the user delay at the same time.

We firstly carry out a theoretical analysis of storage cost, capacity limit and access distribution of scalable streaming contents. Based on these analyses we then propose a replication algorithm in which not only the popularities of streams and servers but also the current network topology is considered. Through simulations, we check the performance of our proposal when the related parameters are changed. Simulation results show that our proposal can efficiently minimize user delay and network traffic.

2. Scalable Streaming

Scalable (or layered) streaming is supported by some standards, such as MPEG-2, MPEG-4 and H.264. Basically, the original signal is coded into several layers, from the lowest (base) to the highest (enhancement) layers. There are some advantages by using scalable streaming. 1) A sender can easily control compression ratio by dropping some of layers according to network condition. 2) A receiver can select the number of layers according to its own network condition, known as receiver-driven layered multicast. 3) A content server can keep only some of layers instead of the whole original one when this stream is not popular. Then it can give other popular streams more space. 4) Accordingly, more kinds of streams can be kept in the cache. Only "thumbnail" layers can be stored instead of simply removing the whole data when there is no sufficient space in the cache.

3 Proposal

3.1 Storage Capacity

We assume each content server exists in a different autonomous system (AS). Here, a server in AS i, i={1,…,I}, has S_i bytes of storage capacity and has clients that request objects at aggregate rate λ_i. We assume that there are J streams in a server and every stream (stream j, j={1,…,J}) is encoded by Q layers. We define Bq,j as the data size of the q'th layer (q={1,…,Q}) of stream j and Pq,j is the request probability for the q'th layer of stream j. We denote a matrix of all Xi,q,j by $\boldsymbol{X}$, where:

Xi,q,j=1 (if the q'th layer of stream j is stored at AS i), (1)

Xi,q,j=0 (otherwise)

The storage capacity is constrained by the space available at AS i, i.e.

$$\sum\sum\sum Bq,j \bullet Xi,q,j <= S_i \qquad i=\{1,\ldots,I\} \tag{2}$$

In this paper, our goal is to choose the Xi,q,j so that both user response time and storage cost can be minimized.

3.2 Storage Cost

We also consider in this paper that a storage cost is associated with how much and how long the storage is used at each server. Such a "storage utility" model is in fact being offered by some companies, in which one can use storage as much as he needs, and pays only for what he uses.

Here we assume that each stream is storaged in a server for a fixed period, and the cache contents are updated periodically. Then, the storage cost for a given AS i is denoted as Mi(\$/byte) and the total storage expense limit is constrained by α, i.e.

$$\sum\sum\sum Mi \bullet Bq,j \bullet Xi,q,j <= \alpha \qquad i=\{1,\ldots,I\} \tag{3}$$

A parameter Mi can be considered as a kind of priority (weighting factor) of each content server.

3.3 Minimize the Traversed AS Hops

Average AS hops that a request must traverse can reflect the download time of a requested object and is an indicator of user perceived latency. Therefore, the traversed AS hops is an important criterion to evaluate the performance of CDN and P2P networks [3] [4]. In this paper, our goal is to replicate different layers of different streams into the server, so as to minimize the the average AS hops.

If clients send a request for an object (q,j) (which means the q'th layer of stream j) to its original server from one of other servers k (k={1,…,I}), we know the total AS hops from the original server to other servers k can be obtained as

$\sum_k \lambda_k P_{q,j} \bullet D_{k,q,j}(\boldsymbol{Xo})$ (4)

where $D_{k,q,j}(\boldsymbol{Xo})$ is the shortest distance from server k to the q'th layer of stream j under the placement of the streams to origin servers $\boldsymbol{Xo}$.

Then, we continue to think about replicating the replica of this object (q,j) onto a given content server n which is not its original server. After keeping this replica at server n, similar to Eq. 4, we can get the AS hops when requets for object (q,j) happen from other server k (k={1,...,I}) as follows:

$\sum_k \lambda_k P_{q,j} \bullet D_{k,n}(\boldsymbol{X})$ (5)

Here, X is the placement after keeping a replica at the content server n. $D_{k,n}$ means the shortest distance from server k to server n.

Finally, we can calculate the reduced As hops if we replicate the q'th layer of stream j into server i.

$R_{i,q,j} = P_{q,j} \bullet \sum_k \lambda_k \bullet \{ D_{k,q,j}(\boldsymbol{Xo}) - D_{k,i}(\boldsymbol{X}) \}$ (6)

When we need to decide the order to replicate which layer of which stream into a given server, we can use the $R_{i,q,j}$ as an index.

3.4 Proposed Algorithm

Here, we give our algorithm as follows:

1) For a given replication object (the q'th layer of stream j), each content server i is given a replica priority ($R_{i,q,j}$) decided by Eq.(6).

2) The algorithm picks the server-object pair which has the highest $R_{i,q,j}$ and stores the object (q, j) in a server i firstly. This results in a new placement $\boldsymbol{X}$. The algorithm then re-calculates the $R_{i,q,j}$ under this new placement and picks the server-object pair which has the highest $R_{i,q,j}$ to be stored.

3) The above pick-up selection is iterated until either the server capacity constraint (decided by Eq.(2)) or storage cost constraint (decided by Eq.(3)) exceeds its limit.

4. Simulation Results

We assume that there are 21 nodes in our network simulator. Among these nodes, there are 15 original servers and 6 content servers, respectively. The physical distances among nodes are originally decided by random. As for the streaming media, there are 1000 different streams. The length of each stream is uniformly distributed between one minute and ten minutes. Each stream is encoded into two constant bit rate layers [1]. In our simulation, not all the clients will stop watching a requested stream after the playback of the whole stream. The position where a client stops watching a stream is decided at random. The request distribution yields to a Zipf distribution with a Zipf parameter of 0.8. Client requests arrive according to the Poisson process. For each stream, the client can request either a base layer only or a complete video consisting of a base layer and an enhancement layer. All clients are always redirected to the closest server. The total request times in the simulations are 10000.

We compare out proposal with three algorithms, which are *LRU*, *LFU and* the *Hierarchical Caching* [2].

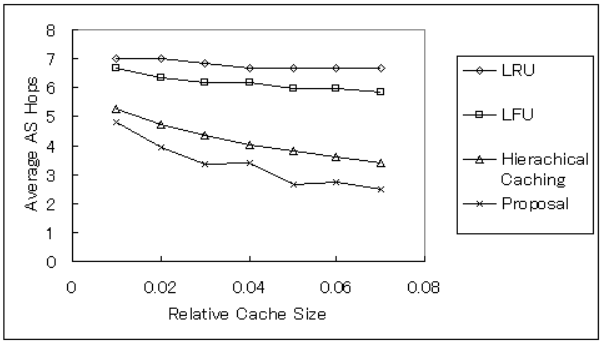

Fig.1: Comparison of Average AS Hops with Different Replication Algorithms.

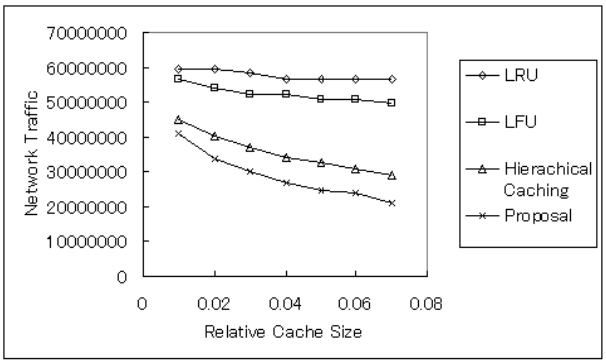

Fig. 2: Comparison of Network Traffic with Different Replication Algorithms.

In Fig.1, although the *Hierarchical Caching* shows better performance than *LRU/LFU*, its average AS hops are larger than the proposal as the *Hierarchical Caching* performs locally and doesn't consider network topology in CDN. We can find that the proposed one can substantially reduce the average hops by almost 50% compared with LFU. The reason is that whether a layer of one stream should be replicated in a content server is decided by considering many aspects, which includes server popularity, stream request frequency and network topology. Fig. 2 evaluates the network traffic among different servers. We can find that the proposed algorithm performs best and can reduce the network traffic. It also verifies that algorithms with low Average AS Hops lead to reduction of network traffic.

5. Conclusion

In this paper, we studied how to optimally replicate streaming media among CDN servers. We presented a scheme to replicate scalable streaming contents to more effectively use the content server's resource. Simulation results showed that the proposed algorithm could obtain better performance than other conventional ones.

References

[1] J.Kangasharju, F.Hartanto, M.Reisslein, K.W. Ross, "Distributing Layered Encoded Video through Caches", IEEE Trans on Computers, vol. 51, n. 6, pp. 622-636, June 2002.

[2] Z.Su, T.Washizawa, J.Katto, and Y.Yasuda, "Integrated Pre-fetching and Caching Algorithm for Graceful Image Caching", (to be published by IEICE Trans on Commun, Vol.E86-B, No.9, Sep. 2003)

[3] L. Qiu, V. N. Padmanabhan, and G. M. Voelker. On the placement of web server replicas. In Proceedings of IEEE Infocom, Anchorage, AK, pp 22-26, April 2001.

[4]Q.Lv, P.Cao, E.Cohen, K.Li, S.Shenker: "Search and Replication in Unstructured Peer-to-Peer Networks". ICS 2002