COHORT-SURROGATE-ASSOCIATE: A SERVER-SUBSCRIBER LOAD SHARING MODEL FOR VIDEO-ON-DEMAND SERVICES

***Phooi Yee Lau*[1,2]*, Sungkwon Park*[1*]*, and Joohan Lee*[1]**
[1]Media Communications Laboratory, Hanyang University,
17, Haengdang-dong, Seongdong-gu, Seoul 133-791, Republic of Korea
{laupy, sp2996, hitch100}@hanyang.ac.kr
[2]Universiti Tunku Abdul Rahman,
Faculty of Information and Communication Technology, Kampar, Malaysia[1]

ABSTRACT

This paper outlines a proxy-subscriber-subscriber approach, named cohort-surrogate-associate (CSA) architecture, to address three main concerns inherent in a video-on-demand (VoD) service: delivery performance, video traffic, and playout delay. The model coerced coordination among subscribers, whereby every subscriber has a list of video data units stored in their set-top-box (STB) as well those that are stored in other subscriber's STB within the same network edge. This way, a VoD request can be systematically offloaded to other subscriber(s) cached with the same video data units, within the same network edge, instead of requesting the video data units from a proxy or remote server. The CSA architecture may be conceptualized as a composition of three 'layer', the associate being the lowest layer, followed by the surrogate and the cohort. The cohort is responsible for grouping associate(s) with same viewing interest together - i.e. watching the same quasi-genre video. The surrogate is responsible for coordinating the delivery of video data units among associate(s) at the same network edge, when needed. All associate(s) in the same network edge can be the contributors of the video data unit. This paper formally defines the role of cohort, surrogate, and associate under different scenarios for VoD services.

Keywords: video-on-demand, video delivery, content delivery network, set-top box, time-weighted video popularity, user interest, load balancing

1.0 Introduction

An increasing demand for multimedia-rich Internet content, inspurred by the growth of information, has rapidly generate a lot of interest in developing a more efficient content delivery method, especially for multimedia streams. The development of broadband access is slowly gaining increasing prominence especially for streaming multimedia-rich contents such as video-on-demand (VoD). Unlike bulk-data transfer, video delivery, in general, is comparatively intolerant to delay, and requires a minimum and continuous guaranteed bandwidth to support the required data rates. Moreover, VoD involves streaming pre-encoded contents and hence contents availability and waiting time are viable for the success of any commercial VoD services. It is important, thus, to design a method that could efficiently deliver videos while lowering network cost for service operators in a win-win situation.

Content Delivery Network (CDN), first adopted by service operators, is based on hierarchy of servers located at specific places of the network – often located at the core of the network. If one server fails, the same content should be available in the other servers at the upper hierarchical levels, insinuating that there will be additional cost to maintain video availability. If a server's capacity is fixed and no storage spaces are available, the least popular video will be removed from this server. Due to this, much recent work focuses on designing VoD systems that are viable for the on-demand services in broadband context [1-2]. For example, the peer-to-peer (P2P)-based delivery system, where the computing and bandwidth cost is pushed toward the network edge, is an alternative approach often used for streaming video for large content libraries. In a P2P system, any peers can be the contributors of video contents and bandwidth. Nonetheless, this architecture is not popular with Telco as it consumes too much bandwidth and it does not fit well into their business model, especially when it comes to managing the video's digital right management.

A number of video distribution systems using different network dimensioning at different levels of complexity exist. Some of the popular video-on-demand architectures are the distributed VoD architecture, the scalable VoD architecture, the P2P VoD architecture and the store-and-forward VoD architecture [3-5]. These architectures need enormous storage facility on broadband-like access with multiple replication capability, on centralized-, distributed-, or peered-based infrastructures. In 2008, Nafaa et al. highlighted and analyzed large-scale VoD service provisioning issues, being: 1) uplink and storage capacities available at STBs, 2) service provisioning, 3) optimization of network resources, and 4) request rejection rates. They presented a highly scalable architecture for P2P-based VoD services provisioning which effectively translated popularity distribution throughout the overall content library into relative content availability in the network to maximize network resources utilization by reducing the overall number of VoD request rejection. This architecture, however, disregarded the subscribers' demography for a given service area. Moreover, the P2P engine used is unpopular among service providers.

Partial preloading, a technique attempt to allocate a part of large disk storage to cache video data units at the user's STB, is an improved solution to the prior [6-7]. Assuming a disk with enough storage to store 40 minutes of video, partial preloading would enable the STB to keep the first 5 minutes of the top 8 videos. However, allocating all disk space to preload videos would probably eliminate the extra bandwidth required to guarantee jitter-free delivery of compressed video data units but may not fully guarantee immediate or uninterrupted service, especially if those stored video data units are not of user's interest. Additionally, disk spaces in the set-top-boxes (STB), most of the time, are left empty and are not being allocated to keep watched/streamed video data units. Those storages, in minimum capacity, are used to store only incoming video data to increase streaming efficiency (jitter-free) after user(s) selects to play a video. As such, there is a strong need, from a user's and a service provider's perspective, to provide an adequate and widely disperse storage system that could carefully managed disk spaces, especially, 1) to cache incoming video data units, 2) to enable not so rigid playback deadlines when network resources are constraint, and 3) to enable continuous media delivery during peak time [8].

In this paper, a new server-subscriber approach, named cohort-surrogate-associate (CSA) architecture, is presented. The focus of this model is to address the scarcity of network resource at the backbone network and the scalability issues inherent in VoD system. It comprises of: 1) associate – which is responsible for preloading video data units in its STB based on a video popularity model using a time element; 2) surrogate – which is responsible coordinate video data unit transfer among associate(s); and 3) cohort – which is responsible for grouping together associate(s) at the same network edge. The CSA architecture is built upon two important resources exploitation: 1) unused capacity in the STBs, 2) unused bandwidth at the network edge. Partially preload video data units into the STBs require a decision making, i.e. selecting which video to be partially preload. Therefore, this paper uses a time-weighted popularity index, proposed earlier, which is guided by a user's viewing interest and time element [10]. The CSA architecture coordinates and manages the delivery of video data units at the network edge among subscribers viewing the same video (same viewing interest). It extends the capabilities of the network by exploiting the unused bandwidth in the network edge, especially when network backbone resources are scarce. The *Playlist* allows a subscriber to freely select a video to be play. It is an application which lists all available videos and their mirrored locations, and is updated based on subscribers' recent access patterns or interest, within the subscribers' profile.

We devote the next section to briefly describe the video popularity index. In Section 3, we describe and discuss the proposed network structure for the proposed CSA architecture. Section 4 outlines the responsibility of cohort and associate while Section 5 outlines the responsibility of surrogate. Section 6 discusses how *Playlist* is organized and Section 7 concludes the paper with future work.

2.0 Video Popularity Index

A time-weighted popularity (TP) index is built by analyzing the viewing interest of the subscribers, in time [10]. It studies the popularity of videos based on the time each subscriber spent watching a specific video with respect to the total hours the subscriber logged. This index is used by the VoD system to push videos to respective storages, i.e. the STBs or the servers. TP index is generated through calculating the index for each subscriber log-in for a particular video - as shown in (1).

$$TP_{x(k)} = \sum_{i=1}^{n} \frac{t_w(i)}{t_d} \qquad i \in \{1,2,3 \ldots n\} \tag{1}$$

$x \in \{subscriber, proxy\ server, remote\ server\}$, k is the video code, n is the number of times subscriber(s) logged in to watch a video, i is a counter, t_w represents the duration subscriber(s) watch a video and t_d represents the total hours logged by subscriber(s).

3.0 CSA Architecture

In a VoD service, contents are normally delivered to subscribers through different server elements and different network provisioning aspects. It is worth noticing that in most video delivery network, the underlying network architecture is already designed as hierarchical networks. The difference lies in the service provider's network dimensioning strategy. – i.e. the server-server connections and the server-subscriber connections. It is important, therefore, to design a VoD system that could efficiently deliver on-demand video requests to subscribers with minimum delay, and at lower network cost for service operators, in a win-win situation. Moreover, the success of VoD depends heavily on advertising and promotion, plus the ability to deliver "anytime" video experience to consumer and offering them a wide variety and the right mixture of content including new and popular movies. Often, proxy servers are employed to reduce network traffic and delays, as proxy servers are located closer to the network edge and are connected to the remote server through high speed broadband connection. However, proxy servers have a finite storage capacity. Therefore, a dynamic management scheme has to be employed for selecting 'appropriate' videos to be cached.

Fig. 1 shows the network structure of our proposed work. Videos can be streamed not only from remote servers but also from proxy server, thus, are able to provide a significant benefit, i.e. reduction of video traffic at the backbone network and minimize playout and start up delay.

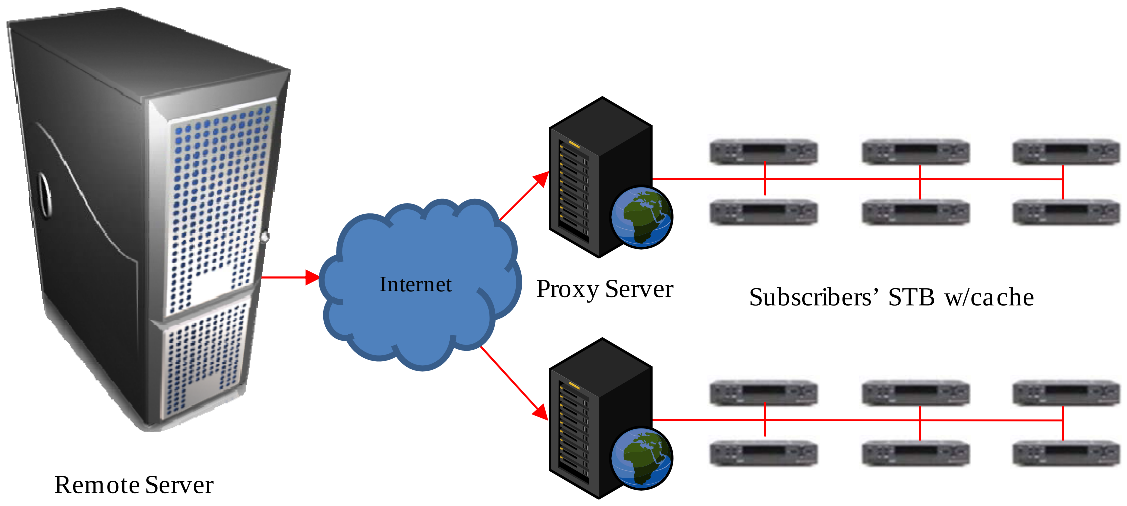

Fig. 1: Network structure for CSA architecture

Let us consider a video with *D* minutes of video which is equally divided into *n* video data unit – see Fig. 2. This video is divided into two segments: 1) initial video data unit, *x*; and 2) subsequent video data units, *y*. The initial video data unit is cache in the STBs to allow subscriber to watch a video instantaneously when they click 'PLAY', even when there are network congestions and a lack of network bandwidth which delays the video data unit being streamed to the subscriber's STB. The rest of the video data units are delivered through the multicast protocol to the STBs to ensure the whole video content can be played out continuously without delay [9].

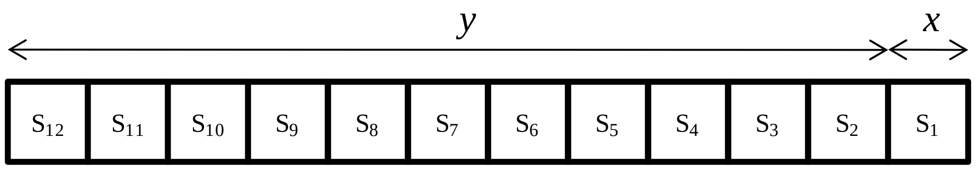

Fig. 2: Video data units (n = 12): a) *x* is the initial video data unit and *y* is the remaining video data units

Each subscriber maintains a small amount of cache to buffer and to store all initial video data unit(s) based on the video popularity [10]. In this instance, because the STBs have limited cache resources, only selected video data units can be stored. Since these cached video data units are also used to serve other subscribers within its proximity, the identity of the video needs to be managed within the network edge. A novel proxy-subscriber-subscriber approach is proposed, named the cohort-surrogate-associate architecture. It exists within all subscribers and is used to manage all video data units cached in the subscribers' STB at the network edge. The following sections further explain the design and rules for the CSA architecture.

Fig. 3 illustrates the relationship among subscribers in the CSA architecture within the same network edge. It aims to reduce video traffic at the backbone network and to lower the playout delay experience by all subscribers. The CSA architecture may be conceptualized as a composition of three 'layer', associate being the lowest layer, followed by surrogate and the cohort. For the ease of exposition, associate are active subscriber(s). Cohort represents a common viewing interest; while surrogate coordinate file transfer for associate(s).

The CSA architecture derives from peer-to-peer system designs that range from tree-based schemes, such as NICE [11] and ZIGZAG [12], to mesh-based schemes [13], such as Narada [14], CoolStreaming [15], PRIME [16], PPStream, PPLive, and SopCast. However, our proposed architecture is different from the above in that we did not build the peer-to-peer architecture. Instead, we focus on how to optimally allocate, balance and replicate video data units for all associate(s) manage by a surrogate within a cohort in order to minimize video delivery time especially during congestions. In other words, we optimized the performance of delivery system by dynamically regulating subscriber's request over the network edge. There are a few challenging issues in the initial stage of conceptualizing the CSA architecture, i.e. request congestion, insufficient bandwidth capacities in the network, and frequent network dynamics changes due to subscribers' turning off or power failures. In the following sections, we discuss the responsibilities of the cohort and associate, as well as, the surrogate's management and responsibilities, taking into account the previous issues discussed.

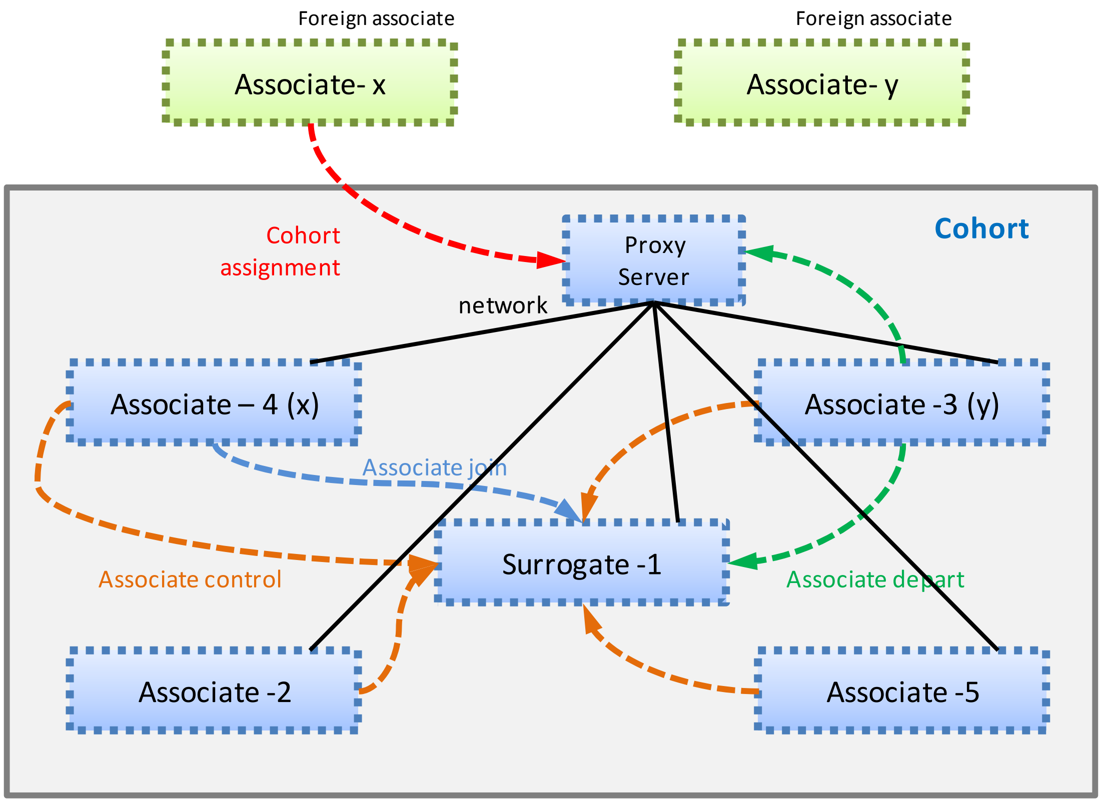

Fig. 3: High level description of CSA architecture

4.0 Responsibility of Cohort and Associate

Fig. 4 shows the high level description of the CSA architecture. In this section, we discuss the roles and responsibilities of the cohort and the associate in the proposed CSA architecture – that is cohort assignment, associate join, associate depart, and associate control.

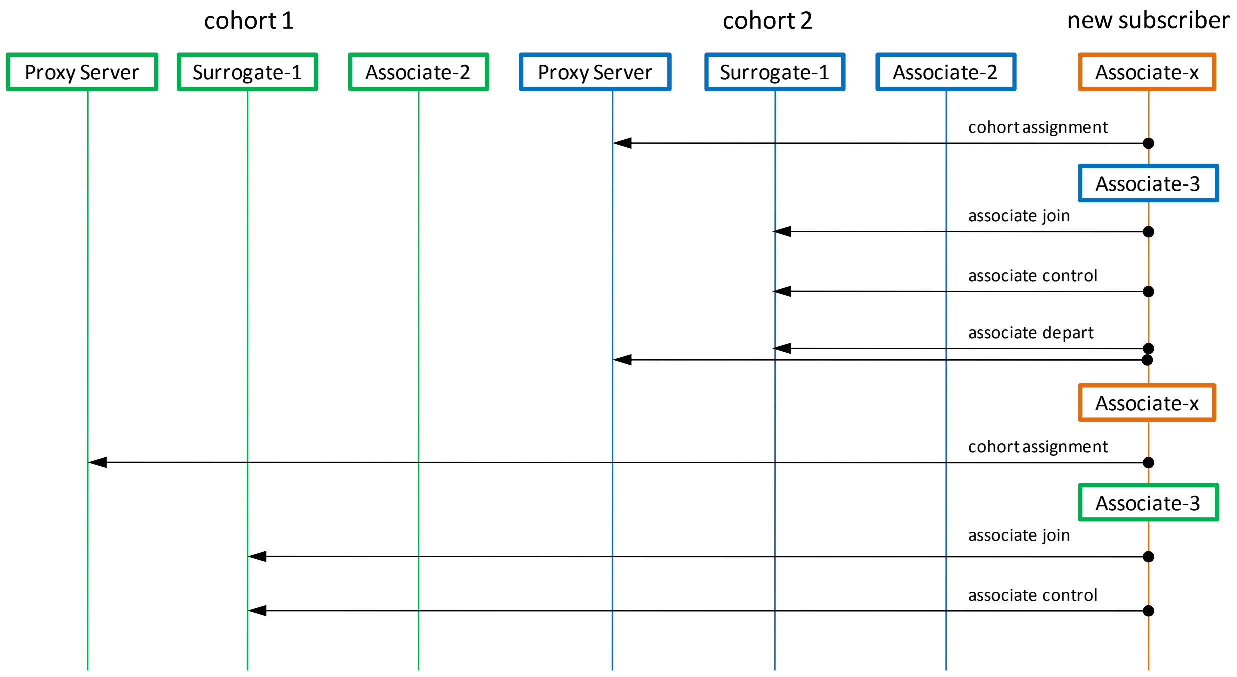

Fig. 4: High level description of the CSA architecture

4.1. Cohort Assignment

A cohort is defined as a group of associate(s) with similar quasi-genre profile. Initially, the cohort is assigned based-on the quasi-genre profile, proposed in [10] - referred to as COHORT ASSIGNMENT. Subscriber with the same quasi-genre profile at the network edge are grouped together within a cohort, in order to offload proxy server burden and to provide quick delivery of any video for other associate(s) STBs. Our idea is that those in the same cohort – i.e. same watching behavior may prefer watching similar videos and associate(s) within the same cohort can receive video data units from each other. The terms used to address the associate are: 1) surrogate: manager of associate(s), 2) associate: member of a cohort; 3) foreign surrogate: manager of foreign associate(s) for foreign cohort; and 4) foreign associate: foreign member of the cohort, as described and shown in Fig. 5.

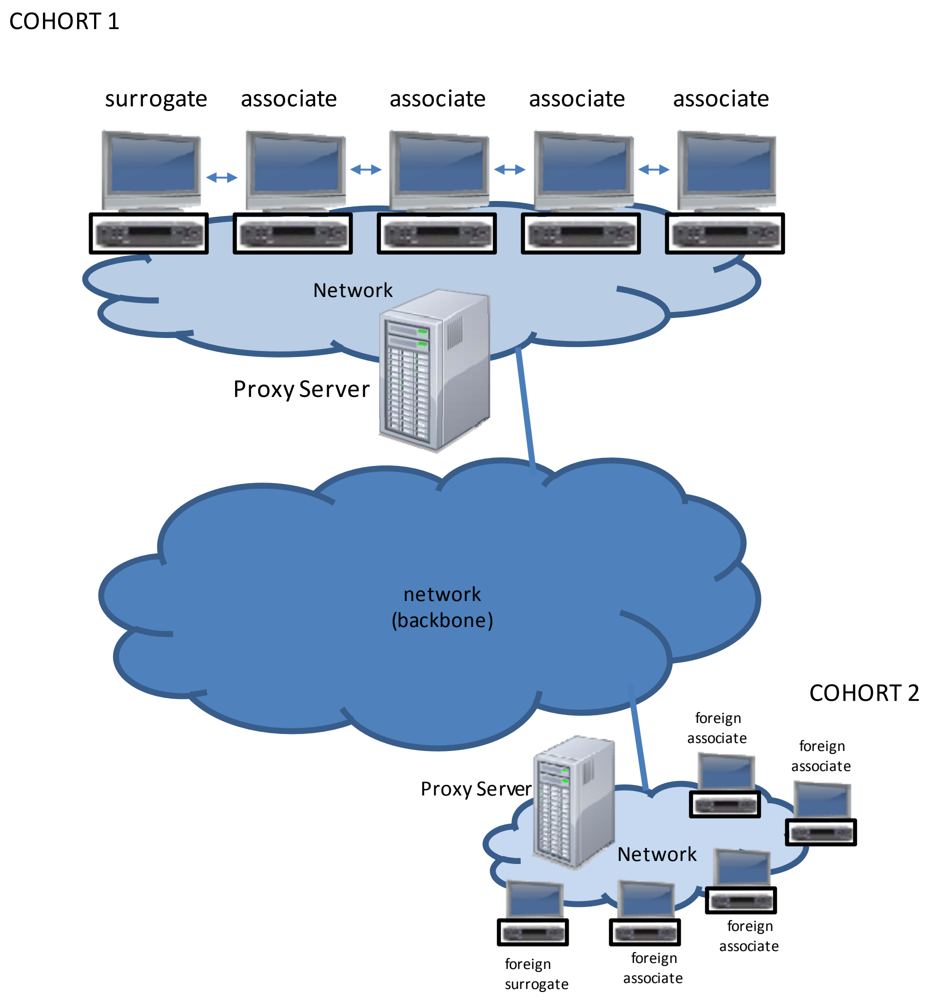

Fig. 5: CSA-Subscriber: Cohort relationship and assignment

4.2. Associate join

A subscriber performs ASSOCIATE JOIN when the subscriber switch-on the STB for the first time, based on the profile registered, and for every other time the STB is switch on thereafter. If the *Playlist* is empty, the associate perform the ASSOCIATE CONTROL and proxy server starts to update the top 10 *Playlist* into the STBs. The associate need to send the following information to the surrogate: 1) playlist; 2) address; and 3) status; for cohort management.

4.3. Associate depart

An associate can leave a cohort and join another foreign cohort, i.e. proxy server. Two types of common associate depart are: 1) Case1: unexpected failure which is caused by power failures or subscribers turning off the power; and 2) Case 2: expected call which is caused by the desire of the associate to leave the cohort to join a foreign cohort. We discuss the following cases, assuming associate *X* in *Y* cohort and foreign *Z* cohort:-

- Unexpected failure: Associate *X* reboots and rejoins *Y* cohort. If the cache in the STBs is not recovered, the ASSOCIATE JOIN is performed and its *Playlist* is updated.
- Expected failure: Associate *X* uses the following steps to attach to foreign *Z* cohort.

1. Associate *X* disengaged itself from *Y* cohort – ASSOCIATE DEPART
2. Associate *X* join a foreign *Z* cohort – COHORT ASSIGNMENT
3. Associate *X* updates its *Playlist* – ASSOCIATE JOIN

4.4. Associate control

In order to maintain the connectivity of the systems, the system requires all associate(s) to perform ASSOCIATE CONTROL and to update the surrogate with latest playlist and status. The following control information is updated: 1) playlist status, and 2) associate status.

5.0 Responsibility and Management of Surrogate

One of the main challenges when designing a VoD service is the ability: 1) to scale up when request peak, 2) to handle a large number of simultaneous requests, and 3) to optimize flow of traffic from the backbone network. We take a closer look at arrival patterns modeled using a Poisson distribution and found that during peak hours [17-18], most VoD servers need to offload VoD request in order to allow playout with minimum delay, reduce the flow of traffic in the backbone network, and reduce request rejection rate. The required reduction in congestion, especially during peak-hours, suggests the possibility of employing P2P architecture at the network edge. Are there any other alternatives? If there are, what would be the best possible style to manage the system and what are their responsibilities? – i.e. the subscriber's location and the number of replicas stored among these subscribers. In this case, the STBs cache may be used to maximize their effectiveness and startup delay by extending their caching capacity to store the initial video data unit. In the following subsection, we explain our proposed work and describe how subscriber's request is offloaded in the CSA architecture, in particular, through the surrogate.

5.1. Surrogate assignment

A surrogate manages and collects associate(s) information in the same cohort. The main idea is to keep track of the associate's *Playlist* in order to share, if possible, the initial video data unit with other associate(s), who has it stored at their STBs, to facilitate speedier access to selected videos – see Fig. 6.

A surrogate can be selected or reselected under various conditions, as listed below.

1. Condition A: When no subscriber is active (no associate / no surrogate).
2. Condition B: When a number of associate(s) join the cohort sequentially.
   a. Associate(s) are indexed by numerical numbers starting from 1.
   b. The first associate will perform the SURROGATE ASSIGNMENT and is assigned as surrogate.
3. Condition C: When the surrogate leaves the cohort
   a. The surrogate leaves the cohort by performing the ASSOCIATE DEPART
   b. The cohort re-assigned the surrogate and performs the SURROGATE ASSIGNMENT, being the next associate.
   c. The surrogate initiates the SURROGATE CONTROL

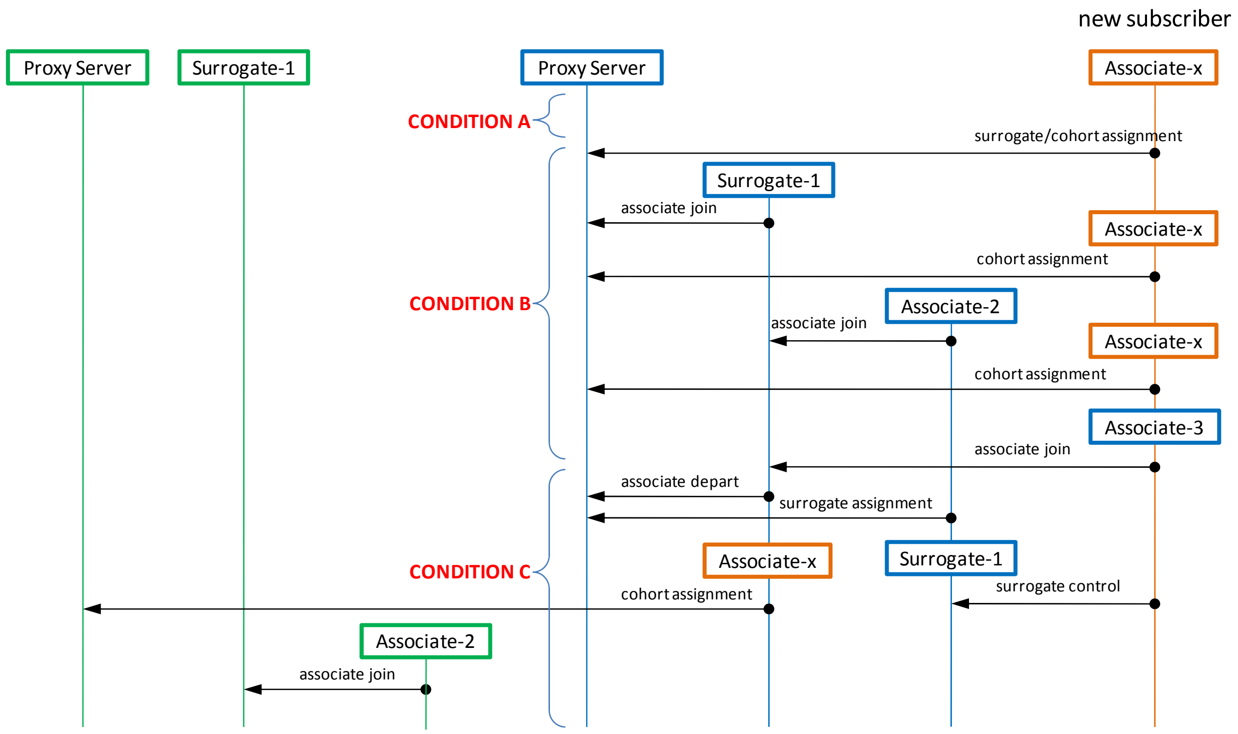

Fig. 6: Sequence diagram of cohort management

5.2. Surrogate control

This is a control message delivered to the surrogate to update associate's status for associate management. Especially, when a surrogate leaves for another cohort, in order to maintain the connectivity of the systems, the new surrogate requires all associate(s) to perform SURROGATE CONTROL. SURROGATE CONTROL message include: 1) associate playlist; 2) associate address; and 3) associate status.

5.3. Surrogate offload

This approach assumes that all surrogates are able to communicate with all associate(s). Surrogate can offload subscriber's request, especially during flash crowd, to other associate(s) within a cohort. As discusses in earlier, a video is aggregated into several video data units, to enable smaller video data units to be delivered faster to all corresponding associate(s). Fig. 7 illustrates the sequence of the offloading process which involves the Associate-2 (being the request party), the Surrogate-1 and the corresponding proxy server.

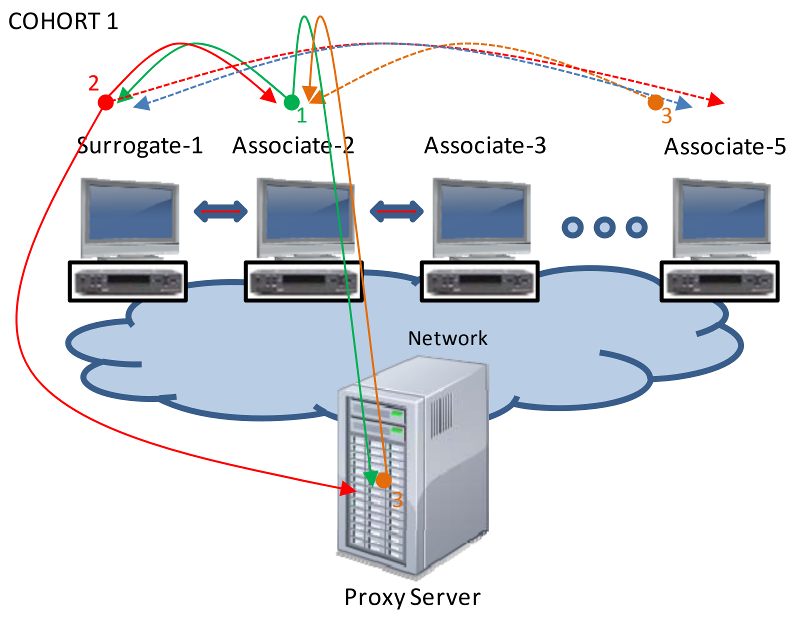

Fig. 7: CSA Architecture: Offloading video request through surrogate and associate

The steps involved in the offloading processing are listed below and shown in Fig. 8.

1. Step 1: The Associate-2 initiates a REQUEST to Surrogate-1 and Proxy Server.
2. Step 2: Two possible scenarios here are as listed below.
   a. Step 2A – Associate-5 with initial video data units: the Surrogate-1 UPDATE the Proxy Server, Associate-2, and Associate-5. (The initial video data unit is available at Associate-5 and remaining video data unit is available at Proxy Server)
   b. Step 2B – No associate with initial video unit: the Surrogate-1 UPDATE the Proxy Server and Associate-2.
3. Step 3: Two possible scenarios here are as listed below.
   a. Step 3A: Associate-5 SEND initial video unit to Associate-2 and Proxy Server SEND remaining video data units to Associate-2.
   b. Step 3B: Proxy Server SEND all video data units to Associate-2.

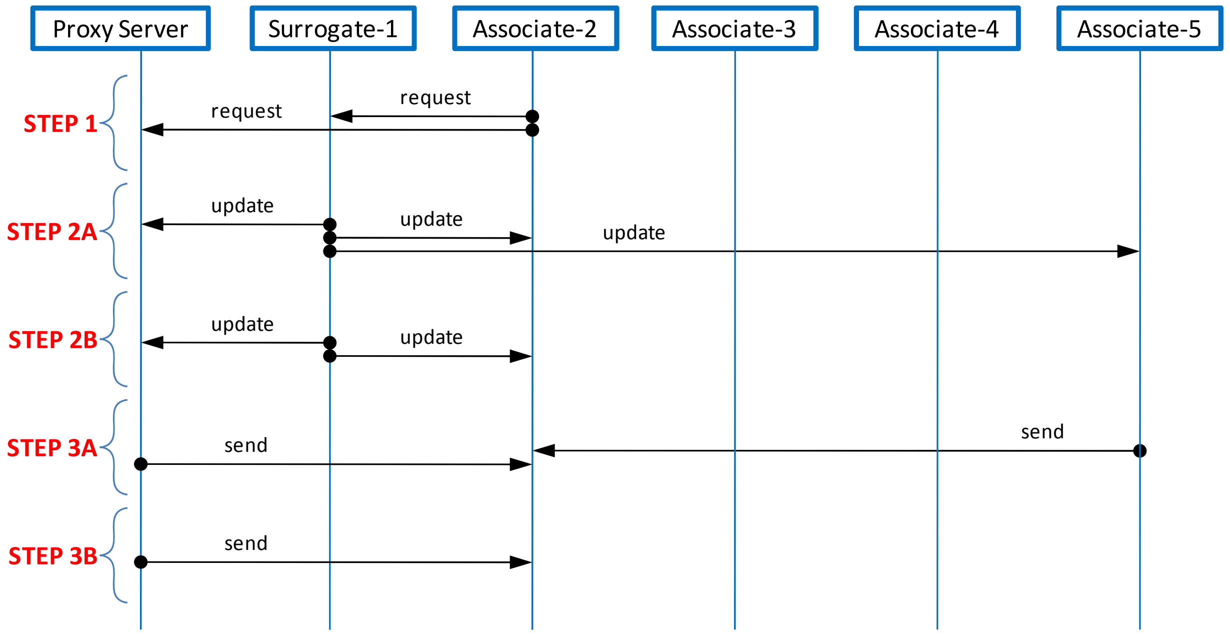

Fig. 8: Sequence diagram for offloading request using surrogate

6.0 *Playlist*

Playlist is an application that displays a list of videos which a subscriber could watch – see Fig. 9. The *Playlist* allows service providers and users the flexibility to select available videos from the VoD server, which is being updated based on the instantaneous video popularity index proposed in [10]. The subscriber can select the video to watch in the *Playlist*, which is being categorized into three different levels, depending on the where those videos are stored, as shown below.

- Level 0 *Playlist* – no startup delay; consist of the list of video data units cached at the STBs based on the $TP_{subscriber(k)}$
- Level 1 *Playlist* – minimum startup delay; consist of the list of videos stored at proxy server – closed to the Subscriber based on the $TP_{proxy\text{-}server(k)}$
- Level 2 *Playlist* – high startup delay; consist of the full list of videos stored at the remote server based on the $TP_{remote\text{-}server(k)}$

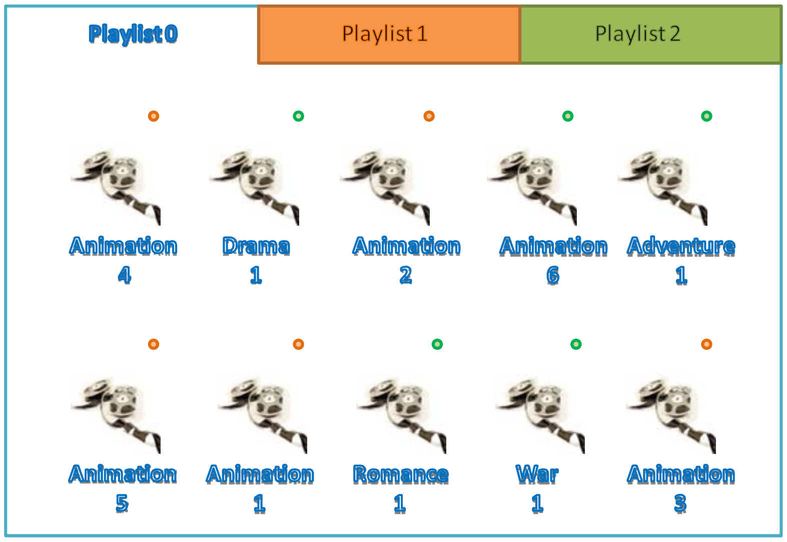

Fig. 9: *Playlist* sample screen

Level 0 *Playlist* is stored at the subscriber's STBs and is used to move popular videos closer to the subscriber based on the $TP_{subscriber(k)}$ derived from the subscriber's viewing interest. It is essential to consider how frequent the content in the STB should be refreshed. Fig. 10 depicts the flow diagram of how the VoD is delivered to the subscriber.

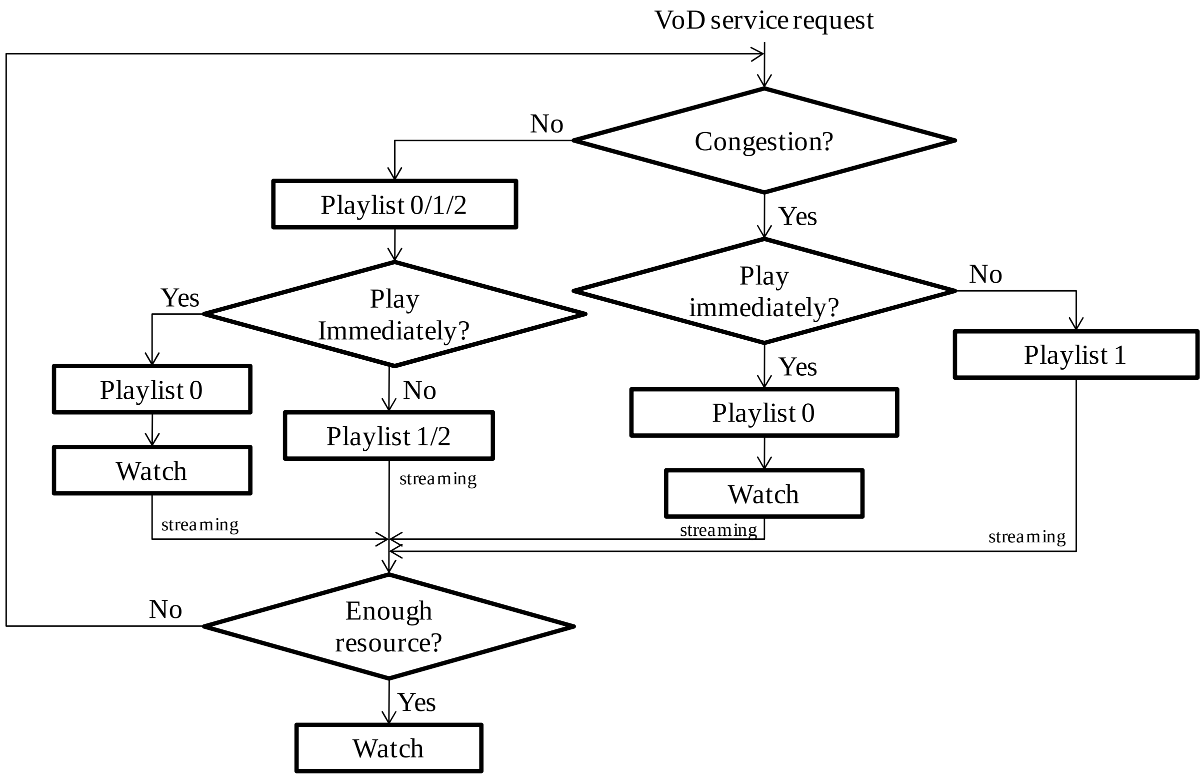

Fig. 10: Active *Playlist* for VoD service request

7.0 Discussion and Future Work

In this paper, we introduce a new load sharing architecture, namely, CSA architecture, which enable subscriber to share their cached video data units with other subscriber within the same network edge. The CSA architecture, though designed in appropriate flavor of P2P networking, focuses mainly on how to optimally allocate, balance and replicate video data units for all associate using a surrogate within a cohort to surrounding associate in order to minimize video delivery time for low startup and playout delay. This architecture only shares a similar concept for peer-sharing of video data units as it involves light coordination between subscriber-subscriber and requires built-in cache. The main responsibility of CSA architecture is to efficiently manage and deliver video data units from the cache at the STBs and in order to reduce the volume of data traffic to/from remote servers. In other words, the CSA architecture mapped VoD requests to the nearest “source”, as much as possible, to reduce request rejection and untimely delay concerns which arise in most traditional VoD system. Furthermore, we employ a time-weighted video popularity-model, proposed earlier, to update video data units at the STBs [10].

We plan to further investigate on deploying the CSA architecture in the IP multicast protocol for a VoD system to improve VoD service and to reduce rejection rate – see Fig. 11. At the moment, notice two implementation limitations of the proposed architecture i.e. requiring: 1) built-in cache system, and 2) cost of actual implementation. Service operators are most reluctant to include built-in cache due to cost and video’s digital right management issues. But due to advances in technology, storage cost is only a fraction of what it was before. And the impact of preloading video data units in the STB has surpassed its disadvantages as subscribers could enjoy zero-delay VoD services. Service providers, while reluctant to provide larger disk space in the STB, could not totally ignore the benefits of bundling extra disk spaces in the STBs. Nonetheless, as far as implementing the proposed model is concerned, we foresee some advantages in terms of system’s flexibility and scalability, and increment of user experience in a win-win situation. Finally, the work here presented can be extended to additionally include improving and modifying the CSA architecture with the possibility of adding a proxy-surrogate server at the network edge to serve subscriber request during congestion – see Fig. 12.

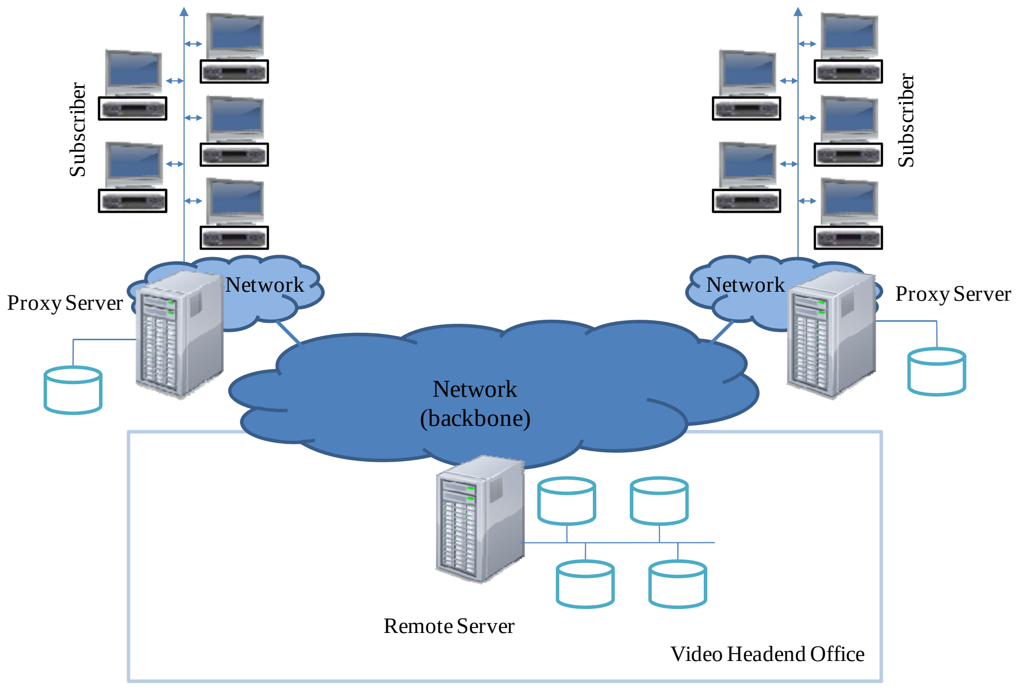

Fig. 11: Prototype of a VoD system

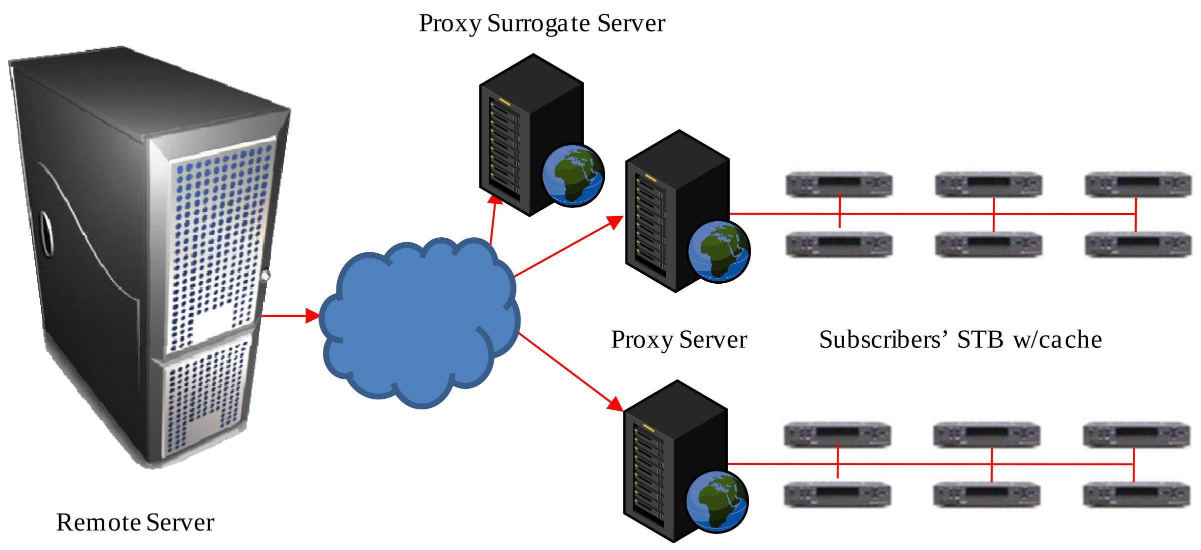

Fig. 12: Extending the CSA architecture with additional proxy-surrogate server

8.0 Acknowledgement

This work was supported by research fund of Hanyang University (HYU-2005-I).

9.0 References

[1] D. Farinha, R. L. Pereira, and T. Vazão, "Performance analysis of VoD architectures", in Proc. International Workshop on Traffic Management and Traffic Engineering for the Future Internet, Porto, Portugal, December 2008.

[2] J. Pâris, S. W. Carter, and D. D. Long, "A Low Bandwidth Broadcasting Protocol for Video on Demand", in Proc. of the International Conference on Computer Communications and Networks, 12-15 October, 1998, Washington, USA, pp. 690.

[3] A. D. Gelman, H. Kobrinski, L. S. Smoot, S. B. Weinstein, M. Fortier, and D. Lemay, "A store-and-forward architecture for video-on-demand service", in Proc. IEEE International Conference on Communications, Denver, USA, June 1991, pp. 842-846.

[4] E. Mykoniati, R. Landa, S. Spirou, R. Clegg, L. Latif, D. Griffin, and M. Rio, "Scalable peer-to-peer streaming for live entertainment content', IEEE Communications Magazine, 2008, 46, (12), pp. 40-46.

[5] X.Y. Yang, P. Hernandez, F. Cores, L. Souza, A. Ripoll, R. Suppi, and E. Luque, "DVoDP/sup 2/P: distributed P2P assisted multicast VoD architecture", in Proc. 20th International Parallel and Distributed Processing Symposium, Rhodes Island, Greece, April 2006, pp.10.

[6] J. Pâris, D. D. Long, and P. E. Mantey, "Zero-delay broadcasting protocols for video-on-demand", in Proc. of the Seventh ACM International Conference on Multimedia (Part 1), October 30 - November 05, 1999, Florida, United States, pp. 189-197.

[7] J. Pâris, “A stream tapping protocol with partial preloading”, in Proc. of the 9th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, August 2001, Cincinnati, USA, pp. 423 - 430.

[8] C. Rodrigues, L. Filho, and R. Leão, “On Scalable Interactive Video-On-Demand Services”, European Journal of Scientific Research, vol. 21, no. 4, pp. 662-686.

[9] E. Lee, and S. Park, “Internet Group Management Protocol for IPTV Services in Passive Optical Network”, IEICE Transactions on Communications, vol. E93-B, No. 2, pp. 293-296.

[10] P. Y. Lau, S. Park, and T. Kim, “Dynamic Time-Weighted Popularity Index: A Video-on Demand Case”, 2010 IEEE International Conference on Network Infrastructure and Digital Content (IEEE IC-NIDC 2010), pp. 809-814, 24-26 September, 2010, Beijing, China.

[11] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, “Scalable application layer multicast”, in Proc. ACM SIGCOMM, Pittsburg, USA, August 2002.

[12] D. A. Tran, K. A. Hua, and T. T. Do, “ZIGZAG: an efficient peer-to-peer scheme for media streaming”, in Proc. IEEE INFOCOM, San Francisco, April 2003.

[13] N. Magharei, and R. Rajaie, “Understanding mesh-based peer-to-peer streaming”, in Proc. ACM International Workshop on Network and Operating Systems Support for Digital Audio and Video, Newport, USA, May 2006.

[14] Y.-H. Chu, S. G. Rao, and H. Zhang, “A case for end system multicast”, in Proc. ACM SIGMETRICS, Santa Clara, USA, Jun 2000.

[15] X. Zhang, J. Liu, B. Liz, and T.-S. P. Yum, “Coolstreaming/ DONet: a data-driven overlay network for peer-to-peer live media streaming”, in Proc. IEEE INFOCOM, Miami, USA, April 2005.

[16] N. Magharei, and R. Rajaie, “PRIME: peer-to-peer receiver driven mesh-based streaming”, in Proc. IEEE INFOCOM, Anchorage, Alaska, May 2007, pp. 1415–1423.

[17] H. Yu, D. Zheng, B. Y. Zhao, and W. Zheng, “Understanding User Behavior in Large Scale Video-on-Demand Systems”, in Proc. of the 1st EuroSys., Leuven, Belgium, April 2006.

[18] M. Zhang, L. Zhao, Y. Tang, J.G. Luo, and S.Q. Yang, “Large-scale live media streaming over peer-to-peer networks through global Internet”, in Proc. ACM Workshop on Advances in Peer-to-Peer Multimedia Streaming, Singapore, November 2005, pp. 21–28.

BIOGRAPHY

Phooi Yee Lau received her B.Sc. from Universiti Teknologi Malaysia, Malaysia in 1996, M. Sc. from Universiti Malaya, Malaysia in 2002, and Ph.D. from Keio University, Japan in 2006. She is currently a researcher at the Media Communications Laboratory, Hanyang University, Republic of Korea, and was previously a researcher at the Image Group of Instituto de Telecomunicações, Portugal, and an Assistant Professor with Universiti Tunku Abdul Rahman, Malaysia. Her current research interests are in image and signal processing, medical image analysis, intelligent system, media communication, and video delivery. She is a member of the IEICE and IEEE.

Sungkwon Park received the B.S. from Hanyang University in 1982, M.S. degree from Stevens Institute of Technology, Hoboken, New Jersey, USA in 1983, and Ph.D. degree from Rensselaer Polytechnic Institute, Troy, NY, USA in 1987, all in Electrical Engineering. He is currently a Full Professor at Hanyang University, Republic of Korea, and was previously an Associate Professor in the Department of Electrical Engineering, Tennessee Technological University, Cookeville, TN, USA. His research interests include digital communication systems, and digital broadcasting. He is a member of the IEICE, and a senior member of IEEE.

Joohan Lee received his B.Sc. from the Department of Electronics and Electrical Computer Engineering, Hanyang University, Republic of Korea in 2006, and his M. Sc. from the Department of Electronics and Computer Engineering, Hanyang University Republic of Korea, in 2008. Currently he is currently working towards his Ph. D degree at the Department of Electronics and Computer Engineering, Hanyang University, Republic of Korea. His current research interests are multimedia communications, IPTV, mobile IPTV, and broadband networks.