SOP: A SIU-based Overlay Proxy Service for Peer-to-Peer Streaming

*Tein-Yaw Chung, Yang-Hui Chang, Shi-Shung Chiu,
Chun-Jei Yang, Yung-Mu Chen and Huai-Lei Fu

Department of Computer Science and Engineering, Yuan Ze University, Taiwan
*E-mail: * csdchung@saturn.yzu.edu.tw*

ABSTRACT

The heterogeneity of end hosts poses a significant challenge in delivering asynchronous multimedia access services across overlay networks. Conventional cache-and-relay methods do not address either the link bandwidth constraint of end hosts, or the variable bit rate properties of video streaming. This work proposes an end-host based proxy service, called a Synchronization Interval Unit (SIU) based Overlay Proxy service (SOP), to enable smooth video playback and boost server capacity in a heterogeneous environment. SOP segments a movie into many SIUs, which are processed off-line with a proposed SIU-based smoothing algorithm (SSA) to handle the link and buffer constraints of heterogeneous end hosts. SSA generates a transmission profile for each SIU, including parameters relating to three resource constraint switching modes, BB, BR and BRR, between the download of each SIU to ensure smooth playback and to raise the cache hit rate. This study also develops four cache replacement strategies, namely, access-unaware, local access-aware, regional access-aware and global access-aware, each offering various levels of prefix, selective and cache-and-relay caching and caching cooperation among peers. A simulation experiment is performed to investigate the performance of SOP in cluster-based P2P networks. SSA with the access-aware caching approach can significantly improve the proxy hit rate, and can lower the server load by 15-35% server load at various user arrival rates. The simulation results indicate that SOP can relieve server load, and that the presented novel mechanisms can effectively improve the streaming service quality.

1: INTRODUCTIONS

Overlay networking systems have been presented to support synchronous [1,2] and asynchronous access services [3,4]. Overlay multicast solutions are attractive for synchronous access to real-time media. Multicasting allows a server to serve many clients at a low network link cost and server bandwidth. Examples of synchronous services include the delivery of live real-time content and near video-on-demand (N-VoD) services are. Many overlay networks, including Narada [1] and D^2MST [5], were developed to provide such multicast streaming services.

For asynchronous services, the requirement of on-demand media distribution poses a significant challenge to network design because the unpredictability of user requests implies that multicasting is no longer useful to reduce network traffic. Thus, overlay networking technologies [3,4] using cache-and-relay methods have been proposed to reduce network traffic. The cache-and-relay methods adopt the temporal correlation of asynchronous requests and buffering capabilities of overlay nodes to increase the involvement of end hosts in relaying media data to other clients, hence reducing the server load and bandwidth.

The problems with current cache-and-relay methods are that they adopt a sliding window method and formulate the problem as a model of media distribution tree (MDT) [3,4], which perform poorly when the media objects are not popular or the end-host buffers are small. Moreover, they characterize the heterogeneity of the end-host link bandwidth based on degree constraint, which is too coarse when the required bandwidth for streaming a media object varies significantly [5]. Therefore, current methods are inefficient, or do not ensure smooth video playback when end hosts have heterogeneous link bandwidth and storage.

This work proposes a Synchronization Interval Unit (SIU) based Overlay Proxy service (SOP) that adopts a segmentation approach. SOP allows every overlay node to offer proxy services, and does not use an infrastructure proxy. Rather than applying a cache-and-relay scheme [3,4], an SOP overlay node may cache many non-contiguous SIUs, each comprising several Groups of Pictures (GOPs), of a movie title that it is playing. This work presents four simple and distributed cache replacement strategies, namely access-unaware, local access-aware, regional access-aware and global access-aware, to increase the SIU hit rate at proxy peers. Through SIU caching, SOP provides proxy services like prefix and selective caching when user requests are infrequent and cache-and-relay when user requests are heavy.

To ensure smooth video playback, SOP solves the difficulty of heterogeneous link bandwidth and storage of end hosts by processing SIUs off-line by the proposed SIU-based smoothing algorithm (SSA). SSA addresses both the buffer constraint and the link bandwidth [5] of

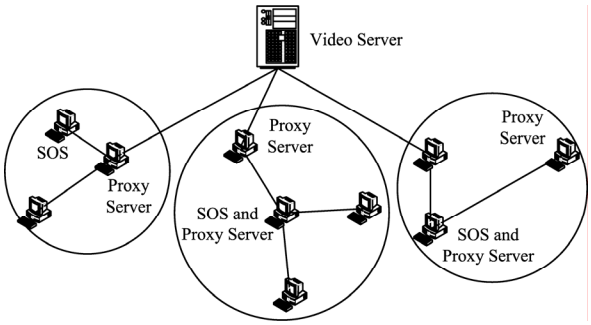

Figure. 1 SOP system architecture.

end hosts. An end host may download SIUs from different peer nodes while it is playing a movie title, and therefore face different resource restrictions in each SIU downloading. Such resource constraint changes or switches require special care to ensure smooth playback at peer nodes. SSA categorizes resource switching into three modes, BB (buffer to buffer constraint), BR (buffer to rate constraint) and BRR, and derives a transmission profile for each SIU to cover all three switching modes.

The performance of SOP and various cache and replacement strategies was studied using computer simulation. The simulation results indicate that SOP can reduce video server load by employing an appropriate caching method. Additionally, the cache hit rate can be achieved is better than that of prefix and selective caching and cache-and-delay caching through awareness of the access patterns of other peers.

The remainder of this paper is organized as follows. Section 2 describes the SOP architecture. Section 3 then defines an SIU, and presents the SSA. Next, Section 4 describes the proxy cache replacement algorithm. Section 5 summarizes the simulation results and the performance analysis. Conclusions are finally drawn in Section 6.

2: SOP ARCHITECTURE

SOP operates on a peer-to-peer network with clustering as depicted in Fig. 1, in which peer nodes are split into many clusters based on their geographical proximity, and each cluster forms a search space. When searching for a resource, a peer searches a local cluster first and then proceeds to other remote clusters. A peer user can restrict its search to a local cluster, or flood the whole peer-to-peer network with its search.

According to Fig. 1, SOP has three major components, a video server, a proxy server and a super-object server (SOS). A video server is a peer node that publishes its video programs via a peer-to-peer network. Every peer node that retrieves a film from a video server becomes a proxy server. A film is streamed to a peer user client in a sequence of SIUs. A proxy server publishes its cached SIUs to a super-object server (SOS) [6] for the film in its local cluster. SOS is a software module that can be run by agent or active networking technologies whenever a new film is played back in a cluster. SOS maintains all SIU caching information for a film in a local cluster. A system such as

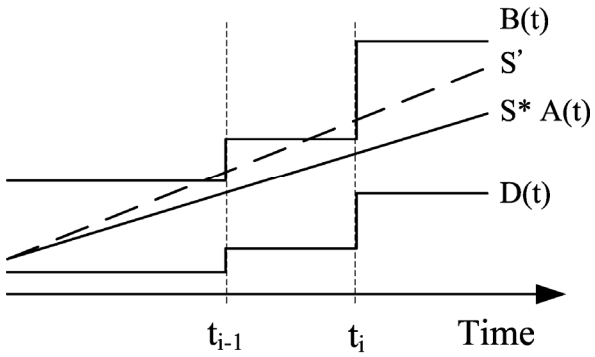

Figure. 2 Buffer-constraint smoothing example.

Chord [7] chooses a peer node with a key that matches that of the film to run the SOS for that film. Therefore, a peer client can easily locate an SOS without a control head, allowing the SIU directory services to be distributed to peer nodes when many video programs are played back at the peer nodes of a cluster.

When a peer node wishes to play a video program, it must first locate an SOS for it. If the node finds an SOS, then the peer user requests the SOS for the SIU profiles in the video program and a list of SIU caching information within the cluster. A peer must transmit a downloading request to the chosen proxy peer before it decides to download an SIU from a proxy peer. On the other hand, each proxy server performs a simple call admission control to ensure that every SIU is sent to other peer nodes at a promised data rate. If an SIU is unavailable in the local SOS, or if downloading from other proxy servers is infeasible owing to resource restrictions, then the peer node simply sends a request to the original video server for downloading.

3: SIU AND SSA ALGORITHM

SIU is the basic caching unit in SOP. Each SIU comprises several groups of pictures (GOPs), and has a unique identity. A GOP is formed of one I frame, along with the B and P frames before the subsequent I frame. This study assumes that each SIU comprises video data requiring 5 seconds of playback time. SOP adopts an SIU-based smoothing algorithm (SSA), which processes each SIU offline and generates a transmission profile for it. This section presents the smoothing algorithm concept and SSA.

3.1: SMOOTHING ALGORITHM

Smoothing algorithms have previously been proposed to maximize the reduction in rate variability when sending a stored video from a server to a client across a network. Previous smoothing algorithms [5] consider both the buffer constraint of a client and the bandwidth constraint between a server and a client when calculating a feasible transmission schedule S* comprising a set of time intervals and associated transmission rates, indicating the periods over which the server must send in a constant-bit-rate (CBR) fashion.

```
Begin
S* = Optimal_buffer_smoothing (B(t), D(t));
Repeat until all SIUs are processed
{
    BB_profile_i = BB_smoothing (S*, B(t), D(t), SIU list);
    BR_profile_i = BR_smoothing (S*, B(t), D(t), SIU list);
    BRR_profile_i = BRR_smoothing (S*, B(t), D(t), SIU list);
    Output
    (SIU_Profile_i
    = {BB_Profile_i, BR_Profile_i, BRR_Profile_i});
}
End
```

Figure 3. SSA algorithm logic flow.

Figure 2 shows an example of a feasible transmission schedule S* that does not cause the client to starve (indicated by A(t) < D(t), where A(t) denotes the cumulative amount of data arriving at the client over [0,t], and D(t) denotes the cumulative amount of data consumed by the client at t) or overflowing (indicated by A(t) < B(t), where B(t) denotes the maximum cumulative data that can be received by the client over [0, t]) during playback. However, S′ is infeasible since it hits B(t), i.e. overflows the client buffer, during video playback. The example in Fig. 2 is a buffer constraint smoothing problem. Considering the buffer size of a client, a buffer-constrained smoothing algorithm finds the smoothest possible transmission schedule for a video program, while during any interval i of the video program, the schedule must satisfy

$$\forall A_{\max}(i) < B(i), A_{\min}(i) > D(i), i \in [t_{i-1}, t_i] \quad (1)$$

In contrast, the rate-constraint smoothing algorithm assumes that the system bottleneck is not the buffer size but the bandwidth between the source and destination. Thus, the problem is to calculate a feasible transmission schedule with transmission rates less than the bottleneck bandwidth.

3.2: SIU-BASED SMOOTHING ALGORITHM (SSA)

SSA runs three smoothing algorithms to build a profile for an SIU. Each algorithm calculates a feasible set of transmission schedules for an SIU in each resource-constraint transition mode (BB, BR and BRR). Figure 3 shows the SSA algorithm. The optimal buffer-constraint smoothing algorithm [5] is first slightly modified to ensure that the rate of *S** only changes at the start time of an SIU transmission. This modification is trivial and not shown here. The modified algorithm is then applied to video data to obtain a set of feasible piece-wise connected constant transmission schedules *S** before video segmentation. The video is then segmented into SIUs, each of which is processed by the BB, BR and BRR algorithms to construct its profile. The SSA also marks those SIUs with transmission rates above a threshold value as a peak SIU during the smoothing processing. This marking is also stored in the profile, and is utilized during selective caching.

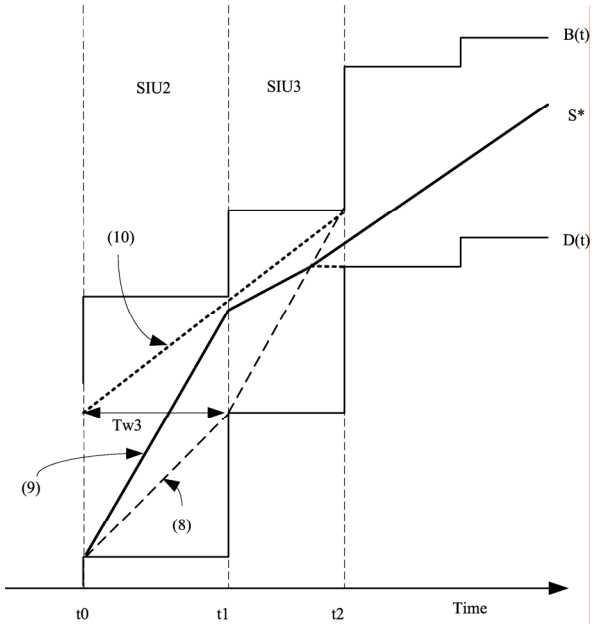

Figure 4. An example of BR transmission schedule.

The following sections describe the three algorithms, namely BB, BR and BRR, that comprise the SSA algorithm.

3.2.1: BB Algorithm. The BB algorithm assumes that only the buffer is a constraint in SIU video data transmission, and readily calculates the transmission schedule for an SIU following *S**. Given the start playback time t_s of SIU_i, the BB algorithm calculates the pre-load time $T_{w\#0}$ and the corresponding downloading rate BB_rate with respect to *S**. The BB_rate is also compared with a peak rate threshold to generate a peak status bit. If the BB_rate is larger than the threshold, then the peak bit is set TRUE; otherwise, the peak bit is set FALSE.

3.2.2: BR Algorithm. Given that SIU_{i-1} is sent following *S**, and the rate of downloading SIU_i is a rate-constraint, the BR algorithm calculates a feasible transmission schedule with a minimum permissible transmission rate *R** for SIU_i without breaking smooth playback. The BR algorithm tries to begin pre-loading SIU_i early at a rate of *R*, so that the finishing transmission time of SIU_i is the same as that of the transmission schedule *S**. To simplify the transmission schedule, the BR algorithm calculates a minimum permissible transmission rate *R**, assuming that the preloading time T_w is made as large as possible but not larger than the playback time of an SIU, and then gradually reduces T_w and hence the transmission rate until a feasible rate is found. Figure 4 illustrates an example of *R** in which T_{w3} equals the playback interval of an SIU in which the cumulative line (9) associated with *R** does not cause buffer overflow. Figure 5 shows the detailed algorithm of BR, while Table 1 shows the notation adopted in the algorithm.

```
PROCEDURE  BR_Smoothing(S*, B(t), D(t), SIU List)
    LET t_p = playback time of SIU_{i-2}
    LET t_p* = playback time of SIU_i
    LET t_e = {t| A(t) = D(t_i )}
    LET t_s = {t| A(t) = D(t_{i-1} )}
    LET I = SIU sequence number
    BRR_mode = TRUE;
    WHILE t_p≠ t_s
    {
        IF D([t_p, t_s]) ≤ Cumulative data of SIU_i and SIU_{i-1} during [t_p, t_s]
            < B([t_p , t_s])
            BR_rate = S*(t_e) – S*(t_s) / t_e – t_p ;
            Tw#1 = t_p* – t_p ;
            OUTPUT(I, BR_rate, T_{w#1}, BRR_mode);
            RETURN
        END IF
        t_p = t_p + 1;
        BRR_mode = FALSE;
    }
    BR_rate = BB_rate;
    T_{w#1} = 0;
    OUTPUT(I, BR_rate, Tw#1, BRR_mode)
END PROCEDURE
```

Figure 5. BR smoothing algorithm.

Table 1. Summary of BR algorithm symbols

t_s	SIU finish transmission time
t_e	SIU start transmission time
t_p	SIU playback time
BR_rate	Minimum transmission rate
$[t_p, t_s]$	Time interval from t_p to t_s
$T_{w\#1}$	Early pre-loading time before SIU playback
SIU List	All SIU profiles for the video program

3.2.3: BBR Algorithm. The BRR algorithm extends the BR algorithm to improve the SIU schedulability, by restricting the transmission rate of the preceding SIU to make the current SIU pre-loadable under a low rate constraint. Intuitively, considering buffer constraint B(t), an SIU may be downloaded at a transmission rate below its BR_rate if the preceding SIU is also downloaded in the BR_rate.

The BRR algorithm first calculates a new cumulative curve associated with the preceding SIU downloaded in BR_rate. A test similar to the BR algorithm is conducted for the current SIU to try to find a lower feasible transmission rate than its BR_rate. If a lower rate is found, then the new BRR_rate and the start transmission latency are stored in the respective SIU profile. To lower the computation overhead, the BRR algorithm only tests whether current SIUi can be pre-loaded at the playback time of SIUi−2. Figure 6 shows the BRR algorithm in detail.

After SSA finishes processing a film, an SIU profile is generated for each SIU consisting of an SIU serial number, the transmission rate and respective startup transmission latency of each transmission mode, and a peak bit. The peak bit is referred during selective caching when a peer determines whether an SIU should be cached at a higher priority than a regular SIU. The profile of each SIU for a film is saved in a file in the

```
PROCEDURE  BRR_Smoothing(S*, B(t), D(t), SIU List)
    LET t_p = playback time of SIU_{i-2}
    LET t_p* = playback time of SIU_i
    LET t_e = {t| A(t) = D(t_i )}
    LET t_s = {t| A(t) = D(t_{i-1} )}
    LET I = SIU sequence number
    Compute A*(t) of SIU_{i-1} with BR_rate;
    Compute A*'(t) in [t_p, t_s]
    = A*(t) + SIU_i data starts downloaded at t_p
      in Rate = S*(t_e) – S*(t_s) / t_e – t_p ;
        IF D([t_p, t_s]) ≤ A*'(t) during [t_p, t_s] < B([t_p , t_s])
            BRR_rate = Rate;
            Tw#2 = t_p* – t_p ;
            OUTPUT(I, BRR_rate, Tw_{#2})
            RETURN
        END IF
End of Procedure
```

Figure 6. BRR smoothing algorithm.

video server. This file is then downloaded to the SOS, and thence to the peers watching the film. According to the SIU profiles, peers can schedule their SIU downloading services from those peers that provide proxy services.

4: SIU CACHING REPLACEMENT ALGORITHM

Peers in a P2P network can join and leave dynamically. In particular, peers are generally end-hosts with heterogeneous and limited capability and up/down link bandwidth. Hence, a peer probably only caches some of the SIUs that it had played since it started accessing a video program.

This section presents four different SIU caching strategies, namely access-unaware, local access-aware (LAA), regional access-aware (RAA) and global cooperative caching (GCC), to enable a peer to determine which SIU is most valuable and which SIU should be replaced when its buffer is full. In the access-unaware and LAA strategies, a peer makes caching and replacement decision alone, without any information from other peers. These strategies are simple with little overhead. Conversely, in the RAA and GCC methods, a peer adopts the SIU caching information and other peers' behavior collected by SOS for SIU caching and replacement decision. RAA and GCC only cause small overhead on the SOS, but potentially improve the view of the peer access pattern and the effectiveness of caching. For the convenience of discussion, Table 2 defines the variables adopted in the descriptions. Note that the residual time of an SIU is defined as the residual playback time of a film in the considered peer based on the assumption that the peer watches the entire film and then leaves the system immediately.

Table 2. Notations for caching strategies.

C_f	Free cache size
N	The number of SIUs in a film
SP_i	SIU_i priority
S_i	The size of SIU_i
LNP_i	The number of peers in the SIU_i's revenue window recorded locally in a peer.
CNP_i	The number of peers in SIU_i's revenue window recorded by SOS.
R_i^b	SIU_i's BB_rate.
B	Peer's uplink bandwidth.
RT	Residual time of the film.

4.1: ACCESS-UNAWARE CACHING

In the Access-Unaware Caching, a peer does not track whether other peers have requested downloading service from it, hence its name. A peer receiving a new SIU_i, caches it if the following rules are satisfied:

1. SIU_i is not downloaded from a proxy peer (i.e., it is from the video server)
2. $B \geq R_i^b$
3. $C_f \geq S_i$

The above rules indicate that an SIU is cached by a peer only if it cannot be downloaded from other peers, the peer is capable of offering downloading service for the SIU, and the peer has a large enough cache for the SIU.

4.2: ACCESS AWARE CACHING

Access aware schemes define a new priority function for SIUs. This priority function is based on the assumption that users do not always watch the whole film in video playback, meaning that the earlier SIUs have higher access rates in the film. The caching priority of an SIU is defined as follows:

$$SP_i = 1 - (i/N)^2 \qquad (2)$$

The priority function gives larger priority values to SIUs with smaller sequence numbers. However, the priority value decreases quickly as the sequence number rises.

4.2.1: LOCAL ACCESS AWARE ALGORITHM. In the local access-aware (LAA) method, each peer logs its own peer access information and adopts the following cost function to evaluate SIU_i as a reference in the caching and replacement decision.

$$V_i = SP_i \times (LNP_i \times R_i^b) \qquad (3)$$

The cost function gives priority to SIUs that potentially provide the highest bandwidth savings on a server.

In LAA, each proxy peer determines SIU caching and replacement on its own and proxy peers do not cooperate with each other. LAA provides a relay-and-forward service to other peers that have previously requested SIU downloading via the cost function defined in Eq. (3), and also offers prefix and selective caching service as in the access-unaware method.

In LAA, a peer initially performs prefix caching and caches the highest possible number of SIUs. When it does not have enough caching space to cache current SIU, it calculates the value of cached SIUs with Eq. (3), and then adopts a value-space tradeoff function to search for SIUs for replacement. The value-space tradeoff function gives priority to SIUs with high value-to-space ratios. LAA first sorts the cached SIUs in an ascending order of priority values, and then includes SIUs in their order into a replacement set if their accumulative value is below current SIUs. Meanwhile, LAA adds up the sizes of SIUs in the replacement set. If it finds sufficient memory space before the accumulative value of the SIUs exceeds the current SIU, then it replaces all SIUs in the replacement set with the current SIU; otherwise, it does not cache the current SIU. Therefore, LAA attempts to maximize the total value of cached SIUs in cache replacement.

4.2.2: REGIONAL ACCESS AWARE ALGORITHM. The regional access aware (RAA) method makes SOS record chronological peer access time in a region to help peers making caching and replacement decision. Based on the access information, a peer calculates each SIU's CNP and value from the access information with the following function:

$$V_i = SP_i \times (CNP_i \times R_i^b). \qquad (4)$$

The RAA algorithm works in the same way as the LAA algorithm. In RAA, each peer replaces SIUs with small V to maximize the total value of cached SIUs in cache replacement.

4.2.3: GLOBAL COOPERATION CACHING ALGORITHM. The global cooperation caching (GCC) method makes proxy peers cooperate in replacing caches. The caching information published by SOS can show the number of copies an SIU cached in the proxy peers within a cluster. SOS can also provide information about each proxy peer's capacity. Thus SOS can calculate the total duplicate capacity of SIU_i in a cluster as follows:

$$DC_i = \sum_{k=1}^{m} B_k / R_i^b. \qquad (5)$$

where m denotes the number of copies of SIU_i cached in the cluster, and B_k denotes the uplink bandwidth of a proxy peer that has cached SIU_i. The value of an SIU is then given by

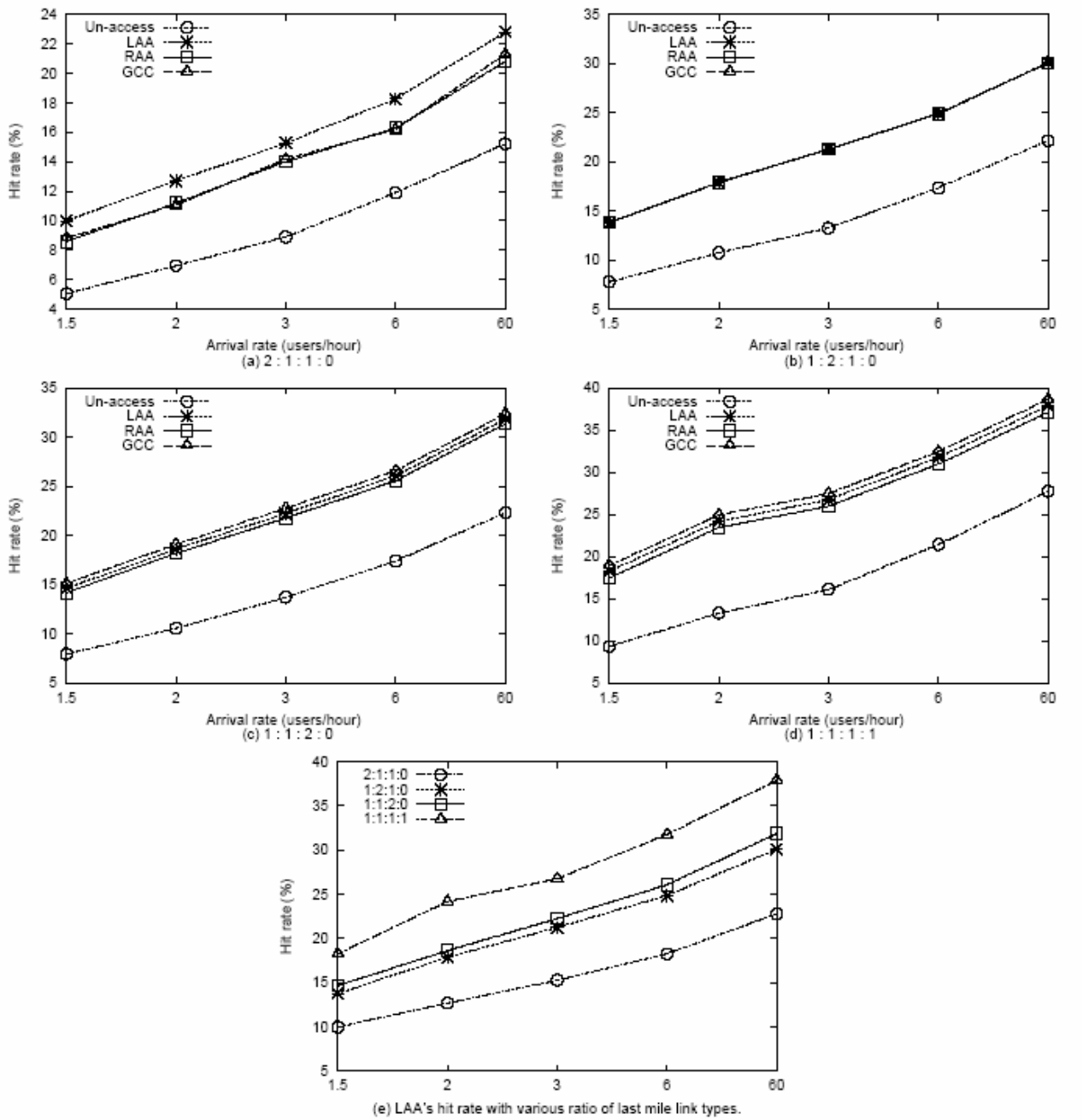

Figure 7. Performance of caching and replacement schemes using the single source scheme with various ratio of last mile link types.

$$V_i = SP_i \times [(CNP_i \times R_i^b) / DC_i]. \qquad (6)$$

According to the above cost function, the value of an SIU is inversely proportional to the number of copies that it is cached in a cluster. Therefore, the peers in a cluster can cooperate to avoid caching too many copies of an SIU unless the SIU has a very high value in a proxy service.

5: PERFORMANCE ANALYSIS

This section describes the performance of an SOP in a cluster-based P2P network using computer simulation. In the simulation, a random number generator was adopted to generate the GOP sizes. The generated playback rate of the film was around 1.5Mbps per second, close to that of a regular MPEG2 video. The produced film was around 50 minutes long, and was segmented into 650 SIUs, each with a playback time of 5 seconds.

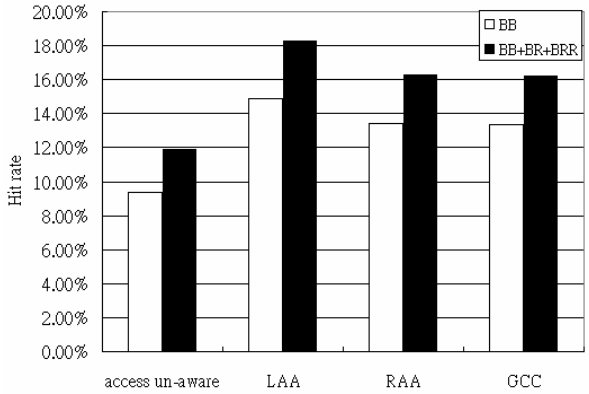

Figure 8. Effect of transmission mode on SIU hit rate.

A cluster-based P2P network with cluster sizes 200 was adopted to test the performance of SOP. Every peer in a cluster shares the same SOS for a video program. Four uplink bandwidth sizes, 512Kbps, 1.5Mbps, 2Mbps and 4 Mbps, with a default ratio of 2:1:1:0, were utilized to test the impact of various transition modes of BB, BR and BRR. In the simulation, it is assumed that each end host uses 64Mbytes buffer for SIU caching.

Figure 7 shows the performance of each caching replacement algorithm using a single source method under different last-mile technologies. The simulation adopted different ratios of last-mile link types to investigation the performance of each cache replacement algorithm. The simulation results indicate that when user arrival rate increases, the cache hit rate also increases accordingly. Conversely, the SIU hit rate rises when many large bandwidth links are adopted since more peers can offer SIU proxy service in the BB mode. This SIU hit rate variation can be learned from Fig. 7(e), which summaries the SIU hit rate of LAA under various arrival rates and combinations of last-mile technologies.

The results in Fig. 7(a)-(d) indicate that the access-aware methods had a 5–8% higher hit rate than the access-unaware methods. Figure 7(a) shows that LAA had the highest hit rate when more users adopt small bandwidth links. However, as the number of high-bandwidth links increased, the cache hit rate of GCC improved gradually, and eventually outperformed LAA as displayed in Fig. 7(b)-(d). The simulation results also reveal that RAA performed slightly worse than LAA when using many high-bandwidth links. This is because the peers in RAA tend to make caching decisions similar to each other, resulting in significant duplicate SIU caching. Finally, Fig. 7(d) shows that the access-aware methods have hit rates of over 35% hit rate at a high user arrival rate, which means that the proposed access aware schemes can reduce up to 35% of server load by the proxy services provided by peer users, and therefore can substantially raise the service capacity of a VoD server.

Figure 8 shows the effect of the smoothing transmission mode on the proxy hit rate. The hit rate of the SOP system using BB, BR and BRR mode is 2-4% higher than that using only BB mode. Thus, employing SSA to pre-process a film can enhance the performance of SOP.

6: CONCLUSIONS AND FUTURE WORK

Overlay networking must face heterogeneity of computing capacity and access link of end hosts in continuous media transmission, especially when end hosts provide proxy services. This study has presented an SIU-based overlay proxy service (SOP) to solve the buffer and link bandwidth constraint of end hosts, and to support true VoD services. An SIU-based smoothing algorithm (SSA) has been developed to obtain a transmission profile for each SIU by offline processing SIUs with respect to buffer and bandwidth constraint. Moreover, a set of distributed caching and replacement strategies has been proposed to improve the SIU hit rate. Extensive simulation results indicate that SSA with the access-aware caching approach can significantly improve the proxy hit rate, and can lower the server load by 15-35% server load at various user arrival rates. The extensive simulation experiment has demonstrated that SOP provides a satisfactory solution to exploit end-host resources for true VoD service.

ACKNOWLEDGMENT

This paper was sponsored in part by ``Aim for the Top University Plan'' of Yuan Ze University and Ministry of Education, Taiwan, R.O.C., and the National Science Council, Taiwan, R.O.C. under Contract No. NSC-95-2213-E-155-005

REFERENCES

[1]. Y. Chu, S. Rao, and H. Zhang, A case for end system multicast, In Proc. ACM SIGMETRICS' 00, pp. 1–12, 2000.

[2]. T. Y. Chung and Y. D. Wang, D^2MST: A shared tree construction algorithm for interactive multimedia applications on overlay networks, IEICE Trans. Commun., vol. E88-B, no. 10, pp. 4023–4029, Oct. 2005.

[3]. A. Bestavros and S. Jin, OSMOSIS: Scalable delivery of real-time streaming media in Ad-Hoc overlay networks, in Proc. ICDCSW 03, pp. 214–219, 2003.

[4]. Y. Cui, B. Li, and K. Nahrstedt, oStream: Asynchronous streaming multicast in application-layer overlay networks, IEEE J. Sel. Areas Commun., vol. 22, no. 1, pp. 91–106, Jan. 2004.

[5]. L. Gao, Z.-L. Zhang, and D. Towsley, Proxy-Assisted Techniques for Delivering Continuous Multimedia Streams, in IEEE/ACM Trans. Netw., vol. 11, no. 6, pp. 884–894, Dec. 2003.

[6]. Z. Xu and Y. Hu, Exploiting Spatial Locality to Improve Peer-to-Peer System Performance, in Proc. WIAPP' 03, pp. 121–125, 2003.

[7]. I. Stoica, et al., Chord :a scalable peer-to-peer lookup protocol for Internet applications, IEEE/ACM Trans. Netw., vol. 11, no. 1, pp. 17–32, Feb. 2003.