San Jose State University
SJSU ScholarWorks

Faculty Publications | Computer Engineering

February 2014

On Optimal Media/Video Distribution in Closed P2P-Based IPTV Networks

Hao Cui
Santa Clara University

Xiao Su
San Jose State University, xiao.su@sjsu.edu

Weijia Shang
Santa Clara University

Follow this and additional works at: https://scholarworks.sjsu.edu/computer_eng_pub

 Part of the Computer Engineering Commons

Recommended Citation

Hao Cui, Xiao Su, and Weijia Shang. "On Optimal Media/Video Distribution in Closed P2P-Based IPTV Networks" *Computer Networks* (2014): 217-232. https://doi.org/10.1016/j.bjp.2013.11.007

This Article is brought to you for free and open access by the Computer Engineering at SJSU ScholarWorks. It has been accepted for inclusion in Faculty Publications by an authorized administrator of SJSU ScholarWorks. For more information, please contact scholarworks@sjsu.edu.

On Optimal Media/Video Distribution in Closed P2P-Based IPTV Networks

Hao Cui
Computer Engineering Department
Santa Clara University
Santa Clara, CA 95053, USA
Email: hcui@scu.edu

Xiao Su
Computer Engineering Department
San José State University
San José, CA 95192, USA
Email: xsu@email.sjsu.edu

Weijia Shang
Computer Engineering Department
Santa Clara University
Santa Clara, CA 95053, USA
Email: wshang@scu.edu

Abstract

Video distribution over the Internet has become a popular service because of technological advances in internet (e.g., higher network bandwidth) and video coding (e.g., H.264/SVC). In this and other similar media distribution applications, a server or distribution center sends a media/video to a group peers with different bandwidth resources and display capacities. In one of the approaches, the peer-to-peer approach, the server sends only one copy of the media over Internet, and each peer receives one segment of the media and exchanges his/her segment with other peers to receive the complete media. A key design issue in this approach is deciding the sizes of the segments delivered to individual peer which affect the time of complete media distribution. Equal sized segmentation does not always result in the least distribution time. In this paper, we study the problem of how to distribute non-scalable and scalable coded media from a server in closed peer-to-peer based IPTV networks. We propose a new distribution algorithm to find the media segment sizes optimized for the bandwidths of participating peers in order to minimize the time it takes to distribute the entire media to all end subscribers. First, we focus on finding the optimal solution in non-scalable media distribution. Then, we extend our method to scalable media distribution to find optimal segment sizes for all media layers. Simulations are conducted by

varying the number of peers and media sizes to investigate the impact of these parameters on both non-scalable and scalable video distribution. The experimental results have demonstrated the scalability and efficiency of the proposed distribution algorithm.

Keywords: IPTV, peer-to-peer, media, video, distribution

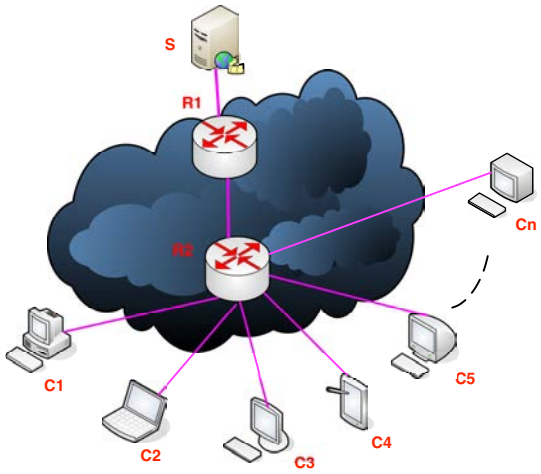


Fig. 1. Client-server based IPTV distribution

On Optimal Media/Video Distribution in Closed P2P-Based IPTV Networks

I. Introduction

Multimedia distribution over the Internet becomes popular because of technological advancements in both internets (e.g., higher network bandwidth) and video coding (e.g., H.264/SVC). Internet Protocol Television (IPTV) is a streaming media delivered over the Internet. In this paper, we study the problem of how to efficiently distribute media files, *i.e.,* video, to IPTV subscribers from a central server.

Fig. 1 shows a typical example of IPTV service, following the canonical client-server architecture. S is a video distribution server in a TV station, broadcasting video files over the Internet to n IPTV subscribed peers, C_1, C_2, and C_n in a local area. Such a system performs well with a small number of subscribers. However, with a large number of subscribers, the server capacity becomes a major limiting factor. For example, to support simultaneous view of a video stream at 800 kbps by 100 subscribers, the server S needs a sustaining bandwidth of 800 kbps $100 = 80$ Mbps.

From the network construction point of view, IP multicast may be the best solution to relieve this problem. However, the IP multicast has never dominated IPTV distribution over the

Internet [1]. The difficulty in group management, such as authorization, the security problem, and the network management support for network congestion and error control, etc., on IP multicast has limited its commercial deployment by ISPs and carriers. The peer-to-peer (P2P) network can be used to alleviate, though not completely solving, the scalability problem by moving the multicast task from IP-layer to application layer. Instead of sending 100 copies of the same video to all 100 subscribers, the server S can divide the video into 100 segments and send one segment to each subscriber. The subscribers or peers in the local area then exchange the segments they received via a local network, and, in the end, each receives a complete copy of video file. In this way, the subscribers not only receive video files from the server, they also help to disseminate the video file to other subscribers using their uplink bandwidth. This approach greatly reduces the pressure on video servers and the backbone network. In the above example, if only one copy of the video file is sent via the backbone network, the demand on the bandwidth can be reduced from 80 Mbps to 0.8 Mbps.

In this paper, the following distribution model is used. The server S breaks a $media\ file$ into $media\ objects$ of playback duration T [1]. If there are n subscribers requesting this media, every media object is divided or partitioned into $n\ segments$. In every playback interval T, the media distribution is broken into two phases: $server \quad distribution$ phase and $peer \quad swarming$ phase. In server-distribution phase, S sends only one copy (instead of n copies) of the object, and each subscriber receives a segment of (instead of the whole) the media object from S. In peer-swarming phase, the subscribers exchange these segments in a P2P fashion. Because subscribers in multimedia applications are usually with different Internet access bandwidths and computing powers, how to divide the media object into n segments will affect $the\ completion\ time$, the time for all subscribers to receive all media segments. The simplest solution is to divide the media object into n equal-sized segments. However, this approach doesn't always give the best performance. In this paper, we propose an algorithm to optimally partition a media object into n segments such that the completion time is minimized during playback interval T. The problem is formulated as a linear optimization program and the optimal segment sizes are calculated based on the upload and download bandwidths of all involved subscribers.

Two types of applications are considered in this paper: non-scalable and scalable media

[1]A media object has to be received by all subscribers with the playback time T.

distributions. In non-scalable media distribution, the media file is coded as one layer that is sent to all subscribers. In a scalable media distribution, the media file is coded as multiple layers. Subscribers can subscribe different numbers of layers. Media segments in a layer will be sent to only those entitled subscribers. In general, the media has better quality and requires more bandwidth if more layers are subscribed. The scalable video coding (SVC) extension in H.264/SVC [2] is an example of multiple layer media. In this extension, a video is coded into a scalable bit-stream with multiple layers. Different numbers of layers can be decoded to produce videos with different qualities. Subscribers can subscribe different numbers of layers based on their download bandwidths and budgets.

Related work in the field is described in Section II . In Section III, we describe our non-scalable P2P media distribution model and propose an optimal non-scalable media distribution algorithm. Building upon this foundation, we then formally create the multilayer media distribution model for the scalable media distribution. We present our study of the parallel distribution of scalable media and provide a solution to the optimal completion time problem in Section IV. We evaluate the performance of non-scalable and scalable distribution models, and the impact of multiple layers of requested video from different peers and media object size on the distribution system. We also investigate the affection of system distribution completion time caused by multi-layer number and different layer peers' departing/joining in Section V. Section VI concludes the paper.

II. Related Work

During the past decade, streaming IPTV media/video data over the Internet has been an area of intense study(see for example [3], [4], and [5]). A variety of video coding standards have been developed to support IPTV application, such as MPEG-2 [6] and H.264/SVC. The MPEG-2 encodes video stream in a pre-encoded non-scalable (single layer) bit-stream. The scalable video coding (SVC) extension in H.264 encodes a high-quality scalable video bit-stream that contains multiple layers. Therefore, IPTV media object is not only coded as non-scalable media stream, but also scalable media stream.

At the same time, with the increasing volume of IPTV distribution through the Internet, how to build and maintain efficient P2P overlay architecture for media stream has become a hot research topic. Based on the underlying overlay construction, two basic solutions to P2P systems have

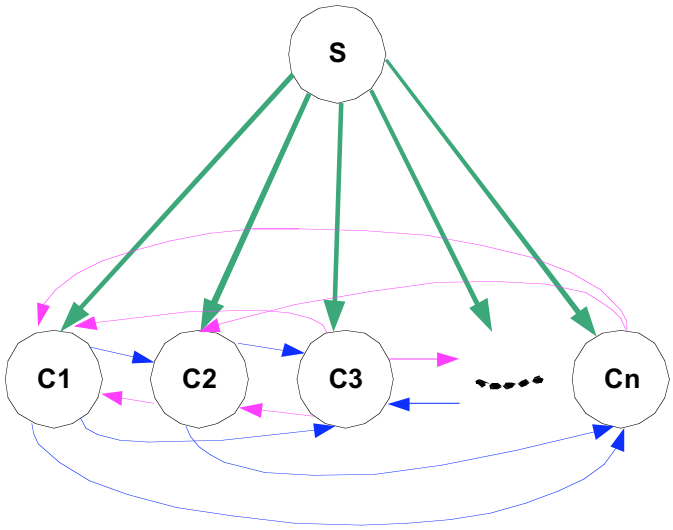


Fig. 2. Mesh-based P2P scheme

been devised for IPTV distribution [7]:

The mesh-based scheme [8] uses a pull-based architecture, as shown in Fig. 2. Peers swarm media streams over a randomly connected mesh networks. Each peer pulls the desired media stream from one of the other peers that has uploaded it. At the same time, this peer also supplies its available data to other peers. The P2P systems, such as Narada [9], CoolStreaming [10], PRIME [11], GridMedia [12], PPStream [13], PPLive [14], UUSee [15] and SopCast [16], use the mesh-based scheme.

As shown in Fig. 3, the tree-based scheme uses a push-based architecture – media stream is delivered over multiple tree-shaped overlays. The packets of the media stream are forwarded from the parent node to its child nodes. The P2P systems, such as NICE [17] and ZIGZAG [18], use the tree-based scheme.

When the mesh-based and tree-based are compared for live P2P streaming approaches [19], the mesh-based approach consistently exhibits a superior performance over the tree-based approach. The time delay in video transmission not only impacts the Quality-of-Service (QoS), but also affects the Quality-of-Experience (QoE). Therefore, it is very important to guarantee the completion of P2P system media distribution within the playback duration time. In [20], a measurement technique for monitoring the video playback quality in the mesh-based streaming

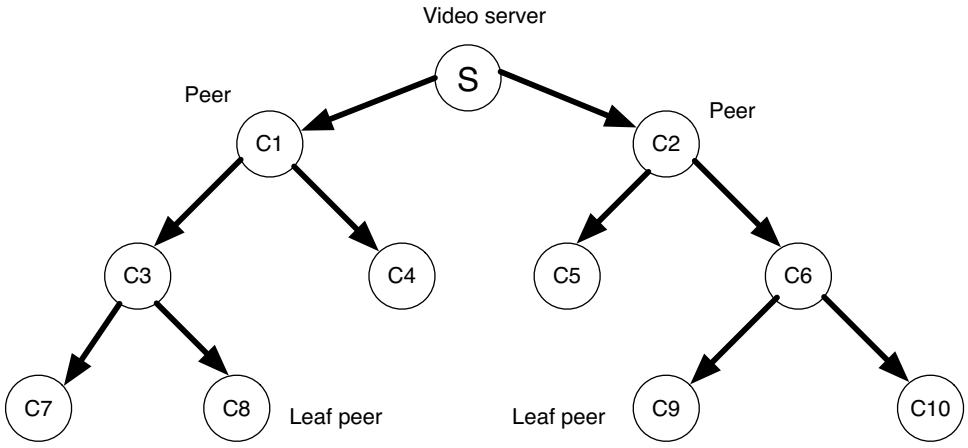


Fig. 3. Tree-based P2P scheme

system was proposed. Rather than the P2P architecture or quality measurement technique, our work focuses on how to optimally allocate media segments to peers to minimize the system media completion time of non-scalable and scalable media distribution in a mesh-based P2P network.

There are several P2P systems devised for IPTV applications, such as PPLive, PPStream, SopCast, and GridMedia [21] , which offer real-time services and have experienced commercial success. They do not disclosure their internal setup. In addition, BitTorrent [22] has been used widely to distribute large files and videos. BitTorrent treats video objects the same as normal files and does not exploit the properties of video bit-streams. BitTorrent employs a tit-for-tat strategy to fight against free riding, resulting in high throughput of the system [23]. However, it is a general file delivery service and divides media objects into equal-sized segments regardless of the peer bandwidths. In CoolStreaming, one of the most popular pull-based systems, the video stream is also divided into blocks with equal sizes to transmit. Unlike BitTorrent and CoolStreaming, we do not simply divide media objects into equal-sized segments for distribution. In our work, the media objects are allocated or divided non-uniformly to minimize system distribution time.

The fluid model [24] proposed by Kumar et. al provides an optimal bound of streaming performance in a fully-connected P2P network, for a video object that is uniformly coded. PROMISE [25] presents CollectCast, a P2P service for media streaming. Based on the working status of peers, it dynamically switches active senders and standby senders to maintain a satisfied

collective network performance for media streaming distribution. In this paper, we address peer heterogeneity by adopting non-scalable and scalable coded media/video representations and developing an algorithm to optimally distribute such layered media/video.

Compared to general Internet-based P2P streaming solutions, IPTV setups are usually characterized by closed-network design, where the server has full information of every subscriber, such as its IP address, uplink and downlink bandwidths on the dedicated networks. Every peer runs the same media distribution algorithm that comes from the setup box. In this paper, our research goal is to design a distribution algorithm to minimize distribution delay for media distribution in a closed mesh-based P2P network. This distribution algorithm is adapted to locate the media segment size to peers based on their bandwidth resources and subscription requirement. This paper is the extension of our previous work as noted in [26] and [27]. Inside this paper, we provide detail discussion on how to create the Optimal Parallel Layer Distribution (OPLD) algorithm for multi-layer media object distribution. Based on the OPLD model, we add in the study of multi-layer peers' departing/joining which impacts the distribution time in the system.

III. Non-scalable Media Distribution And Algorithm

In this section, we will discuss our non-scalable IPTV distribution model (Part III-A) and propose an optimal IPTV distribution algorithm (Part III-B). The bandwidth bottleneck will be analyzed in Part III-C.

A. A Non-scalable Peer-to-Peer Media Distribution Model

The P2P-based IPTV distribution model for a non-scalable media is similar to the one shown in Fig. 2. We consider the case where n subscribers request the same media from server S and receive that media partially from the server and partially from other subscribers in a closed mesh-based P2P network. In this system, for example, when a peer desires to watch a video, it first requests the video from server S. Then, the server S redirects the new peer to a tracker which maintains a list of peers. The tracker provides the new peer the information of other peers in the same system. The new peer will contact these peers to establish the relationship in order to receive video segments from them. Peers in the system periodically exchange their buffer maps with each other to indicate the available video segments. Each peer runs the same media peer-swarming algorithm that is embedded in the setup box. In such setups, incentives to

upload media segments are built in the media distribution algorithms run by the subscribers. In addition, the subscriber network is more static, compared to Internet-based P2P streaming.

As discussed in Section I, the server breaks a media file into media objects with playback duration T, that is, each media object has to be received by all subscribers in time interval T. In each playback duration interval T, a media object D is divided into n disjoint segments S_g_1 S_g_n, where n is the number of subscribers involved. So, $D = \bigcup_{i=1}^{n} S\_g_i$, $S\_g_i \bigcap S\_g_j =$, $i = j$. Let the size of segment S_g_i is s_i, then $D\_S = \sum_i^n s_i$. A distribution of D is characterized by $(s_1$ $s_n)$. The steps which are run by the media server in server-distribution phase and subscribers in peer-swarming phase are presented below:

Media server distribution algorithm:

1: Break media file into objects of playback duration T
2: **for** each playback interval T **do**
3: Divide the current media object D into n segments, S_g_1 S_g_2 S_g_n.
4: Send media segment S_g_i to subscriber C_i, $i = 1$ 2 n.
5: **end for**

Subscriber C_i peer-swarming algorithm:

1: Update media server with IP address, uplink, and downlink bandwidths
2: **for** each playback interval T **do**
3: Receive media segment S_g_i from the media server S
4: Send S_g_i to subscriber C_1 C_2 C_{i-1} C_{i+1} C_n
5: In the same time, receive media segment S_g_j from subscriber C_j, $i = j$.
6: **end for**

The key challenge in the media server distribution algorithm presented in this model is how to divide the current media object D into n segments, S_g_1 S_g_2 S_g_n, so that all the completion time is minimized. The simplest solution is to divide D into n equal sized segments. This may not be optimal. Clearly, the optimal distribution $(s_1$ $s_n)$ is a function of download /upload speeds of involved subscribers. To facilitate the discussion, more terms and notations are defined and together with previously defined notations, are summarized in Table I.

One example using this model for P2P media distribution is presented below. There are three peers in total in the system.

TABLE I

SUMMARY OF NOTATIONS FOR NON-SCALABLE SYSTEM

T	playback interval
D	the media object to be played after interval T
D_S	size of the media object D
n	number of subscribers
C_i	ID of subscriber i
r	total supplying bandwidth of media server S
r_i	supplying bandwidth of media server S allocated to C_i $\sum_{i=1}^{n} r_i = r$
S_g_i	media segment sent from S to C_i, $S\_g_i \bigcap S\_g_j =$, $i = j$, $\bigcup_{i=1}^{n} S\_g_i = D$
s_i	size of media segment S_g_i, $D\_S = \sum_i^n s_i$
d_i	download bandwidth of peer C_i
u_i	upload bandwidth of peer C_i
T_d_i	time for C_i to download S_g_i from S
T_u_i	time for C_i to send S_g_i to another peer
t_i	time for C_i to finish downloading and uploading, *i.e.,* to receive all segments and send S_g_i to all other peers.
t	completion time $t = \max\ t_i$
ed_i	$\min\ d_i\ r_i$

First, Server S works in distribution Phase (Fig. 4). Server S divides the current media object D into $three$ segment sizes $s_1\ s_2\ s_3$, and sends s_i to subscriber C_i, $i = 1\ 2\ 3$.

After receiving s_i from server S, peers start to work in Peer Swarming Phase 1 (Fig. 5). Peer C_i uploads its own segment size s_i and sends it to $C_{i+1(modn)}$, *i.e.,* C_1 sends s_1 to C_2, C_2 sends s_2 to C_3, and C_3 sends s_3 to C_1.

In the next Peer Swarming Phase 2 (Fig. 6), peer C_i sends its owned segment size s_i to peer $C_{i+2(modn)}$, *i.e.,* C_1 sends s_1 to C_3, C_2 sends s_2 to C_1, and C_3 sends s_3 to C_2.

Until all three peers in the system receive all three video segments to playback the video object D, the system completes media distribution.

We assume that downloading and uploading can be in parallel. We also assume that for each peer, the upload speed is slower than the download speed of all peers including itself, *i.e.*, $u_i \quad d_j \quad i\ j\ =\ 1\ 2 \qquad n$. Hence, we can estimate the time t_i for peer C_i to receive all segments and send s_i to all other $n \quad 1$ peers as follows.

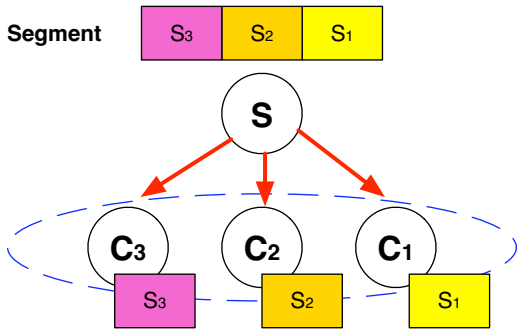


Fig. 4. Server distribution phase. Media object D is divided to three segments size s_1 s_2 s_3. S sends them to peer C_1 C_2 C_3

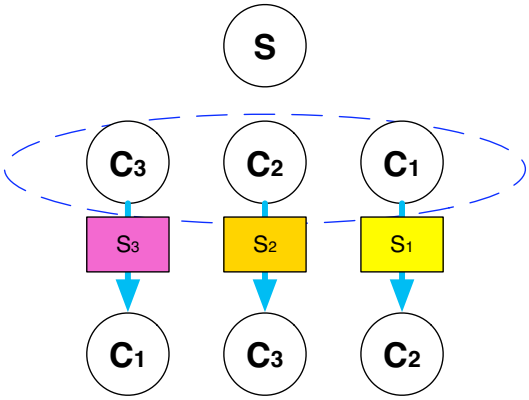


Fig. 5. The first step of Peer Swarming phase: C_i sends s_i to $C_{i+1(modn)}$

$$t_i = T\_d_i + (n \quad 1)T\_u_i = s_i \quad \min \; r_i \; d_i \quad + (n \quad 1)s_i \; u_i \tag{1}$$

and the completion time

$$t = \max \; t_i \quad i = 1 \qquad n \tag{2}$$

This formula is used later to find the optimal distribution of D.

B. *Optimal Media distribution Algorithm*

The following theorem states how to achieve optimal completion time in non-scalable media distribution. We prove the single system has the optimal completion time when the optimal condition is satisfied.

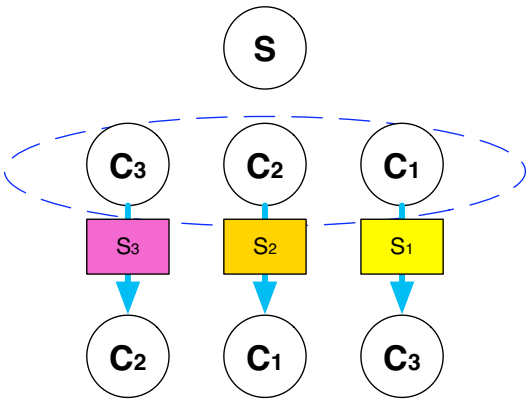


Fig. 6. The second step of Peer Swarming phase: C_i sends s_i to $C_{i+2(modn)}$

THEOREM 1 *The completion time is minimized, if the distribution* $(s_1 \quad s_n)$ *satisfies the following condition:*

$$\begin{aligned} t_1 = s_1 \ ed_1 + (n \quad 1)s_1 \ u_1 = s_2 \ ed_2 + (n \quad 1)s_2 \ u_2 = t_2 \\ = s_3 \ ed_3 + (n \quad 1)s_3 \ u_3 = t_3 \\ \\ = s_n \ ed_n + (n \quad 1)s_n \ u_n = t_n \end{aligned} \tag{3}$$

where $ed_i = \min \ d_i \ r_i$ *, and the completion time* $t = t_i \ i = 1 \ 2 \quad n$

As indicated in Theorem 1, the distribution $(s_1 \quad s_n)$ is optimal if all $t_i \ i = 1 \quad n$ are equal. Intuitively, by the Eqn.(3), every peer's download bandwidth is fully utilized during the peer swarming phase process.

Proof: Let t be the completion time when the above condition is satisfied, then

$$t = \max \ t_i \ = s_i \ ed_i + (n \quad 1)s_i \ u_i \ i = 1 \ 2 \quad n$$

Assume there exists a distribution $(s_1^{'} \quad s_n^{'})$ that does not satisfy Eqn.(3), and the corresponding completion time is $t^{'} < t$, where $s_i^{'}$ is the size of media segment $S^{'}\_g_i$ to be sent to C_i. Then $\sum_{i=1}^{n} s_i^{'} = \sum_{i=1}^{n} s_i = D\_S$. Without loss of generality, we assume, $t_1^{'} \quad t_2^{'} \quad t_n^{'} = t^{'}$,

where $t_i^{'} = s_i^{'}\ ed_i + (n \quad 1)s_i^{'}\ u_i$. Hence we have

$$t_i^{'} \quad t^{'} < t = t_i$$

$$or \quad s_i^{'}\ ed_i + (n \quad 1)s_i^{'}\ u_i < s_i\ ed_i + (n \quad 1)s_i\ u_i$$

$$(s_i^{'} \quad s_i)(1\ ed_i + (n \quad 1)\ u_i) < 0$$

$$Because \quad (1\ ed_i + (n \quad 1)\ u_i) > 0$$

$$s_i^{'} < s_i \quad i = 1\ 2 \qquad n$$

Then we have $\sum_{i=1}^{n} s_i^{'} < \sum_{i=1}^{n} s_i$. This contradicts the fact that regardless of how the media object is divided, the sum of the media segment sizes is equal to the size of the media object. Therefore, we prove that the distribution $(s_1 \qquad s_n)$ satisfying Eqn.(3) is optimal. ■

C. Peer-Side and Server-side Bandwidth Bottleneck

In this sub-section, we discuss how to find the optimal distribution $s_1 \qquad s_n$ for two cases, differentiated by where the bandwidth bottleneck resides in the network.

1) The *peer-side bottleneck* occurs when the supplying bandwidth from S is larger than the aggregate downloading bandwidth of all the subscribers, *i.e.,* $\sum_{i=1}^{n} d_i \quad r$. This happens most when end subscribers have slower links connected to IPTV.

2) The *server-side bottleneck* occurs when the supplying bandwidth from S is smaller than the aggregate downloading bandwidth of the peers, *i.e.,* $\sum_{i=1}^{n} d_i > r$. This case happens when n is large. *i.e.,* when the number of subscribers is larger than what IPTV can accommodate.

As the server's bandwidth is sufficient, *i.e.,* $r > \sum_{i=1}^{n} d_i$, we don't need to worry about how the server allocates its bandwidth among peers, *i.e.,* deciding $r_i\ i = 1\ 2 \qquad n\ \sum_{i=1}^{n} r_i = r$. It is sufficient to apply the optimal condition to decide $s_i\ i = 1\ 2 \qquad n$.

Combining the $n \quad 1$ equations in Eqn.(3) and the fact that the media segment sizes sum up to D_S, the optimal distribution $s_1\ s_2 \qquad s_n$ can be obtained, by solving the following system

of n linear equations,

$$\begin{pmatrix} \frac{1}{d_1}+\frac{n-1}{u_1} & -(\frac{1}{d_2}+\frac{n-1}{u_2}) & & & \\ & \frac{1}{d_2}+\frac{n-1}{u_2} & -(\frac{1}{d_3}+\frac{n-1}{u_3}) & & \\ & & \cdots & \cdots & \\ & & & \cdots\cdots & \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \\ \cdots \\ \cdots \\ s_n \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ D\_S \end{pmatrix}$$

When the bandwidth is limited at the server side, *i.e.,* $\sum_{i=1}^{n} d_i > r$, we need to find both media distribution $(s_1 \quad s_n)$ and server bandwidth allocation $(r_1 \quad r_n)$. According to Theorem 1, the following distribution and allocation that satisfy the optimal equations below are

$$s_1 \ u_1 = s_2 \ u_2 = \quad = s_n \ u_n$$

$$s_1 \ r_1 = s_2 \ r_2 = \quad = s_n \ r_n$$

Hence, the optimal distribution $(s_1 \quad s_n)$ and allocation $(r_1 \quad r_n)$ can be obtained as follows:

First, we calculate the optimal distribution $(s_1 \quad s_n)$ by solving the following linear system of equations.

$$\begin{pmatrix} \frac{1}{u_1}-\frac{1}{u_2} & & & & \\ & \frac{1}{u_2} & -\frac{1}{u_3} & & \\ & & \cdots & \cdots & \\ & & & \cdots\cdots & \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \\ \cdots \\ \cdots \\ s_n \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ D\_S \end{pmatrix}$$

Then, based on s_i, we allocate server bandwidth r_i accordingly so that all the peers finish their downloading and uploading at the same time. That is, optimal allocation $(r_1 \ r_2 \quad r_n)$ is calculated by solving the following equations.

$$\begin{pmatrix} \frac{1}{s_1}-\frac{1}{s_2} & & & & \\ & \frac{1}{s_2} & -\frac{1}{s_3} & & \\ & & \cdots & \cdots & \\ & & & \cdots\cdots & \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} r_1 \\ r_2 \\ \cdots \\ \cdots \\ r_n \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ r \end{pmatrix}$$

Here is an example: assume the media server's supplying bandwidth (r) is set to 10 Mbps, the media object is equal to 6 Mbits, and each peer's download and upload bandwidths are summarized in Table II. So, the bandwidth bottleneck is limited at the peer-side.

The optimal distribution solution is applied to this example. The sizes of media segments, the download time, the upload time, and the completion time are calculated and summarized in Table III. The optimal completion time is 22.92 seconds.

TABLE II

NON-SCALABLE PEERS' DOWNLOAD AND UPLOAD BANDWIDTHS.

C_i	d_i(kbps)	u_i(kbps)
1	1000	400
2	1000	200
3	800	300
4	800	200
5	600	160
6	600	130

TABLE III

UPLOAD AND DOWNLOAD TIMES USING THE OPTIMAL DISTRIBUTION SOLUTION.

C_i	s_i(Mbits)	d_i(kbps)	T_d_i(s)	u_i(kbps)	T_u_i(s)	$(n \quad 1)T\_u_i$(s)	t_i(s)
1	1.70	1000	1.70	400	4.20	21.23	22.92
2	0.88	1000	0.88	200	4.41	22.04	22.92
3	1.28	800	1.60	300	4.26	21.32	22.92
4	0.87	800	1.09	200	4.37	21.83	22.92
5	0.70	600	1.16	160	4.35	21.76	22.92
6	0.57	600	0.95	130	4.39	21.97	22.92

IV. SCALABLE MEDIA DISTRIBUTION

P2P network systems, such as PPStream, PPLive, and SopCast, have been widely used to improve the performance of scalable layered video, as defined in H.264/SVC. To stream an H.264/SVC video to different subscribers, the video only needs to be encoded once into a scalable bit-stream, consisting of multiple layers, which can be decoded by subscribers in different ways. A low-bandwidth subscriber can choose to decode only a subset of layers, rendering videos of primitive quality, whereas a high-bandwidth subscriber can decode all the layers to produce video of high quality.

Our proposed P2P IPTV scalable media object distribution model is shown in Fig. 7. The media objects are encoded into an m-layer representation. In this paper, we number the layers sequentially, so that layer i depends on all lower-numbered layers *, i.e.,* from 1 to $i \quad 1$. A layer i subscriber needs to receive all the media objects from layer 1 to layer i to be able to decode the video correctly. Also, a layer i subscriber should not receive media objects of layer $i+1 \qquad m$ which are not paid by this subscriber. For example, layer 1 subscribers are supposed to receive

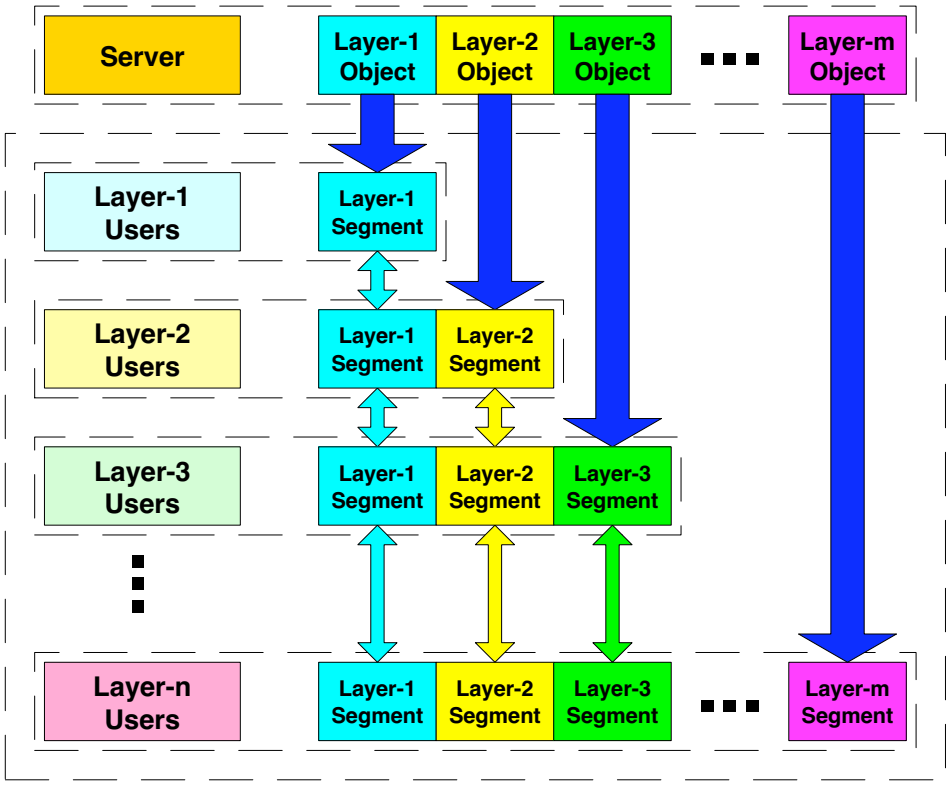


Fig. 7. Peer-to-peer video distribution model of scalable media objects

layer 1 media object only, layer 2 subscribers should receive layer 1 & 2 media objects and so on. In other words, a media object in layer k should be sent to all and only subscribers in layer k $k+1$ m. We follow the two phase model described in Section III: server distribution & peer swarming to distribute and exchange media segments. That is, server S divides the objects in all layers into segments and sends segments to all subscribers entitled. Then, peers exchange segments.

The problem that a server faces is how to optimally divide all media objects in all layers into media segments and send them to entitled subscribers in different layers. Let us define a general scalable media distribution model. Let $C_{i,j}$ be the jth subscriber in layer i. In layer i, there are n_i subscribers ($i = 1\ 2$ m). Let D_k be the media object in layer k, $k = 1\ 2$ m. D_k is divided into segments to be sent to subscribers in layers k $k+1$ m. Let $S\_g_{i,j,k}$ be the segment of D_k to be sent to subscriber $C_{i,j}$, $i = k$ m, $j = 1$ n_i. Also $\bigcup_{i=k}^{m} \bigcup_{j=1}^{n_i} S\_g_{i,j,k} = D_k$, $S\_g_{i,j,k} \bigcap S\_g_{l,t,p} =$, $(i\ j\ k) = (l\ t\ p)$. Let $s_{i,j,k}$ be the size of

$S\_g_{i,j,k}$, $t_{i,j,k}$ be the time needed for $C_{i,j}$ to finish downloading and uploading $S\_g_{i,j,k}$.

According to the definition, $t_{i,j,k}$ consists of two parts: the downloading time of $S\_g_{i,j,k}$ from the server: $s_{i,j,k}$ $d_{i,j}$, and the uploading time to send $S\_g_{i,j,k}$ to all the entitled peers, *i.e.*, all the peers subscribed to layer k and above. That is

$$t_{i,j,k} = \frac{s_{i,j,k}}{d_{i,j}} + (n_k + n_{k+1} + \quad + n_m \quad 1)\frac{s_{i,j,k}}{u_{i,j}} \tag{4}$$

Let $t_{i,j}$ be the time for $C_{i,j}$ to finish all required downloading of segments from the server and uploading of these segments sent to all other entitled subscribers. Then,

$$t_{i,j} = \sum_{k=1}^{i} t_{i,j,k} \tag{5}$$

Define

$$m\_t_i = \max\ t_{i,j} \tag{6}$$

Clearly, the completion time of scalable system is

$$m\_t = \max\ m\_t_i \tag{7}$$

A scalable distribution is denoted by ($s_{i,j,k}$, $i = 1 \quad m$). The distribution $S\_g_{i,j,k}$ affects the completion time in (7). The optimal distribution depends on the upload bandwidth $u_{i,j}$ and download bandwidth $d_{i,j}$ of $C_{i,j}$ $i = 1\ 2 \quad m\ j = 1\ 2 \quad n_i$. Notations are summarized in Table IV.

An example can be used to illustrate the terms and concepts presented so far. It shows how the distribution ($S\_g_{i,j,k}$, $i = 1 \quad m$, $j = 1 \quad n_i$, $k = 1 \quad m$) can affect the completion time. In this example, the media is encoded into three layers ($m = 3$) and each layer has two subscribed peers ($n_1 = n_2 = n_3 = 2$). So, the whole system has six peers in total. Let the size of D_i, the media object in layer i, $i = 1\ 2\ 3$, is 3000 kbits, *i.e.,* $SL_1 = SL_2 = SL_3 = 3000$ kbits. The six subscribers are, layer 1 subscribers, $C_{1,1}$ and $C_{1,2}$, layer 2 subscribers, $C_{2,1}$ and $C_{2,2}$, and layer 3 subscribers, $C_{3,1}$ and $C_{3,2}$. For continuous playback, all subscribers need to receive their entitled media objects within T seconds, either from the media server S or other peers. Each peer's download and upload bandwidths are summarized in Table V. In this example, we assume that the bandwidth is bottlenecked at the peer-side when downloading from the server. *i.e.,* $r_{i,j} \quad d_{i,j}$, where $r_{i,j}$ is supplying bandwidth of media server S allocated to $C_{i,j}$.

Next, we investigate different media distributions. For each distribution, we calculate the completion time.

TABLE IV

SUMMARY OF NOTATIONS FOR SCALABLE SYSTEM

m	total number of media layers
$C_{i\ j}$	the jth subscriber in layer i
D_i	media object in layer i
SL_i	size of media object D_i in layer i
n_i	number of subscribers in layer i
$d_{i\ j}$	download bandwidth of subscriber $C_{i\ j}$
$u_{i\ j}$	upload bandwidth of subscriber $C_{i\ j}$
$S\_g_{i\ j\ k}$	A segment of D_k sent from S to $C_{i\ j}$, $\bigcup_{i=k}^{m}\bigcup_{j=1}^{n_i} S\_g_{i\ j\ k} = D_k$, $S\_g_{i\ j\ k} \bigcap S\_g_{l\ t\ k} =$, $(i\ j) = (l\ t)$, i k
$s_{i\ j\ k}$	size of media segment $S\_g_{i\ j\ k}$
$s_{i\ j}$	total size of media segments that are received by subscriber $C_{i\ j}$ $s_{i\ j} = \sum_{k=1}^{i} s_{i\ j\ k}$, $(i$ $k)$
$t_{i\ j\ k}$	time for $C_{i\ j}$ to finish downloading and uploading $S\_g_{i\ j\ k}$.
$t_{i\ j}$	time for $C_{i\ j}$ to finish downloading and uploading media segments $S\_g_{i\ j\ k}$, $k = 1$ i of all layers. $t_{i\ j} = \sum_{k=1}^{i} t_{i\ j\ k}$
m_t_i	max time of subscribers in layer i to finish downloading and uploading. $m\_t_i = \max\ t_{i\ j}$, $j = 1$ n_i
m_t	time for whole system to finish downloading and uploading. $m\_t = \max\ m\_t_i$, $i = 1$ m

TABLE V

PEERS' DOWNLOAD AND UPLOAD BANDWIDTHS IN THE EXAMPLE.

User	Layer	$d_{i\ j}$(kbps)	$u_{i\ j}$(kbps)
$C_{1\ 1}$	1	100	100
$C_{1\ 2}$	1	100	100
$C_{2\ 1}$	2	200	200
$C_{2\ 2}$	2	200	200
$C_{3\ 1}$	3	300	300
$C_{3\ 2}$	3	300	300

A. Layer-by-Layer Distribution (LBL)

The first possible distribution is for media server S to optimally distribute media objects to its subscribers layer by layer using the non-scalable distribution algorithm presented in Section III. The total system completion time is the sum of the completion times for each layer respectively.

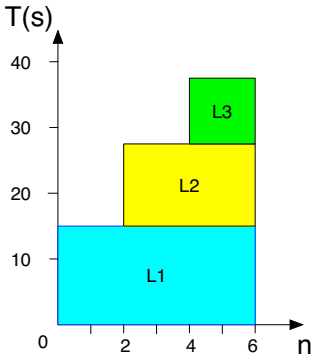


Fig. 8. Layer-by-layer distribution. Layer 1 distribution takes 15 seconds; layer 2 takes 12 seconds; layer 3 takes 10 seconds. The completion time is 37 seconds.

As shown in Fig. 8, layer 1 media object D_1 is optimally distributed among all the six subscribers, followed by layer 2 media object D_2 distribution to the four subscribers ($C_{2,1}$, $C_{2,2}$, $C_{3,1}$, and $C_{3,2}$), and followed by layer 3 media object D_3 distribution to the two subscribers ($C_{3,1}$ and $C_{3,2}$). The optimal non-scalable distribution algorithm presented in Section III is applied to each layer separately and independently, and the resulted media distribution is summarized in Table VI. For example, Layer 1 is treated as a non-scalable media object with six subscribers. By equation (3), D_1 is divided into six segments whose sizes are shown in the second column in table VI.

TABLE VI

MEDIA DISTRIBUTION BY LBL ALGORITHM.

$C_{i\ j}$	$s_{i\ j\ 1}$(kbits)	$t_{i\ j\ 1}(s)$	$s_{i\ j\ 2}$(kbits)	$t_{i\ j\ 2}$(s)	$s_{i\ j\ 3}$(kbits)	$t_{i\ j\ 3}$(s)
$C_{1\ 1}$	250	15	-	-	-	-
$C_{1\ 2}$	250	15	-	-	-	-
$C_{2\ 1}$	500	15	600	12	-	-
$C_{2\ 2}$	500	15	600	12	-	-
$C_{3\ 1}$	750	15	900	12	1500	10
$C_{3\ 2}$	750	15	900	12	1500	10

The individual completion times are calculated according to (4) to (7). For example, $t_{2,1,2}$ consists of two parts: the time for $C_{2,1}$ to download $S\_g_{2,1,2}$ whose size equals to 600/200 and the time for $C_{2,1}$ to upload or send $S\_g_{2,1,2}$ to $C_{2,2}$, $C_{3,1}$, $C_{3,2}$ that equals to (2+2-1)600/200.

That is

$$t_{2,1,2} = 600\ \ 200 + (2 + 2 \quad 1)600\ \ 200 = 12s$$

Similarly, all other $t_{i,j,k}$, $i\ \ k = 1\ \ 2\ \ 3$, $j = 1\ \ 2$ are calculated and summarized in Table VI. According to equation (4),

$$t_{1,1} = t_{1,1,1} = 15s$$

$$t_{1,2} = t_{1,2,1} = 15s$$

$$t_{2,1} = t_{2,1,1} + t_{2,1,2} = 15 + 12 = 27s$$

$$t_{2,2} = t_{2,2,1} + t_{2,2,2} = 15 + 12 = 27s$$

$$t_{3,1} = t_{3,1,1} + t_{3,1,2} + t_{3,1,3} = 15 + 12 + 10 = 37s$$

$$t_{3,2} = t_{3,2,1} + t_{3,2,2} + t_{3,2,3} = 15 + 12 + 10 = 37s$$

Based on equation (6) and (7), the system completion time for LBL, $m\_t(LBL)$,

$$m\_t(LBL) = \max\ \ m\_t_1\ \ m\_t_2\ \ m\_t_3 \ \ = \ \ 15\ \ 27\ \ 37 \ \ = 37s$$

B. Parallel Layer Distribution (PLD)

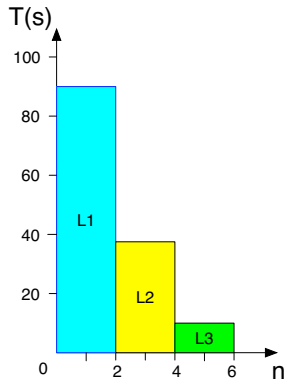


Fig. 9. Parallel layer distribution

In the second distribution, the media server S distributes media objects in all layers to a set of subscribers in parallel, and each peer uploads the received media segments to those entitled

TABLE VII

MEDIA DISTRIBUTION BY SIMPLE PLD ALGORITHM.

$C_{i\ j}$	$s_{i\ j\ 1}$(kbits)	$s_{i\ j\ 2}$(kbits)	$s_{i\ j\ 3}$(kbits)	$t_{i\ j}$(s)
$C_{1\ 1}$	1500	-	-	90
$C_{1\ 2}$	1500	-	-	90
$C_{2\ 1}$	0	1500	-	37.5
$C_{2\ 2}$	0	1500	-	37.5
$C_{3\ 1}$	0	0	1500	10
$C_{3\ 2}$	0	0	1500	10

peers. The basic idea is as follows, S divides D_1 into two segments and sends one segment to $C_{1,1}$ and the other segment to $C_{1,2}$. These two peers upload and send their received segments to all the peers, $C_{1,1}$, $C_{1,2}$, $C_{2,1}$, $C_{2,2}$, $C_{3,1}$, and $C_{3,2}$. Similarly, D_2 is divided into two segments, one of which is sent to $C_{2,1}$ and the other is sent to $C_{2,2}$. Then, $C_{2,1}$ and $C_{2,2}$ upload and send their segments to entitled peers $C_{2,1}$, $C_{2,2}$, $C_{3,1}$, and $C_{3,2}$, respectively. D_3 is divided into two segments, one of which is sent to $C_{3,1}$ and the other is sent to $C_{3,2}$. $C_{3,1}$ and $C_{3,2}$ will exchange their received segments among themselves. Unlike LBL, distributions of the three layers are performed in parallel, as illustrated in Fig. 9. The media distribution results are shown in Table VII. According to Eqn. (4) - (7),

$$t_{1,1} = t_{1,1,1} = 90s$$

$$t_{1,2} = t_{1,2,1} = 90s$$

$$t_{2,1} = t_{2,1,1} + t_{2,1,2} = 0 + 37\ 5 = 37\ 5s$$

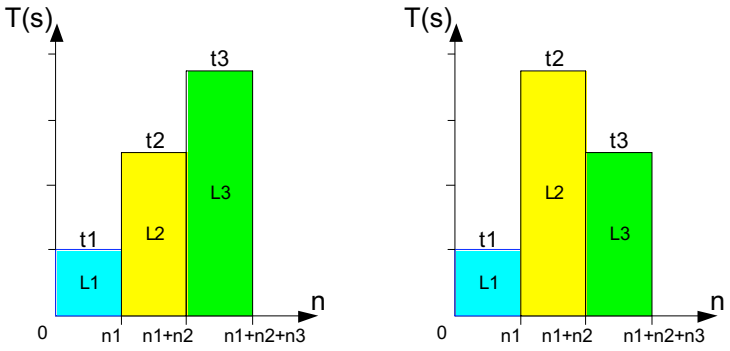
$$t_{2,2} = t_{2,2,1} + t_{2,2,2} = 0 + 37\ 5 = 37\ 5s$$

$$t_{3,1} = t_{3,1,1} + t_{3,1,2} + t_{3,1,3} = 0 + 0 + 10 = 10s$$

$$t_{3,2} = t_{3,2,1} + t_{3,2,2} + t_{3,2,3} = 0 + 0 + 10 = 10s$$

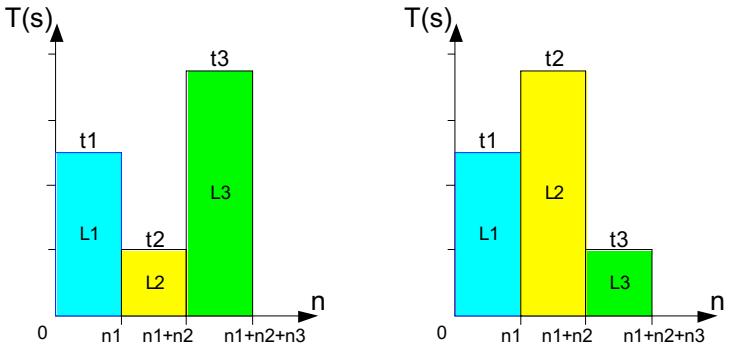
Therefore, $m_t_1 = 90$s, $m_t_2 = 37\ 5$s, and $m_t_3 = 10$s. The completion time $m\_t(PLD)$ is the maximum of the three:

$$m\_t(PLD) = max\ \ m\_t_1\ \ m\_t_2\ \ m\_t_3\ \ = max\ \ 90\ \ 37\ 5\ \ 10\ \ = 90s$$



(a) (b)

Fig. 10. (a) $t1 < t2 < t3$ (b) $t1 < t3 < t2$



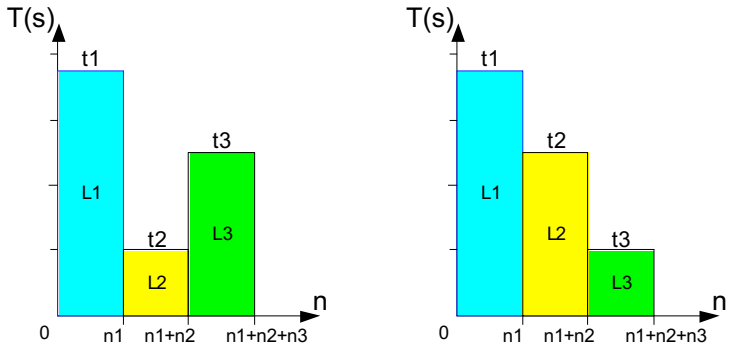
(a) (b)

Fig. 11. (a) $t2 < t1 < t3$ (b) $t3 < t1 < t2$

This clearly results in much longer completion time than LBL distribution. Notice that this distribution is not the best parallel layer distribution yet. Observe that four peers, $C_{2,1}$, $C_{2,2}$, $C_{3,1}$, and $C_{3,2}$, act as pure receivers of layer 1 media object and do not contribute any bandwidth in distributing layer 1 media segments. Similarly, peers, $C_{3,1}$ and $C_{3,2}$ do not contribute in distribution of layer 2 media segments either.

To gain more insights on how to improve completion time in PLD approach, let us enumerate six possible scenarios in this 3-layer example and analyze bottleneck in each case. The complete distributing time for layer 1, 2 and 3 subscribers are $t1$, $t2$ and $t3$.

Case a) $t1 < t2 < t3$ as shown in Fig. 10(a). The bandwidth bottleneck lies in distributing



(a) (b)

Fig. 12. (a) $t2 < t3 < t1$ (b) $t3 < t2 < t1$

layer 3 media object D_3, because no subscribers in layer 1 and 2 can help with layer 3 distribution, and $t3$ cannot be reduced any further. Thus, the whole system completion time is equal to $t3$ and cannot be better.

Case b) $t1 < t3 < t2$ as in Fig. 10(b). Because layer 3 peers can help distributing layer 2 media segments once layer 3 media object distribution is completed, the completion time can be improved and should be in the interval of $[t2\ t3]$.

Case c) $t2 < t1 < t3$ as in Fig. 11(a). The system bottleneck is in distributing layer 3 media segment, so the system completion time is still $t3$ and cannot be reduced.

Case d) $t3 < t1 < t2$ as shown in Fig. 11(b). Layer 3 peers can help distributing layer 1 and 2 media segments. Therefore, the whole system completion time is equal to $t3$ plus the time for layer 3 peers to distribute layer 1 and 2 media objects.

Case e) $t2 < t3 < t1$ as shown in Fig. 12(a). Layer 1 media object distribution takes the longest to complete. Fortunately, both layer 2 and layer 3 peers can assist in delivering layer 1 media object, so the system completion time should be smaller than $t1$.

Case f) $t3 < t2 < t1$ as shown in Fig. 12(b). Layer 2 and 3 peers are able to contribute more bandwidth in the distribution of layer 1 media object. Besides, layer 3 peers can also help layer 2 peers in delivering layer 2 media object. Therefore, the system completion time will be smaller than $t1$.

From the above six cases, we find that the completion time achieved by PLD can be improved

by making every peer a contributor for the layers that the peer is involved in.

C. Optimal Parallel Layer Distribution (OPLD)

An interesting question is how to systematically find the optimal parallel layer distribution in a large system with many subscribers. Here, the objective is to find $s_{i,j,k}$, $i = 1\ 2 \quad m\ j = 1\ 2 \quad n_i\ k \quad i$, so that the system completion time is minimized. This problem can be formulated as a linear program as follows:

$$
\begin{aligned}
&\text{min} \quad \max\ t_{i,j}\ i = 1\ 2 \quad m \quad j = 1\ 2 \quad n_i && (8)\\
&\text{subject to } SL_k = \sum_{i=k}^{m} \sum_{j=1}^{n_i} s_{i,j,k} \quad \text{for all } k = 1\ 2 \quad m\\
&\qquad s_{i,j,k} \quad 0\\
&\text{where} \quad t_{i,j} = \sum_{k=1}^{i} t_{i,j,k} \quad \text{for all } 1 \quad i \quad m \quad 1 \quad j \quad ni\\
&\qquad t_{i,j,1} = \frac{s_{i,j,1}}{d_{i,j}} + (\sum_{p=1}^{m} n_p \quad 1)\frac{s_{i,j,1}}{u_{i,j}} \quad \text{for all } i \quad 1\\
&\qquad t_{i,j,2} = \frac{s_{i,j,2}}{d_{i,j}} + (\sum_{p=2}^{m} n_p \quad 1)\frac{s_{i,j,2}}{u_{i,j}} \quad \text{for all } i \quad 2\\
&\qquad t_{i,j,k} = \frac{s_{i,j,k}}{d_{i,j}} + (\sum_{p=k}^{m} n_p \quad 1)\frac{s_{i,j,k}}{u_{i,j}} \quad \text{for all } i \quad k\\
&\qquad t_{i,j,i} = \frac{s_{i,j,i}}{d_{i,j}} + (n_i \quad 1)\frac{s_{i,j,i}}{u_{i,j}} \quad \text{for all } i \quad m
\end{aligned}
$$

In the constraints, $t_{i,j,k}$ and $t_{i,j}$ are defined in Eqn. (4) and (5). The constraint on media object size, SL_i, reflects the fact that D_i is distributed to all the subscribers in layers i to m.

All constraints in Eqn. (8) are affine. By adding a slack variable m_t, (8) can be converted to a standard linear program in (9) and can be solved for optimal solutions by a free GNU program, lp_solve [28].

$$\begin{aligned}
\text{min} \quad & m_t && (9)\\
\text{subject to} \quad & SL_k = \sum_{i=k}^{m} \sum_{j=1}^{n_i} s_{i,j,k} \quad \text{for all } k = 1\ 2 \quad m\\
& s_{i,j,k} \quad 0\\
& m_t \quad t_{i,j}\\
\text{where} \quad & t_{i,j} = \sum_{k=1}^{i} t_{i,j,k} \quad \text{for all } 1 \quad i \quad m \quad 1 \quad j \quad ni\\
& t_{i,j,1} = \frac{s_{i,j,1}}{d_{i,j}} + (\sum_{p=1}^{m} n_p \quad 1)\frac{s_{i,j,1}}{u_{i,j}} \quad \text{for all } i \quad 1\\
& t_{i,j,2} = \frac{s_{i,j,2}}{d_{i,j}} + (\sum_{p=2}^{m} n_p \quad 1)\frac{s_{i,j,2}}{u_{i,j}} \quad \text{for all } i \quad 2\\
& t_{i,j,k} = \frac{s_{i,j,k}}{d_{i,j}} + (\sum_{p=k}^{m} n_p \quad 1)\frac{s_{i,j,k}}{u_{i,j}} \quad \text{for all } i \quad k\\
& t_{i,j,i} = \frac{s_{i,j,i}}{d_{i,j}} + (n_i \quad 1)\frac{s_{i,j,i}}{u_{i,j}} \quad \text{for all } i \quad m
\end{aligned}$$

This is illustrated by the following example. According to the given parameters in Table V, the linear program to find the optimal distribution $(s_{i,j,k})$ is

$$\begin{aligned}
\text{min} \quad & m_t = \max\ t_{i,j}\ \ i = 1\ 2\ 3 \quad j = 1\ 2 && (10)\\
\text{subject to} \quad & SL_1 = s_{1,1,1} + s_{1,2,1} + s_{2,1,1} + s_{2,2,1} + s_{3,1,1} + s_{3,2,1} = 3000\\
& SL_2 = s_{2,1,2} + s_{2,2,2} + s_{3,1,2} + s_{3,2,2} = 3000\\
& SL_3 = s_{3,1,3} + s_{3,2,3} = 3000\\
& s_{i,j,k} \quad 0 \quad i = 1\ 2\ 3 \quad j = 1\ 2 \quad k = 1\ 2\ 3
\end{aligned}$$

where

$$t_{1,1,1} = \frac{s_{1,1,1}}{100} + (6 \quad 1)\frac{s_{1,1,1}}{100}$$
$$t_{1,2,1} = \frac{s_{1,2,1}}{100} + (6 \quad 1)\frac{s_{1,2,1}}{100}$$
$$t_{2,1,1} = \frac{s_{2,1,1}}{200} + (6 \quad 1)\frac{s_{2,1,1}}{200}$$
$$t_{2,2,1} = \frac{s_{2,2,1}}{200} + (6 \quad 1)\frac{s_{2,2,1}}{200}$$
$$t_{3,1,1} = \frac{s_{3,1,1}}{300} + (6 \quad 1)\frac{s_{3,1,1}}{300}$$
$$t_{3,2,1} = \frac{s_{3,2,1}}{300} + (6 \quad 1)\frac{s_{3,2,1}}{300}$$
$$t_{2,1,2} = \frac{s_{2,1,2}}{200} + (4 \quad 1)\frac{s_{2,1,2}}{200}$$
$$t_{2,2,2} = \frac{s_{2,2,2}}{200} + (4 \quad 1)\frac{s_{2,2,2}}{200}$$
$$t_{3,1,2} = \frac{s_{3,1,2}}{300} + (4 \quad 1)\frac{s_{3,1,2}}{300}$$
$$t_{3,2,2} = \frac{s_{3,2,2}}{300} + (4 \quad 1)\frac{s_{3,2,2}}{300}$$
$$t_{3,1,3} = \frac{s_{3,1,3}}{300} + (2 \quad 1)\frac{s_{3,1,3}}{300}$$
$$t_{3,2,3} = \frac{s_{3,2,3}}{300} + (2 \quad 1)\frac{s_{3,2,3}}{300}$$

and

$$m\_t \quad t_{1,1} = t_{1,1,1}$$
$$m\_t \quad t_{1,2} = t_{1,2,1}$$
$$m\_t \quad t_{2,1} = t_{2,1,1} + t_{2,1,2}$$
$$m\_t \quad t_{2,2} = t_{2,2,1} + t_{2,2,2}$$
$$m\_t \quad t_{3,1} = t_{3,1,1} + t_{3,1,2} + t_{3,1,3}$$
$$m\_t \quad t_{3,2} = t_{3,2,1} + t_{3,2,2} + t_{3,2,3}$$

Tool lp_solve is used to find the optimal distribution $(s_{i,j,k})$ and completion time $t_{i,j}$ of peer $C_{i,j}$. The detailed results are presented in Table VIII. It takes 30 seconds for the system to complete the distribution. It is clearly much shorter than 90 seconds, the time obtained by the

TABLE VIII

MEDIA DISTRIBUTION BY OPTIMAL PLD ALGORITHM.

$C_{i\ j}$	$s_{i\ j\ 1}$(kbits)	$s_{i\ j\ 2}$(kbits)	$s_{i\ j\ 3}$(kbits)	$t_{i\ j}$(s)
$C_{1\ 1}$	500	-	-	30
$C_{1\ 2}$	500	-	-	30
$C_{2\ 1}$	500	750	-	30
$C_{2\ 2}$	500	750	-	30
$C_{3\ 1}$	500	750	1500	30
$C_{3\ 2}$	500	750	1500	30

simple PLD approach in Section IV-B, as it has made every peer a contributor in every possible layer. This completion time is also shorter than 37 seconds obtained by the LBL approach in Section IV-A, due to the media distribution being more "balanced" and the system bandwidth being fully utilized which then allow all the peers to complete their distribution at the same time.

From the optimization formulation in (8), we know that, for the peers in higher layer i, $i > 1$, their completion time shown below includes two parts.

$$t_{i,j} = t_{i,j,i} + \sum_{k=1}^{i-1} t_{i,j,k}$$

The first part $t_{i,j,i}$ is the time to distribute layer i media segment $S\_g_{i,j,i}$. Peers in lower layers from layer 1 to layer $(i \quad 1)$ can not support layer i to distribute media segment $S\_g_{i,j,i}$. The second part $\sum_{k=1}^{i-1} t_{i,j,k}$ is the time for peers in layer i to help delivering lower layer media segments. If lower layer peers have more contributions in distributing lower layers segments, the second part $\sum_{k=1}^{i-1} t_{i,j,k}$ will be reduced. This will help reducing the whole system completion time.

From the discussion of the six cases, we know that the peers' completion time of higher layer dominates the whole system completion time. Therefore, when peers in the lower layers leave the system, peers in higher layers need to contribute more into the lower layer media segment distribution. This will increase the whole system completion time. In contrast, when new lower layers' peers join into the system, they will help reduce the whole system completion time. The information on how to handle peers departing and joining by a more balanced distribution will

be addressed in our future paper.

V. Performance Evaluations

In this section, we first evaluate the performance of the proposed non-scalable media optimal distribution algorithm (OPT) by comparing it with two other heuristic algorithms.

RAND Algorithm – Server S divides the non-scalable media object into n segments with random size, and sends them to n end users in the same cluster.

AVE Algorithm – Server S divides the non-scalable media object into n segments of equal size and distributes them to the end users.

Later, we evaluate the performance of the P2P scalable media distribution model. We compare the performance of the two media distribution algorithms that have been studied: layer-by-layer distribution (LBL) discussed in Section IV-A and optimal parallel layer distribution (OPLD) discussed in Section IV-C.

A. Simulation Setup

During the simulation setup, we assume that the bandwidth is bottlenecked at the peer-side. We also assume that for each peer's upload speed is slower than the download speed of all peers including itself. We simulate each test case for 100 times and report the average of the completion times of the test algorithm.

In our simulations for non-scalable media distribution, the server has media objects of size [2, 20] Mbits to distribute to its subscribers. To simulate network settings, we randomly the download bandwidth in [3000, 10000] kbps, and upload bandwidth in [768,3000] kbps. The number of end users ranges from 10 to 100 in our system.

For the simulation setup for scalable media distribution, the server has scalable media objects that are coded in three layers to distribute to subscribers in the system. Each layer has at least 10 subscribers. To simulate network settings, we randomly generate each peer's download bandwidth and upload bandwidth from a specified range of a given layer, as shown in Table IX.

B. Simulation Results

1) $OPT\ vs\ \ RAND\ and\ AVE\ with\ different\ peers\ number$

TABLE IX

PEER BANDWIDTH RANGES IN EXPERIMENTS.

$Layer$	$d_{i,j}$(kbps)	$u_{i,j}$(kbps)
1	[1000, 3000]	[300, 1000]
2	[1000, 6000]	[300, 2000]
3	[1000, 9000]	[300, 3000]

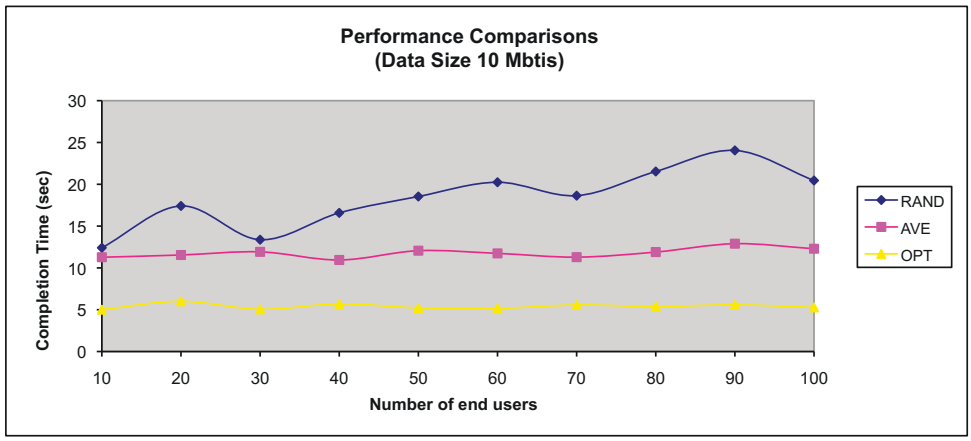


Fig. 13. Comparison of non-scalable media completion time for different number of peers.

In the first experiment for non-scalable media distribution, we have studied how the completion time changes with the number of end peers in the system. Fig. 13 plots the time comparisons of distributing a 10 Mbits non-scalable media object using the three algorithms – RAND, AVE and OPT. The proposed OPT Algorithm completes media distribution using the shortest time. When delivering the 10 Mbits media object to all peers, using OPT Algorithm takes less than half of the time the AVE Algorithm, and takes between 1/8 to 1/4 of the time of the RAND Algorithm. We can also observe that the media distribution time of the OPT algorithm almost stays constant when the number of peers increases. This shows that our media distribution algorithm is scalable.

2) *OPT vs. RAND and AVE with different media object size*

In the second experiment for non-scalable media distribution, we have investigated how media completion time changes with media object size. Fig. 14 shows the comparison of completion times of the three algorithms, when the size of the media object ranges from 2 to 20 Mbits. As the file size increases, the OPT Algorithm has better performance than

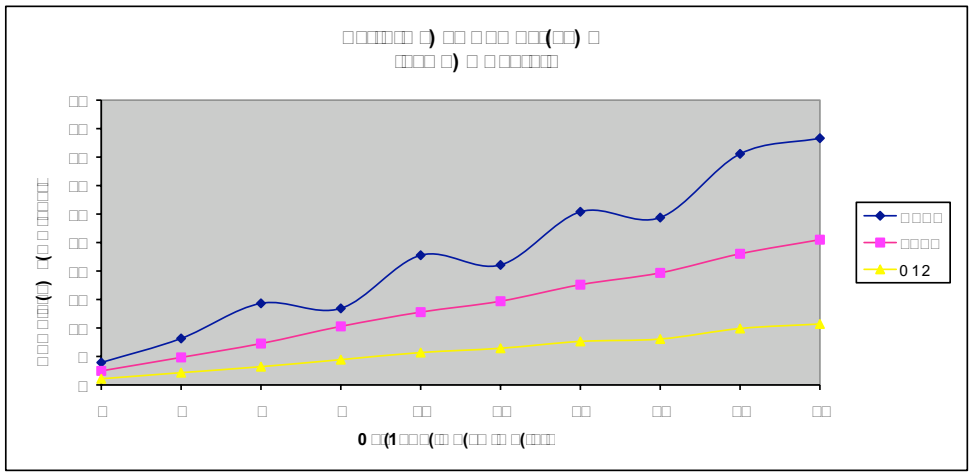


Fig. 14. Comparison of non-scalable media completion time for different media object size.

both AVE and RAND Algorithms. The ratio of the distribution time of OPT to AVE stays at around 0.3 as the file size increases.

3) *OPLD vs LBL with different peers number*

In the first experiment for scalable media distribution, we compare both OPLD and LBL algorithms by varying the number of subscribers. We fix the size of media objects in each layer to be 10 Mbits, and vary the number of peers in each layer from 10 to 100. Fig. 15 shows that no matter how the number of peers changes, the completion time in delivering 10 Mbits media object in each layer to their subscribers by using OPLD algorithm takes about 20% less than that of the LBL Algorithm. The completion times of the OPLD algorithm almost stay constant when the number of peers increases.

4) *OPLD vs LBL with different media object size*

In the second experiment for scalable media distribution, we have investigated how media object size affects the completion time on OPLD and LBL by fixing the number of peers in each layer to be 20. The size of the media object in layer 1 ranges from 5 to 25 Mbits, in layer 2 ranges from 10 to 30 Mbits, and in layer 3 ranges from 15 to 35 Mbits respectively. Fig. 16 shows the comparison of completion times of OPLD and LBL. While OPLD consistently shows better performance than LBL, the performance gap between the two increases while the media object size increases.

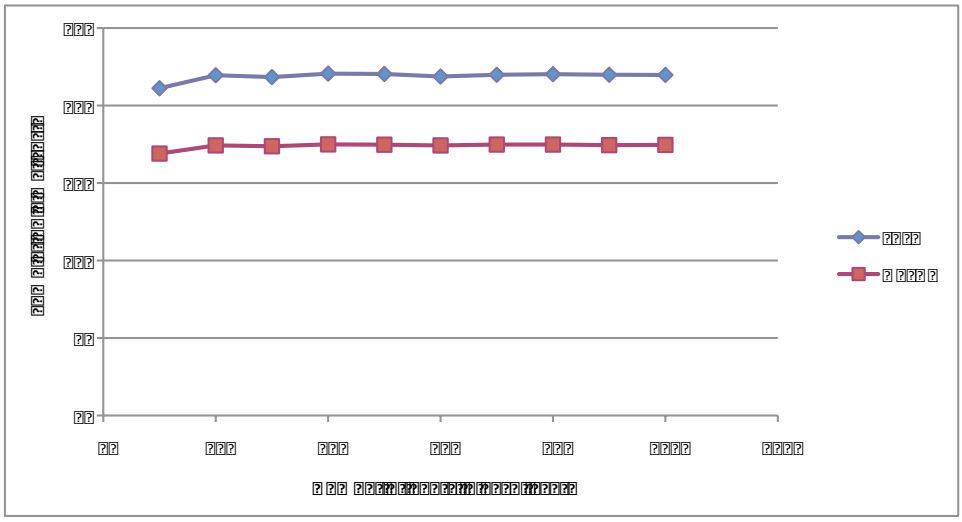


Fig. 15. Completion time as the function of scalable media peers number. ($SL_1 = SL_2 = SL_3$)

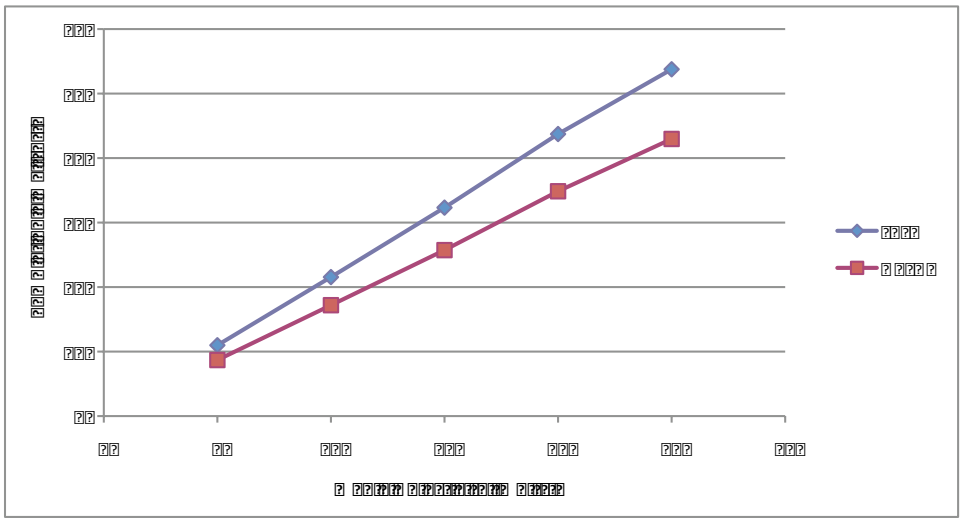


Fig. 16. Completion time as the function of scalable media object size in each layer. ($n_1 = n_2 = n_3 = 20$)

5) *OPLD vs LBL with different layers number*

In the third experiment for scalable media distribution, we have explored how the layers number affects the completion time on OPLD and LBL by fixing the number of peers in each layer to be 20. The layers number ranges from 2 to 10. The sizes of the media object in layers are $SL_1 = 10Mbits$, $SL_2 = 9Mbits$, , and $SL_{10} = 1Mbits$. Each layer peers bandwidth was randomly generated based on the range showed on Table X. From the

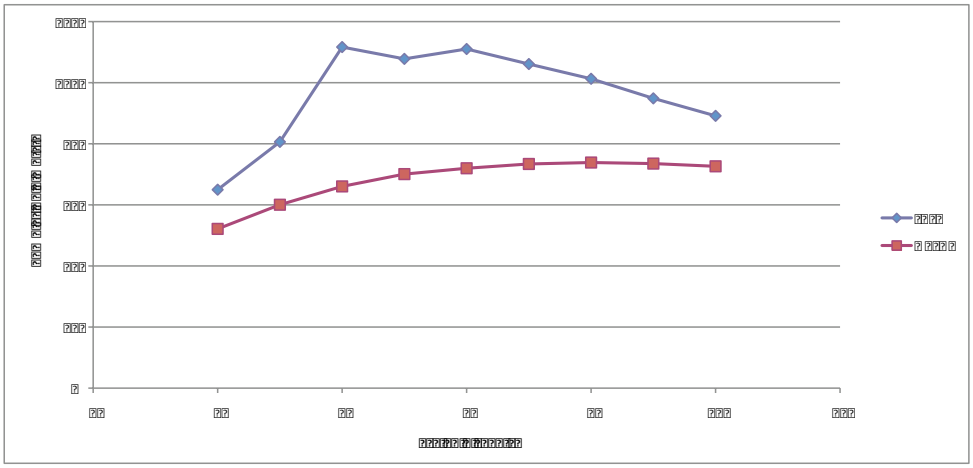


Fig. 17. Completion time as the function of scalable media layers number. ($n_1 = n_2 =$ $= n_{10} = 20$)

TABLE X

PEER BANDWIDTH AND MEDIA SIZE IN LAYERS NUMBER EXPERIMENT.

$Layer\ i$	SL_i(Mbits)	$Peers\ number$	$d_{i\ j}$(kbps)	$u_{i\ j}$(kbps)
1	10	20	[100, 1000]	[100, 384]
2	9	20	[100, 2000]	[100, 512]
3	8	20	[100, 3000]	[100, 640]
4	7	20	[100, 4000]	[100, 768]
5	6	20	[100, 5000]	[100, 896]
6	5	20	[100, 6000]	[100, 1024]
7	4	20	[100, 7000]	[100, 1152]
8	3	20	[100, 8000]	[100, 1280]
9	2	20	[100, 9000]	[100, 1408]
10	1	20	[100, 10000]	[100, 1536]

comparison of completion times of OPLD and LBL as shown in Fig. 17, the LBL showed its fluctuation in performance by varying the scalable layered size and peers bandwidth, while OPLD coherently showed better performance than LBL.

During media object delivering, peers in different layer departing/joining will affect the media object distribution completed time. In order to investigate how they affect the whole system, we did the following three layers' peers departing and joining simulation. To simulate network settings, we randomly generated each peer's download bandwidth and upload bandwidth from a specified range of a given layer, as shown in Table XI.

TABLE XI

PEER BANDWIDTH RANGES IN DEPARTING/JOINING EXPERIMENTS.

$Layer$	$d_{i\ j}$(kbps)	$u_{i\ j}$(kbps)
1	[100, 1500]	[100, 384]
2	[100, 3000]	[100, 512]
3	[100, 6000]	[100, 768]

TABLE XII

MEDIA DISTRIBUTION TIME T(IN SEC) BY EACH LAYER WITH SAME DEPARTING RATE.

Layer 1	Layer 2	Layer 3	0% departing		10% departing		20% departing		30% departing		40% departing		50% departing	
n_1	n_2	n_3	$t(s)$	%	$t(s)$	%	$t(s)$	%	$t(s)$	%	$t(s)$	%	$t(s)$	%
40	70	100	63.99	100.00%	6.41	10.02%	12.79	19.99%	19.06	29.79%	25.21	39.40%	31.78	49.67%
40	100	70	63.36	100.00%	6.34	10.00%	12.71	20.05%	18.87	29.79%	24.99	39.45%	31.42	49.59%
70	40	100	61.76	100.00%	6.29	10.19%	12.42	20.11%	18.67	30.23%	24.77	40.12%	30.87	49.99%
70	100	40	59.87	100.00%	5.97	9.97%	12.00	20.04%	17.94	29.97%	23.90	39.92%	29.83	49.82%
100	40	70	58.35	100.00%	5.78	9.91%	11.61	19.89%	17.44	29.88%	23.05	39.51%	28.96	49.63%
100	70	40	56.83	100.00%	5.65	9.93%	11.28	19.85%	16.82	29.60%	22.44	39.49%	28.04	49.33%
40	40	40	60.46	100.00%	6.03	9.98%	12.03	19.90%	17.85	29.52%	23.79	39.35%	29.81	49.30%
70	70	70	60.86	100.00%	6.03	9.91%	12.17	19.99%	18.23	29.94%	24.29	39.90%	30.22	49.65%
100	100	100	61.22	100.00%	6.07	9.91%	12.25	20.02%	18.36	29.99%	24.45	39.93%	30.56	49.92%

C. Different Layer Peers Departing

In this peers departing simulation, we first investigated how the whole system is impacted by departing peers in different layer during media object distribution. As shown in Table XII, layers 1, 2 and 3 have different peers' number from 40 to 100. Each layer has a 10 Mbits fix size of the media object. In this experiment, each layer will simulate same 10% to 50% peers departing rate each time from each layer. We calculate the recovery time after peers in each layer leave.

In order to compare the recovery time with that system delivering time without peer departing, we convert them into percentage based on the recovery time to system delivering time in the same table. From Table XII, we can see if each layer has the same percentage departing peers rate, the whole system has a similar recovery time based on the peers departing rate.

Then, we study how the system is affected by a single layer peers departing. In this simulation, layers 1, 2 and 3 have 10 Mbits media object each. They have peers from 40 to 100. As shown in Table XIII, when layer 1 has 10% to 50% peers departing, no peer from layer 2 and layer 3

TABLE XIII

MEDIA DISTRIBUTION TIME T(IN SEC) BY LAYER 1 PEER DEPARTING

Layer 1	Layer 2	Layer 3	0% departing		10% departing		20% departing		30% departing		40% departing		50% departing	
n_1	n_2	n_3	$t(s)$	%	$t(s)$	%	$t(s)$	%	$t(s)$	%	$t(s)$	%	$t(s)$	%
40	70	100	64.59	100.00%	0.81	1.32%	1.63	2.66%	2.45	4.02%	3.23	5.34%	4.02	6.61%
40	100	70	63.70	100.00%	0.84	1.32%	1.69	2.66%	2.56	4.02%	3.40	5.34%	4.21	6.61%
70	40	100	62.01	100.00%	1.45	2.33%	2.81	4.54%	4.18	6.74%	5.54	8.94%	6.82	11.00%
70	100	40	60.00	100.00%	1.51	2.52%	3.03	5.05%	4.52	7.53%	5.99	9.98%	7.42	12.37%
100	40	70	58.01	100.00%	2.08	3.58%	4.08	7.03%	6.04	10.40%	7.90	13.63%	9.72	16.75%
100	70	40	57.25	100.00%	2.22	3.88%	4.35	7.60%	6.43	11.24%	8.41	14.69%	10.40	18.17%
40	40	40	60.65	100.00%	1.47	2.43%	2.90	4.79%	4.31	7.10%	5.69	9.38%	7.08	11.67%
70	70	70	61.23	100.00%	1.49	2.44%	2.97	4.85%	4.41	7.20%	5.82	9.50%	7.19	11.74%
100	100	100	60.89	100.00%	1.48	2.43%	2.94	4.83%	4.35	7.15%	5.73	9.42%	7.05	11.58%

TABLE XIV

MEDIA DISTRIBUTION TIME T(IN SEC) BY LAYER 2 PEER DEPARTING

Layer 1	Layer 2	Layer 3	0% departing		10% departing		20% departing		30% departing		40% departing		50% departing	
n_1	n_2	n_3	$t(s)$	%	$t(s)$	%	$t(s)$	%	$t(s)$	%	$t(s)$	%	$t(s)$	%
40	70	100	64.38	100.00%	1.85	2.87%	3.66	5.68%	5.43	8.44%	7.19	11.17%	8.88	13.79%
40	100	70	63.72	100.00%	2.74	4.30%	5.34	8.38%	7.93	12.44%	10.46	16.41%	12.87	20.19%
70	40	100	61.83	100.00%	1.19	1.93%	2.34	3.78%	3.45	5.58%	4.58	7.40%	5.67	9.18%
70	100	40	60.07	100.00%	2.85	4.74%	5.68	9.46%	8.37	13.93%	11.05	18.40%	13.69	22.79%
100	40	70	58.13	100.00%	1.06	1.82%	2.10	3.61%	3.13	5.39%	4.14	7.12%	5.09	8.75%
100	70	40	56.78	100.00%	1.93	3.41%	3.83	6.74%	5.64	9.92%	7.38	12.99%	8.96	15.78%
40	40	40	60.44	100.00%	1.87	3.09%	3.72	6.16%	5.47	9.05%	7.21	11.92%	8.85	14.65%
70	70	70	60.73	100.00%	1.92	3.15%	3.81	6.27%	5.61	9.24%	7.34	12.09%	9.03	14.87%
100	100	100	61.25	100.00%	1.91	3.11%	3.75	6.12%	5.58	9.11%	7.34	11.99%	9.02	14.72%

leaves. Then recalculate them in percentage of recovery time to system delivering time in the same table. While layer 2 has 10% to 50% peers departing, no layer 1 and layer 3 peers depart. Then recalculate them in percentage of recovery time to system delivering time in the same table. As for layer 3, when it has 10% to 50% peers departing, none layer 1 and layer 2 peers go away. Then recalculate them in percentage of re-cover time to system delivering time in the same table.

From theses tables, we can summarize them in the following items:

TABLE XV

MEDIA DISTRIBUTION TIME T(IN SEC) BY LAYER 3 PEER DEPARTING

Layer 1	Layer 2	Layer 3	0% departing		10% departing		20% departing		30% departing		40% departing		50% departing	
n_1	n_2	n_3	$t(s)$	%	$t(s)$	%	$t(s)$	%	$t(s)$	%	$t(s)$	%	$t(s)$	%
40	70	100	64.84	100.00%	3.87	5.97%	7.48	11.54%	10.96	16.91%	14.12	21.77%	17.21	26.54%
40	100	70	63.51	100.00%	2.72	4.28%	5.35	8.42%	7.81	12.30%	10.17	16.02%	12.35	19.45%
70	40	100	61.71	100.00%	3.72	6.03%	7.38	11.95%	10.95	17.74%	14.39	23.32%	17.52	28.39%
70	100	40	59.75	100.00%	0.10	3.77%	4.40	7.37%	6.59	11.02%	8.90	14.89%	11.10	18.57%
100	40	70	58.32	100.00%	3.44	5.90%	6.80	11.65%	9.85	16.89%	12.95	22.21%	15.80	27.10%
100	70	40	56.81	100.00%	2.38	4.18%	4.63	8.16%	6.91	12.16%	9.01	15.86%	11.17	19.66%
40	40	40	60.77	100.00%	2.87	4.72%	5.66	9.31%	8.37	13.77%	10.83	17.82%	13.07	21.51%
70	70	70	60.88	100.00%	2.80	4.59%	5.58	9.16%	8.31	13.65%	10.76	17.67%	12.99	21.34%
100	100	100	60.86	100.00%	2.85	4.69%	5.55	9.12%	8.25	13.56%	10.68	17.54%	13.06	21.46%

1). The higher percentage peers departing rate, the higher recovery time the system needs.

2). The more departing peers in higher layer, the more time the system will take to recover.

3). If only single peers leave, their recovery time is no more than that of all layers which have the same departing percentage .

D. Different layer peers Joining

In the peers joining experiment, we study a system with two groups of three layers 1 ,2 and 3 peers. Each layer has a 10 Mbits fix size of the media object in each group. We setup two cases to investigate how the increasing peers in different layer affect the whole system media object delivery.

In the first case, three layers 1, 2 and 3 have the same peers numbers before other peers join. Group 1 has 40 peers in each layer and group 2 has 50 peers in each layer, as shown in Table XVI and Table XVII. As we double the peer number on one layer each time, we calculate their percentage of completed time to the original completed time before peers increase.

In the second case with a system of three layers 1, 2 and 3, one layer has different peers numbers from other two layers peers number before other peers join. We increase this layer peers to make the equality of peer number in each layer. To start with, for group 1, one layer has 40 peers and other two layers have 80 peers in each layer. For group 2, one layer has 50 peers

TABLE XVI

MEDIA DISTRIBUTION TIME T(IN SEC) BY SINGLE LAYER INCREASING PEERS SIMULATION 1 ON SAME PEERS NUMBER IN EACH LAYER.

Layer 1 n_1	Layer 2 n_2	Layer 3 n_3	Completion Time $t(s)$	Percent(%)
40	40	40	60.46	100.00
80	40	40	56.88	94.08
40	80	40	62.01	102.58
40	40	80	63.36	104.81

TABLE XVII

MEDIA DISTRIBUTION TIME T(IN SEC) BY SINGLE LAYER INCREASING PEERS FSIMULATION 2 ON SAME PEERS NUMBER IN EACH LAYER.

Layer 1 n_1	Layer 2 n_2	Layer 3 n_3	Completion Time $t(s)$	Percent(%)
50	50	50	60.78	100.00
100	50	50	56.48	92.93
50	100	50	62.22	102.37
50	50	100	63.39	104.29

TABLE XVIII

MEDIA DISTRIBUTION TIME T(IN SEC) BY SINGLE LAYER INCREASING PEERS SIMULATION 1 ON DIFFERENT PEERS NUMBER IN EACH LAYER.

Layer 1 n_1	Layer 2 n_2	Layer 3 n_3	Completion Time $t(s)$	Percent(%)
40	80	80	63.78	104.33
80	40	80	60.12	98.34
80	80	40	58.75	96.11
80	80	80	61.13	100.00

and other two layers have 100 peers in each layer, as shown in Table XVIII and Table XIX. We increase the peer number on the less peers layer to make three layers having the equal number. Then we calculate their percentage of completed time before increasing peer number to the completed time of each layer that has equal peer number.

From Table XVI to Table XIX, we can see that when peers join into the system, the whole delivery time may increase or reduce. This is based on which layer the peers join into. Increasing lower layer peers will reduce the system completion time later. Inversely, increasing higher layer

TABLE XIX

MEDIA DISTRIBUTION TIME T(IN SEC) BY SINGLE LAYER DECREASING PEERS SIMULATION 2 ON DIFFERENT PEERS NUMBER IN EACH LAYER.

Layer 1 n_1	Layer 2 n_2	Layer 3 n_3	Completion Time $t(s)$	Percent(%)
50	100	100	63.79	104.21
100	50	100	60.29	98.50
100	100	50	58.89	96.20
100	100	100	61.22	100.00

peers will extend the system completion time. This is because the lower layer peers can help higher layer peers to distribute lower layer object. It helps the whole system to reduce the delivery time. On the contrary, increasing higher layer peers will results in longer delivery time to complete their layer object distribution by themselves. In the meantime, lower layer peers can't help them. As a result, this will increase the whole system completion time.

In addition, to the above two joining cases, although group 1 and 2 increase different peers number, their increased percentage in system completion time follows the peers' increased percentage instead of physically increasing peers number.

VI. CONCLUSION

In this paper, how to optimally distribute non-scalable and scalable media objects coded to a group of subscribers using P2P network model is addressed. More specifically, how to find the optimal segment sizes is addressed. This problem is formulated as a linear program according to which optimal solutions can be found to minimize the completion time. Based on this method, we have evaluated performance with different numbers of layers, different media object sizes and different numbers of peers and how they impact the system completion time.

Although we can find the optimal distribution for scalable media/video by solving the linear program through lp_solve, there are multiple optimal solutions. Some solutions are more balanced than others. That is, the completion times of individual peers inside one layer are approximately equal. Balanced optimal solutions are preferred, because the impact of departing peers is least. In future work, we will study how to add more constraints to find balanced optimal solutions with the same completion time. We plan to incorporate this factor in a P2P-based media distribution prototype in order to evaluate our method in a real network environment.

ACKNOWLEDGMENT

This work is in part supported by NSF CAREER grant CNS-0546870. Any opinions, findings, and conclusions of the authors do not necessarily reflect the views of NSF.

REFERENCES

[1] A. Sentinelli, G. Marfia, M. Gerla, L. Kleinrock, and S. Tewari, "Will iptv ride the peer-to-peer stream? [peer-to-peer multimedia streaming]," *Communications Magazine, IEEE*, vol. 45, no. 6, pp. 86 –92, june 2007.

[2] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the h.264/avc standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 9, pp. 1103–1120, Sept. 2007.

[3] G. Conklin, G. Greenbaum, K. Lillevold, A. Lippman, and Y. Reznik, "Video coding for streaming media delivery on the internet," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 11, no. 3, pp. 269 –281, mar 2001.

[4] D. Wu, Y. Hou, W. Zhu, Y.-Q. Zhang, and J. Peha, "Streaming video over the internet: approaches and directions," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 11, no. 3, pp. 282 –300, mar 2001.

[5] C.-Y. Lin, C. Chen, and W.-J. Ho, "An adaptive protection framework for mpeg base layer video streaming over internet," in *Information, Communications and Signal Processing, 2003 and Fourth Pacific Rim Conference on Multimedia. Proceedings of the 2003 Joint Conference of the Fourth International Conference on*, vol. 3, dec. 2003, pp. 1365 – 1369 vol.3.

[6] "ISO/IEC IS 13818-2 (MPEG-2 Video): Information technology - generic coding of moving pictures and associated audio information," Apr 1996.

[7] J. Liu, S. Rao, B. Li, and H. Zhang, "Opportunities and challenges of peer-to-peer internet video broadcast," *Proceedings of the IEEE*, vol. 96, no. 1, pp. 11 –24, jan. 2008.

[8] N. Magharei and R. Rajaie, "Understanding mesh-based peer-to-peer streaming," in *Proc. ACM Int'l Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, May 2006.

[9] Y.-H. Chu, S. G. Rao, and H. Zhang, "A case for end system multicast," in *Proc. ACM SIGMETRICS*, Jun 2000.

[10] 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, Florida, Apr 2005.

[11] N. Magharei and R. Rajaie, "PRIME: peer-to-peer receiver-driven mesh-based streaming," in *Proc. IEEE INFOCOM*, May 2007, pp. 1415–1423.

[12] 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 (P2PMMS)*, 2005, pp. 21–28.

[13] "PPStream," http://www.ppstream.com/.

[14] "PPLive," http://www.pplive.com/en/. [Online]. Available: http://www.pplive.com/en/

[15] "UUSee," http://www.uuc.com/.

[16] "SopCast," http://www.sopcast.org/.

[17] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *Proc. ACM SIGCOMM*, Aug. 2002.

[18] D. A. Tran, K. A. Hua, and T. T. Do, "ZIGZAG: an efficient peer-to-peer scheme for media streaming," in *Proc. IEEE INFOCOM*, Apr. 2003.

[19] N. Magharei, R. Rejaie, and Y. Guo, "Mesh or multiple-tree: A comparative study of live p2p streaming approaches," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, may 2007, pp. 1424 –1432.

[20] X. Hei, Y. Liu, and K. Ross, "Iptv over p2p streaming networks: the mesh-pull approach," *Communications Magazine, IEEE*, vol. 46, no. 2, pp. 86 –92, february 2008.

[21] L. Zhao, J.-G. Luo, M. Zhang, W.-J. Fu, J. Luo, Y.-F. Zhang, and S.-Q. Yang, "Gridmedia: A practical peer-to-peer based live video streaming system," in *Proc. IEEE Workshop on Multimedia Signal Processing*, Nov 2005, pp. 1–4.

[22] Bittorrent. [Online]. Available: http://www.bittorrent.com/

[23] B. Cohen, "Incentives build robustness in BitTorrent," in *Proc. First Workshop on Economics of Peer-to-Peer Systems*, May 2003, pp. 251–260.

[24] R. Kumar, Y. Liu, and K. W. Ross, "Stochastic fluid theory for P2P streaming systems," in *Proc. IEEE INFOCOM*, Alaska, USA, 2007.

[25] M. Hefeeda, A. Habib, B. Botev, D. Xu, and B. Bhargava, "Promise: peer-to-peer media streaming using collectcast," in *Proceedings of the eleventh ACM international conference on Multimedia*, ser. MULTIMEDIA '03. New York, NY, USA: ACM, 2003, pp. 45–54. [Online]. Available: http://doi.acm.org/10.1145/957013.957022

[26] H. Cui, X. Su, and W. Shang, "An optimal media distribution algorithm in P2P-based IPTV," in *Proc. 3rd Int'l Conf. on Communications and Networking in China*, Aug 2008.

[27] H. Cui, X. Su, and W. Shang, "Optimal dissemination of layered videos in p2p-based iptv networks," in *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*, 2009, pp. 738–741.

[28] "lp_solve," http://sourceforge.net/projects/lpsolve.