Journal of Communications

ISSN 1796-2021

Volume 7, Number 3, March 2012

Contents

Special Issue: Multimedia Streaming

Guest Editors: Mohammed Ghanbari, Hideki Tode, Mea Wang, and Bin Wei

Special Issue on Multimedia Streaming

Guest Editorial

The penetration of broadband residential access and high-speed wireless access has dramatically increased the demand for multimedia content. As the broadband access rate increases, multimedia streaming applications are embedded in more and more hardware devices, e.g., TVs, cars, and cell phones. In the past decade, multimedia streaming has evolved from simple client-server applications to large-scale Peer-to-Peer (P2P) applications. Hundreds of sites, including CNN, MSN, and Yahoo, have joined the parade of multimedia streaming. Furthermore, sites like YouTube also provide the Video-on-Demand services that allow users to view the video clips from any playback point. In addition to video and audio streaming, the advances in multimedia streaming also stimulate the emerging Internet telephone and television services.

Multimedia applications are significantly different from other conventional networking applications. In particular, multimedia applications are very sensitive to end-to-end delay and bandwidth fluctuation, but are tolerable to occasional data loss. The quality of the video highly depends on the available bandwidth across the network. Endeavors have been made by researchers and application developers to improve the QoS (Quality of Services) in multimedia streaming from all aspects. On the application end, various codec and compression techniques have been proposed, e.g., MPEG and H.261, to reduce the bandwidth demand while maintaining the quality of the multimedia content. In communication networks, protocols and algorithms have been proposed and analyzed, including RTSP, RTP, RTCP, and SIP. From the network architecture point of view, we have the client-server setup and the Peer-to-Peer infrastructure. YouTube is by far the most successful client-server approach to video streaming with the cost of high bandwidth demand at the content source. In contrast, the Peer-to-Peer (P2P) approach invited peers (end hosts) to contribute their upload bandwidth and computational resources, resulting in better scalability, flexibility, and less demand on the servers. The success of P2P multimedia streaming has been demonstrated by systems like Octoshape and UUSee. Despite the advantages and the disadvantages of the existing solutions and technologies, there is no doubt that multimedia streaming is growing at a phenomenal rate.

In this special issue, we present six high-quality publications after a thorough peer-review process. The papers touch upon various aspects of multimedia streaming, from protocol/algorithm design to system analysis/characterization. Interestingly, the papers well reflect the current trends in multimedia streaming, namely, supporting Video-on-Demand functionalities and making the streaming systems more network friendly. The papers can broadly be divided into three categories. The first category proposes systems around different coding techniques. This is very essential as coding techniques can fundamentally changes the way multimedia content being disseminated across the network. The second category proposes systems to enhance viewing experience. As Video-on-Demand becoming an indispensable feature of multimedia streaming, it has received a great amount of attention in real-world applications research and development. It is very important as well as challenging to provide real-time user interaction while maintaining the playback quality in a streaming session. The third category focuses on network friendliness of multimedia streaming systems. The rapid growth in streaming traffics draws concerns for ISPs. To this end, papers in this category propose and analyze various approaches for making multimedia streaming systems more efficient in utilizing network resources.

The first two papers are oriented around two coding techniques, namely, scalable video coding (H.264) and network coding. The first paper, "Distributed Congestion Control of Scalable Video Streams" by J. Wagner and P. Frossard, extends a utility-based congestion control framework to efficiently handle heterogeneous delays in the network and video streaming. The authors proposed an implementation of the media-friendly distributed congestion control scheme for H.264/SVC streams. The simulation results show that bandwidth allocated in the propose systems respects network constraints and converges rapidly to a stable state. The second paper "A Framework for Video Network Coding with Multi-generation Mixing" by M. D. Halloush and H. Radha, proposes a framework for addressing the inefficiency in applying network coding in multimedia streaming. When network coding is applied within a generation of video packets, portions of the video can become undecodable even if only one packet is missing in a generation. The authors proposed to incorporate scalable video coding to enhance the reliability of the streaming service. In other words, lower video layers are encoded with higher layers using network coding to improve the decidability of the original video. The work is very interesting and timely.

The next paper by J. Noh and B. Girod, "Time-Shifted Streaming in a Tree-Based Peer-to-Peer System," presents a new design for live video multicast and Video-on-Demand (VoD) during a live session. The authors extend the Stanford Peer-to-Peer Multicast (SPPM) protocol to support playback control in VoD. The time-shifted streaming refers to the ability to store received video packets at participating peers and to share them with other peers upon request. The system is further detailed with a fast prefetching technique and peer selection scheme to improve the interactive response time of the VoD feature in a live session. The experimental results show that the proposed system significantly reduces the bandwidth demand on the server while providing better video availability.

The remaining three papers focus on reducing traffic imposed by streaming systems. The first paper, "Network Friendly Transmission Control for Progressive Download over TCP" by H. Hisamatsu, G. Hasegawa, and M. Murata, is

doi:10.4304/jcm.7.3.177-179

motivated by the tcpdump analysis on traffic from YouTube and nicovideo to a university network. The measurements show that the video streaming using TCP consumes excessive bandwidth and sends data at a much higher rate than the needed rate. The authors propose a transfer mechanism for video streaming over TCP that are able to maintain the buffers at a reasonable level on the receivers, in order to conserve network bandwidth. The second paper, "Characterizing Locality-Aware P2P Streaming" by J. Zhao and C. Wu, presents an analytical study on the relation between streaming performance and traffic locality in P2P live streaming systems. The paper provides several useful suggestions for reducing cross-ISP traffic volume, including server capacities into ISPs, which concurs with the next paper. The last paper, also our invited paper, "An Analysis and Comparison of CDN-P2P-hybrid Content Delivery System and Model" by Z. Lu, Y. Wang and Y. R. Yang, provides a survey on the recently emerging ideas on combining stable edges in content delivery networks (CDN) and scalable last-mile transmission from P2P networks, to control the traffic volume and to make streaming systems more ISP friendly. The paper points out that it is important to address the reliability, security, and ISP-friendliness in multimedia streaming systems. Furthermore, we still need better ways to integrate CDN and P2P without introducing additional overheads.

We feel that this special issue succeeds in its attempt to give readers an insight into on-going activities on fundamental as well as applied research in multimedia streaming systems. We would like to thank all the authors who submitted their work for this issue, and the reviewers for their timely and constructive feedback. With these papers, we hope to provide readers a comprehensive view of our current achievements and shed lights on future research direction in multimedia streaming. We also thank the staff at the JCM Academy Publisher for their help in handling the manuscripts. Lastly, we would like to extend our sincere appreciation to Dr. Haohong Wang, Editor-in-Chief of the Journal of Communications, for his great support and providing us the opportunity to organize this special issue.

Guest Editors (in alphabetic order of last name)

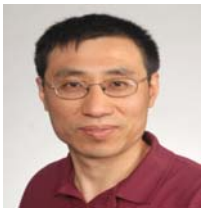
Mohammed Ghanbari, University of Essex, UK (ghan at essex.ac.uk)
Hideki Tode, Osaka Prefecture University, Japan (tode at cs.osakafu-u.ac.jp)
Mea Wang, University of Calgary, Canada (meawang at ucalgary.ca)
Bin Wei, AT&T Research, USA (bw at research.att.com)

Mohammed Ghanbari is an Emeritus Professor at the School of Computer Science and Electronic Engineering, University of Essex, UK with the main research interest in the areas of Video Networking . He had the Chair of Video networking from 1996-2011 at the same school. He is best known for his pioneering work on layered video coding (which earned him IEEE Fellowship in 2001), now is known as SNR scalability in the standard video codecs. He has registered for eleven international patents and has published more than 550 technical articles on various aspects of video networking. He has authored/co-authored 6 books and edited a book, where his book *Video coding: an introduction to standard codecs,* published by IET press in 1999, received the year 2000 best book award by IET. He is a Fellow of IEEE, Fellow of IET and Chartered Engineer (CEng).

Hideki Tode received the B.E., M.E., and Ph.D degrees in communications engineering from Osaka University, Japan, in 1988, 1990, and 1997, respectively. He was adopted as assistant professor at the Department of Communications Engineering, Osaka University on Dec. 1991. In 1998 and 1999, he promoted to a lecturer and associate professor in the Department of Information Systems Engineering, and, in 2002, shifted in the Department of Information Networking, Graduate School of Information Science and Technology, Osaka University. From 2008, He is a professor in the Department of Computer Science and Intelligent Systems, Graduate School of Engineering, Osaka Prefecture University. His current research interests include high speed router architecture, QoS-aware network controls, optical network architecture and application-level contents distribution technologies. Dr.Tode is a member of IEEE, and a senior member of IEICE Japan.

Mea Wang received her Bachelor of Computer Science (Honours) degree from the Department of Computer Science, University of Manitoba, Canada, in 2002. She received her Master of Applied Science and Ph.D. degrees from the Department of Electrical and Computer Engineering at the University of Toronto, Canada, in 2004 and 2008, respectively. She is currently an Assistant Professor in the Department of Computer Science at the University of Calgary.

Her research interests include peer-to-peer networking, multimedia networking, cloud computing and networking system design and development. Her work on practical network coding for P2P multimedia streaming system, "R2: Random Push with Random Network Coding in Live Peer-to-Peer Streaming," has been highly recognized and won the 2009 Multimedia Communications Best Paper Award. She has been serving as an Associate Editor for the Journal of Communications since 2010, the Guest Editor of the Special Issue on "Multimedia Streaming (P2P)" and the Special Issue on "Multimedia Streaming (Scalability)" for IEEE Multimedia Communications Technical Committee E-Letter in October and December 2009, respectively.



Bin Wei is a research staff member at AT&T Labs - Research. He has been working on multimedia, communications and middleware support for various user devices, ranging from a display wall to handheld mobile devices. He has published many papers in major conferences. Currently, he is serving as the Vice Chair for IEEE Multimedia Communications Technical Committee and a Steering Committee member for International Conference on Multimedia and Expo (ICME). He received a Ph.D. on Computer Science at Princeton University. He is a Senior Member of ACM.

Distributed Congestion Control of Scalable Video Streams

Jean-Paul Wagner and Pascal Frossard
Ecole Polytechnique Fédérale de Lausanne (EPFL)
Signal Processing Laboratory - LTS4
Switzerland - 1015, Lausanne
Email: {jean-paul.wagner, pascal.frossard}@epfl.ch

***Abstract*— The paper addresses the problem of distributed congestion control in networks where several video streaming sessions share joint bottleneck channels. Based on the rate-distortion characteristics of the video streams as well as the requirements of heterogeneous streaming clients, the proposed algorithm determines the utility of rate increments in terms of overall quality benefits. We build on the utility-based congestion control framework initially proposed by Kelly [1], [2] and extend it such that it could efficiently handle heterogeneous delays in the network as well as the specific requirements of video streams. We then describe an original implementation of the proposed distributed congestion control algorithm using the common RTP/UDP/IP protocol stack for the efficient streaming of scalable video streams. Each video session independently adapts its streaming rate based on receiver feedbacks, which permits extension to large scale system without central coordination. Finally, we provide extensive simulation results that demonstrate the performance of the proposed solution. It is shown to successfully distribute the network resources among the different sessions such that the overall video quality is maximized. Experiments further demonstrate that the proposed scheme fairly cohabits with TCP, which is certainly an important advantage in today's network architectures.**

I. Introduction

The Internet has been rapidly evolving into a transport medium for the distribution of data such as audio and video streams due to the emergence of attractive multimedia applications. Multimedia streaming applications impose different requirements than those underlying typical FTP or HTTP traffic that is controlled by the Transport Control Protocol (TCP) [3]. On the one hand, the media streams have to be delivered at a rather high and sustained rate in order to deliver an acceptable Quality of Service (QoS) to the media client. On the other hand, an encoded media stream generally carries significant redundancy and therefore presents some inherent robustness to packet loss. The experience of the the end user is mostly driven by the delay and the fluctuations of quality in media streaming applications. Unsurprisingly, TCP does not perform very well in regulating the rate of media streams since it has not been designed for controlling such data flows. In particular, the typical sawtooth behavior in the controlled transmission rate profiles and the delays experienced by some packets due to the retransmission and exponential back-off policies are not ideal for media streaming. TCP-friendly rate control (TFRC) [4] permits to reduce the fluctuations of the transmission rates. It is more appropriate for delay-sensitive data and still offers a fair distribution of network resources. However, both TCP and TFRC do not consider the relative characteristics of the media streams and thus cannot lead to optimized performance in terms of overall quality of experience.

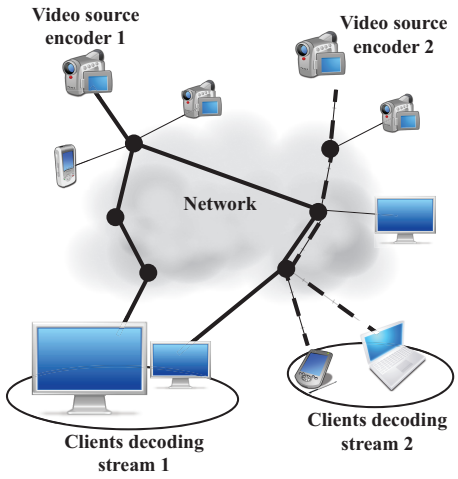


Fig. 1. A multi-session media streaming system whose objective is to deliver the best possible media quality to a set of heterogeneous clients while keeping the network stable in dynamic scenarios.

In this paper we explore an alternative rate and congestion control algorithm that is adapted to the specific characteristics of media streams receivers. We consider the framework illustrated in Figure 1. The media sources are captured and encoded in various points at the edges of an overlay network. Clients with heterogeneous needs in terms of frame rate or display resolution capabilities can connect to any media source from anywhere in the network. Our objective is to jointly achieve network stability with a fair distribution of resources in terms of video quality for all decoding clients, even in the presence of changes in the network.

We propose a *media-friendly* control algorithm, which

Manuscript received April 15, 2011; revised July 15, 2011; accepted October 15, 2011
This work has been partly supported by the Swiss Innovation Promotion Agency, under grant CTI - 7388.2 ESPP-ES.

doi:10.4304/jcm.7.3.180-191

adapts the rate in the network to the needs of the media streams. Our work is based on a class of distributed congestion control algorithms that have been first proposed by Kelly [1], [2]. Their general objective is to maximize the aggregate *utilities* the end users retrieve from their network usage. The family of congestion control algorithms described in [1], [2] have the benefit of i) potentially allocating smooth rate profiles without abrupt transitions along time and ii) distinguishing each flow based on a *utility* criterion. Since we target a fair distribution of media quality in the client population, we propose to use a utility criterion that captures the quality benefit of an increment of bandwidth for each media stream. We extend the utility-based congestion control algorithms to practical systems with heterogeneous feedback delays and we show that the algorithms remain stable in these imperfect settings. Then we compute generic utility curves for scalable video streams, which permit to define several levels of quality by changing the spatial, temporal or SNR resolutions and hence to fully use the flexibility of scalable video streams. The use of a distortion-based criterion has been proposed in [5] but in more restricted settings that do not consider the different scalability dimensions. We finally propose a novel implementation of the media-friendly distributed congestion control scheme for various streaming scenarios where H.264/SVC streams are transmitted using RTP/UDP/IP protocols equipped with a new client feedback mechanism for network state inference. We provide extensive simulation results that demonstrate the convergence of the distributed congestion control algorithm, even in dynamic environments. The results also show that the bandwidth is properly distributed among the clients such that the quality is properly adapted in all concurrent streaming sessions. Finally, the proposed algorithm is shown to cohabit fairly with TCP flows.

The remainder of this paper is organized as follows. In Section II we provide some background on Kelly's theoretical framework [1], [2]. We extend it in Section III with an important stability proof for imperfect network settings. In Section IV we describe the proposed distributed congestion control algorithm and show how we can incorporate media-friendliness through an effective choice of utility functions adapted to scalable video streams. In Section V we describe the careful and fully scalable implementation of the proposed algorithm that can significantly boost the performance of the congestion control in practical scenarios. In Section VI we validate our findings through extensive NS-2 simulations. Finally we conclude with Section VII.

II. Background

In this section, we provide some background and references on Kelly's Network Utility Maximization (NUM) problem that has been first stated in [2]. We give first a brief overview on distributed algorithms that solve the aforementioned problem. Then we provide a stability proof that is of high importance for the practical framework studied in this paper. Namely, we show that a system controlled using the NUM framework is stable in the case of general rate utility functions and arbitrary round-trip delays. Finally, we discuss the *fairness* between flows resulting from the rate allocations computed in the NUM framework.

A. Network Utility Maximization

Let $s_i(t)$ be the rate assigned to flow i at time t. Further, assume that we assign to each flow i a *utility function*. This function describes by a utility value $U_i(s_i(t))$ the usefulness of the streaming rate $s_i(t)$ for an application that is served by flow i [6]. We suppose that the utility functions are continuous functions of the rate and that they are concave. Let $\mathcal{I}$ be the set of all flows in a given network, characterized in turn by a set of network links l with their respective capacity constraints.

The original NUM problem consists in finding for every time t a set of rates $s_i(t)$ that maximizes the sum of utilities for all flows, $\sum_{i\in\mathcal{I}} U_i(s_i(t))$, while satisfying the rate constraints on each link in the network. The direct solution of this optimization problem is difficult to find for large networks, and it further assumes that a central entity controls the rate of each flow. Instead, Kelly has proposed in [2] to solve a relaxed version of the original problem, by adjusting the rate of the flows in order to satisfy the following differential equations at any time t:

$$\frac{d}{dt}s_i(t) = \kappa_i s_i(t)\left(U_i'(s_i(t)) - p_i(t)\right). \qquad (1)$$

The term $s_i(r)\cdot U_i'(s_i(t))$ is referred to as the *willingness to pay* for the resource (i.e., the available bandwidth) and $p_i(t)$ is a pricing function, updated by the network, which indicates the price of the offered resource. In practice, $p_i(t)$ is often a congestion indication function that is fed back to the controller from either the network or the end user and κ_i is a constant gain factor.

Note that this system of differential equations drives each rate $s_i(t)$ into a steady state in which there is an equilibrium between the willingness to pay, and the flow's contribution to the resource's price. The former depends on the utility function $U_i(s_i(t))$ through its gradient, while the latter depends on the current network state, which is expressed by the congestion indication function $p_i(t)$. It is worth noting that Equation (1) can be straightforwardly discretized and turned into a rate update equation that leads to a practical implementation of a distributed control algorithm for solving the original NUM problem. To date, this has yielded the most promising implementations of control algorithms, even if there are other ways of solving the original NUM problem in a distributed way [7].

B. Fairness

The control algorithms based on Kelly's framework provide a fair distribution of the network resources among the different competing flows in the network. The fairness characteristic might however be defined in several different ways. In particular, the fairness could be linked to the distribution of the network bandwidth, or rather to

the distribution of network resources that balance utilities among flows. We refer to [8], [9] for a detailed study on the fairness in Kelly's framework. For the sake of completeness we provide a short summary here:

- If the controller relies on end-to-end feedback measures only, the control algorithm provides *proportional fairness*, meaning that flows congesting multiple routers are allocated less bandwidth in the stable state than flows congesting less routers.
- If the controller has access to the congestion levels of each router, which is however not a realistic scenario in today's Internet, the controller can adjust the rate for each flow according to the most congested router, thus providing *max-min fairness* [10].
- If additionally each flow is characterized by a potentially different utility function, the resulting rate allocations are weighted by the respective utility values in the stable state.

The practical system that we propose in this paper relies on end-to-end measures and uses different utility function for each flow. Hence, the control algorithm based on Kelly's framework provides in this case a *utility-proportional* fairness in the bandwidth allocation.

Finally, both stability and convergence of the distributed congestion control algorithm have been extensively studied for systems based on the differential equations given in (1) (see for example [2], [11]–[13] and references therein).

III. Stability with random feedback delays

The ideal system described in the previous section has only marginal practical relevance since it assumes that network or client feedbacks are immediately available at each source in order to update the rate of the corresponding flow. In practical systems however, each flow i has a fixed starting point (the sender) as well as a fixed end point (the receiver) in the network. The route that connects them may change in time, which therefore affects the round-trip delay between sender and receiver. Moreover, the congestion level of each router that is traversed by the flow is dynamically changing. The delay experienced by feedback information is not only heterogeneous across flows in the network, but it has also a random distribution for each flow. When the system experiences some delays in the feedback loops, the differential equation of Eq. (1) reads as:

$$\frac{d}{dt}s_i(t) = \kappa_i \left(s_i(t)U_i'(s_i(t)) - s_i(t - D_i^R)p_i(t - D_i^B) \right), \tag{2}$$

In this case, D_i^R is the round-trip delay and D_i^B is the delay on the back-trip from the point in the network that created the feedback to the sender. Note that the congestion indication function $p_i(t - D_i^B)$ is synthesized D_i^B time units earlier, and that it relates to the rate allocated by the controller D_i^R time units earlier.

In the last years, a substantial amount of research has been devoted to providing stability results for controllers with delayed feedbacks. For example the authors in [11] provide results for the case of equal round-trip delays for each flow. More recently, authors in [14] have studied the stability of systems with arbitrary but constant round-trip delays. Further stability results are provided in [10], [15]–[19] and references therein. In each of these works, the scenario corresponds to particular applications: either each flow uses the same utility function, or the delays are different for the various flows but constant in time, or the parameters of the rate update equation are dynamically adjusted with respect to the experienced delay. The latter implies that a precise measurement of the experienced round-trip delay is available at the controller, so that oscillations can be avoided in the system. The application considered in this paper does however not correspond to any of these scenarios. Therefore we need to extend the stability proofs to a more generic case.

The application we are targeting specifically calls for stability results for a scenario with *general* concave utility functions for each media stream and arbitrary round-trip delays. In [20], the author provides an elegant stability proof for the case where general utility functions are used, and where the round-trip delays in the network are arbitrary across flows, but constant in time. The author conjectures that the system remains stable if the round-trip delays take on arbitrary values. In what follows we borrow the framework from [20] and we extend the stability result to the case where generic, concave utility functions are used and where the feedback is arbitrarily delayed for each flow.

We consider a network made up of L links, indexed by l. We call $\Delta_{l,i}$ the round trip delay experienced by data transmitted from source i to traverse link l and get back to the source. Hence the experienced round trip delay D_i^R for user i takes on values in the set $\{\Delta_{1,i}, \ldots, \Delta_{L,i}\}$, depending on the receiver of flow i. Further we denote by Δ_l the maximum round trip delay for data to get sent from any source, traverse link l and get back to the source: $\Delta_l = \sup_i\{\Delta_{l,i}\}$. Let $\overline{D}$ be an upper bound on all experienced round trip delays in the network: $\overline{D} = \sup_i\{D_i^R\}$. Clearly, the following relation holds:

$$\overline{D} \geq \Delta_l, \quad 0 \leq l \leq L. \tag{3}$$

The main result of [20] states that the system of delayed differential equations (2) is asymptotically stable if $\kappa_i > 0$, $U_i(\cdot)$ is a concave function for each i, and if all the round trip delays in the network coincide with a single scalar, i.e., $D_i^R = D, \forall i$. A sufficient condition for this to be the case is that the spectral radius of matrix N, given as

$$N = D \sum_{0 \leq l \leq L} M^{(l)} \tag{4}$$

is smaller than 1. This is shown to be the case *for any scalar D*, if the matrix M, which does not depend on the round trip delays, is positive definite. This condition is in turn verified if $\kappa_i > 0$ and the utility functions $U_i(\cdot)$ are concave. We now extend this stability result to arbitrary feedback delays.

Theorem 1: Assume that the round trip delays D_i^R take on arbitrary values bounded by the scalar $\overline{D}$. Then the system given in Eq. (2) is asymptotically stable under the assumptions that $\kappa_i > 0$ and the Utility functions $U_i(\cdot)$ are concave $\forall i$.

Proof: Following the same reasoning as in [20], a sufficient condition for this to be true is that the spectral radius of matrix N' is smaller than 1, with

$$N' = \sum_{0 \le l \le L} \Delta_l M^{(l)}, \tag{5}$$

and M the same as in the previous development. Let us introduce the following matrix:

$$\overline{N} = \overline{D} \sum_{0 \le l \le L} M^{(l)}. \tag{6}$$

From Eq. (4) we know that the spectral radius of $\overline{N}$ is less than 1. Using Eq. (3) we can further bound the elements of matrix N' as follows:

$$N' \le \sum_{0 \le l \le L} \overline{D} M^{(l)} \tag{7}$$

The spectral radius of the right-hand side of (7) is smaller than 1. The proof is concluded using the following Lemma, which states that the spectral radius of the right-hand side of (5) is smaller than the one of the right hand side of (7), hence it is also smaller than 1.

Lemma 1: Let A and B be two matrices, the entries of which, indexed by couples (n, m), all satisfy

$$|A_{n,m}| \le B_{n,m} \tag{8}$$

and hence, the $B_{n,m}$ are real, nonnegative. Then, the spectral radius, i.e., the largest positive eigenvalue of A, is smaller or equal to that of B. [20]

■

This result of stability is important for the design of the congestion control algorithm proposed in the next section. It is moreover most relevant to any practical system as in practice the round trip delays that are observed are always bounded.

IV. Distributed Rate Allocation Algorithm

We propose now a distributed rate allocation algorithm for scalable video sequences based on the utility maximization framework described above. We first present a discretized version of the system of differential equations given in the relation (2), which leads to discrete-time rate update equation. Then we propose utility functions that are adapted to practical scenarios for video streaming.

A. Rate update equation

The distributed control algorithm can only act at discrete time instants in practice. An approach based on finite differences can be used to obtain a discrete-time version of the Eq. (2) as suggested in [11]. This results in the following update equation:

$$\begin{aligned} s_i(t) &= s_i(t-1) + \kappa_i \big(s_i(t-1) U_i'(s_i(t-1)) \\ &\quad - s_i(t - D_i^R) p_i(t - D_i^B) \big), \end{aligned} \tag{9}$$

The congestion indication function (pricing function) for flow i can be computed based on the *received* rate $r_i(t - D_i^B)$ measured at the client at time $t - D_i^B$. In this case, it reads

$$p_i(t - D_i^B) = \frac{s_i(t - D_i^R) - r_i(t - D_i^B)}{s_i(t - D^R)}. \tag{10}$$

Note that delays induce a temporal shift between the time at which the reference sending rate, the actual sending rate, and the received rate are computed in the system. This slows down the convergence of the system. Using Eq. (9), the sender updates the rate $s_i(t)$ using the last taken control $s_i(t-1)$ as reference rate and a delayed feedback term. This feedback represents pricing information about the control taken D_i^R time units earlier. For example, if there is no congestion, the received rate is equal to the sending rate at time $t - D_i^R$. In that case the update equation will lead to a pure increase of the rate. In the event of a congestion, only part of the sending rate is received so that $r_i(t - D_i^B) < s_i(t - D_i^R)$. Hence the price of the resource for flow i is increased and the rate will be re-adjusted downwards accordingly. Note that the receivers provide periodic feedbacks to the servers and that these feedbacks do not correspond to specific frames. The feedbacks are generally not instantaneous; if they are frequent enough the servers however react properly to changing network conditions and the system stays stable.

In order to cope with delays and improve the stability of the system, we can rather use $s_i(t - D_i^R)$ as the reference rate for the update equation [16], [21], so that the temporal drift between reference rate and pricing function is virtually cancelled. However, differently from [21] we do not intend to use a constant *willingness to pay* throughout the network, as the rate allocation should reflect the relative utilities of the various streams. Hence, in order to eliminate the temporal drift between the term of Eq. (9) that increases the reference rate and the feedback term that penalizes the reference rate, we need to evaluate the Utility function at the rate given by $s_i(t - D_i^R)$ as well. Finally, the proposed controller use the following discrete-time rate update equations:

$$\begin{aligned} s_i(t) &= s_i(t - D_i^R) + \kappa_i s_i(t - D_i^R) \\ &\quad . \ \left[U' \left(s_i(t - D_i^R) \right) - p_i(t - D_i^B) \right]. \end{aligned} \tag{11}$$

From a networking point of view, it can be inferred from the rate update equation (11) that the average experienced loss rate $p_i(\cdot)$ is equal to the gradient value of the utility function, evaluated at the rate operation point that is reached in the stable state. The controller then maximizes the system's aggregate utility by iteratively allocating more rate to streams that have a larger benefit at the current rate operation point.

As the normalized congestion signal $p_i(\cdot)$ takes values in the interval $[0, 1]$, the utility gradient values of any stream in the system are normalized by a system-wide constant. This constant corresponds to the largest gradient value for utility functions in the system. It is important to note that the normalization constant should not be

stream-dependent in order to avoid distortions between the relative utility curves assigned to different streams. The maximum gradient value in the system also drives the maximum observed stable-state loss rate per stream throughout the system. Hence, by correctly scaling all the gradients, we can bound the maximum loss rate π to the maximum value that can be tolerated by the target application. Hence, we can finally rewrite the Eq. (11) as:

$$\begin{aligned} ccs_i(t) &= s_i(t - D_i^R) + \kappa_i s_i(t - D_i^R) \\ &\cdot \left[\pi \cdot U'\left(s_i(t - D_i^R)\right) - p_i(t - D_i^B)\right] \quad (12) \end{aligned}$$

B. Video Utility Functions

In the analysis about the stability of the distributed rate allocation algorithm, the only assumption on the continuous utility functions relies in their concavity. It leaves quite some flexibility in the choice of these functions, such that they correspond to the characteristics of the target application. An obvious choice for utility functions is to relate the utility of a media stream to the rate-distortion characteristics of the encoded sequence. In that case the control algorithm iteratively allocates the rates among streams proportionally to their contribution to the average video quality computed on all the streams. We focus in the rest of this paper on video encoders that provide the possibility to generate traffic that can be tuned to achieve any rate within a bounded rate interval. Such encoders include for example Motion-JPEG2000, in which each frame is progressively intra-coded: the rate of each frame can thus be adapted by selectively dropping wavelet coefficients, while increasing the distortion gracefully. Another example is given by the progressive refinement (PR) slices of the scalable video coding (SVC) amendment to the H.264/MPEG-4 AVC standard. Using SVC, a video can be decoded at the full encoding rate, yielding the highest possible decoded quality in terms of SNR, frame rate and resolution. Aside from this, one has the option to decode a number of sub-streams that have been specifically included while encoding. Each of the sub-streams represents a version of the video that is degraded in either SNR, frame rate or spatial resolution, or any combination of these. Typically it is up to the application to decide which sub-stream is most useful. Finally, each of these sub-streams can be encoded using progressive refinement slices, providing fine granularity scalability (FGS). These slices can be cut at any point in order to finely tune the rate within a substream, while gracefully increasing the distortion. Both of the aforementioned encoding choices rely on fine grained rate adaptation that results in SNR scalability. Hence they are able to generate the kind of traffic we aim for: any rate within a bounded rate interval can be achieved. As this results in SNR scalability, the resulting rate-distortion curves over that interval are concave and can be used as utility functions for the respective streams.

Examples of the utility functions used in this paper are illustrated in Figure 2 for a number of test sequences at different frame rates and resolutions. They have been

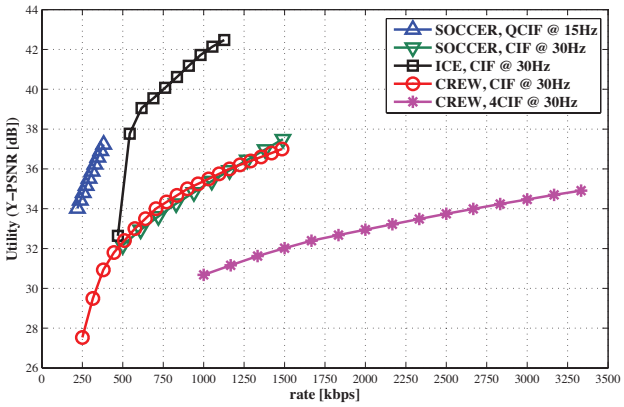


Fig. 2. Utility curves can reflect the rate-distortion characteristics. Here we show the rate-distortion curves of several H.264-SVC(FGS) encoded test sequences (SOCCER, CREW, ICE).

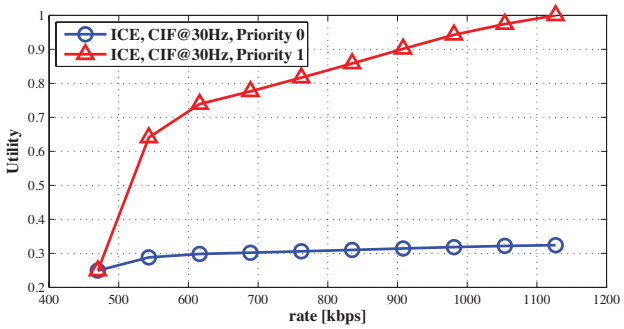


Fig. 3. Utility curves can also reflect traffic prioritization. Here priority class 0 uses the (scaled) R/D information for the SOCCER sequence, whereas users in priority class 1 use a utility function that has a gradient five times as large at each rate point.

computed on video sequences encoded using the H.264 SVC extension and using PR slices. The sequences have first been segmented into Groups of Pictures (GOPs) of equal size, and each GOP has been encoded independently into PR slices. The utility functions for the complete sequence have finally been extracted by decoding each GOP at a number of fixed rate points, and by averaging the resulting Y-PSNR value at each rate point for all the GOPs of the sequence. Note that we do not rely on an analytical model to specify the utility curves for each stream contrarily to [5], [22], but we rather use the exact rate-distortion information.

Finally, it should be noted that our framework is very generic and that any concave continuous function is valid as utility function. In particular, priority or service classes can also be incorporated in the utility functions. For example, we can design utility functions for several groups of streaming clients interesting in the same video sequence. The rate for receivers in the lower priority class is adapted using to the rate-distortion function. The receivers in the higher priority class use a different utility function, whose gradient corresponds to a scaled version of the rate-distortion curve at any rate point. Clients in the higher priority class hence see the quality of their

video streams be adapted faster and reach a higher level. Such utility functions are illustrated in Figure 3, where the utility gradient is multiplied by five for the high priority class.

V. Practical Implementation

We propose now a practical implementation of the control system described above. We have shown that the distributed control algorithm is stable, even with heterogenous feedback delays, which are likely to happen in real scenarios. We outline the scalable nature of the proposed framework and we explain in detail the design choices for the rate update equation and the distributed control in a video streaming system.

A. Design issues

From the above development, it is clear that the rate update equation (see Eq. (12)) has ideally to be applied as often as possible in order to emulate a continuous time control system. This requires the availability of accurate feedback at each moment in time and at each end-point in the network. At the same time, it implies that the sending rate can be adapted at any time instant. However, when we are dealing with real video streams, it is clear that we cannot adapt the video rate at any arbitrary time instant. For example, dropping random parts of the bitstream results in large and uncontrolled losses of quality due to the inherent decoding dependencies between video elements. Scalable video coding provides an interesting solution for flexible adaptation of the bitstream. For example, encoding formats such as H.264-SVC(FGS) form independently decodable compressed units such as Groups of Pictures (GOP), which are typically groups of 16 or 32 frames. They further offer the possibility to extract a substream of a given rate from any independently decodable entity of the stream. The rate control has therefore to be performed on GOPs, and the rate update equation of the control system should be synchronized with GOP boundaries. It is applied after each transmitted GOP and defines the rate of the substream to be extracted from the next GOP.

The decision of the control algorithm relies on the feedback received from the clients, about the state of the streaming session. In addition, it uses the result of the decisions taken in previous iteration of the algorithm, as seen in the update equation (12). The controller has to know the sending rate that has been computed $t-D_i^R$ time units earlier, where D_i^R is the *arbitrary* round trip delay. Maintaining the history of earlier decisions for each flow at each sender does however not provide a viable solution as it does not scale. In addition, it relies on very accurate round trip Time measurements in order to avoid any drift between the sending rate that is actually used in the update equation and the feedback that is received. The utility function is evaluated on the sending rate, yielding the willingness to pay for the bandwidth that is offered, while the congestion signal that is fed back gives the price of the bandwidth resources. Any temporal drift between the computation of these values slows down the convergence of the distributed algorithm. A scalable system has therefore to avoid any temporal drift by grouping together the congestion signal and the sending rate it corresponds to. In the next section, we propose a light-weight application layer protocol that respects the above constraints using the time-stamp information included in the transported video streams.

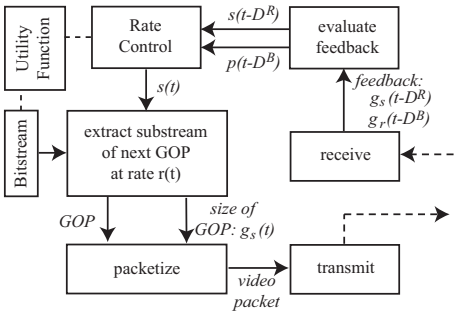


Fig. 4. Schematic view of our sender implementation.

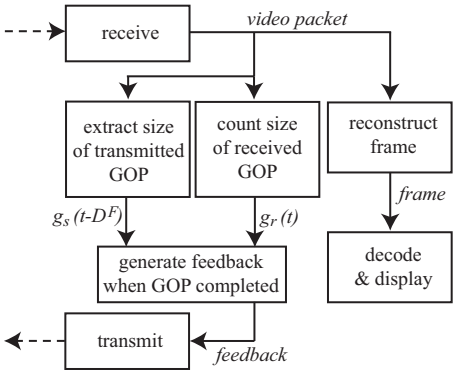


Fig. 5. Schematic view of our receiver implementation.

B. Proposed control system

We propose now an implementation of the distributed control system at the application layer, for streaming scalable video over the classical RTP/UDP/IP protocol stack. For the sake of clarity we drop the index i that specifies a particular flow and sender/receiver pair in what follows. We describe in details the behavior of a client and sender pair; all the concurrent streaming sessions in the system adopt the same strategy.

A schematic view of the sender implementation is first given in Figure 4. Upon reception of a feedback message, the rate controller given by Equation (12) updates the targeted sending rate to $s(t)$, which depends on the media stream that is being transmitted through the chosen utility function. The next GOP of the scalable bitstream is then optimally truncated so that its rate matches the target rate. The resulting substream is then packetized according to the video frames and Network Abstraction Layer (NAL) units and injected into the network. The exact size of the extracted GOP, denoted by $g_s(t)$, is added to the header of all the packets in the GOP. Note that the sender knows the

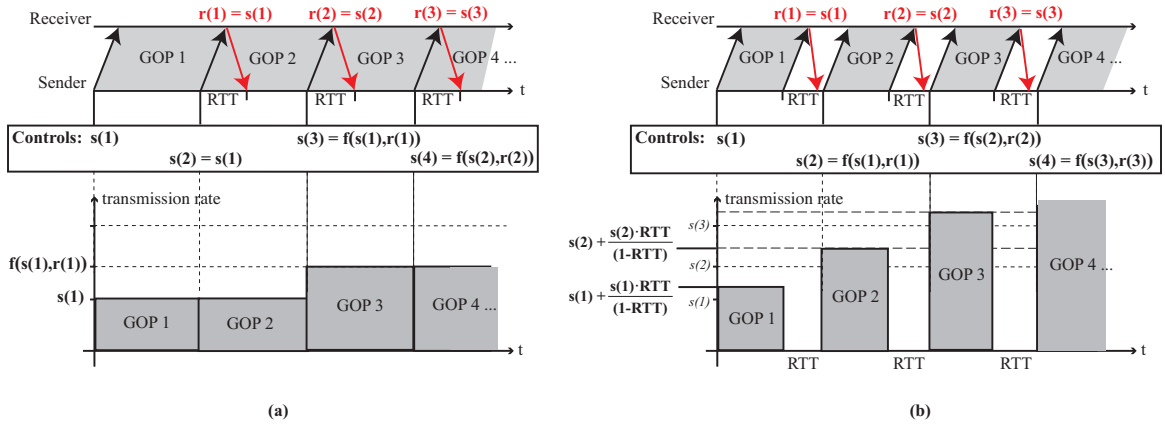


Fig. 6. Guard interval illustration. *Left:* When no guard interval is used, the available feedback is only considered in the control loop at a later stage. For example, the rate for GOP 3 is dependent on the rate and feedback of GOP 1. *Right:* By using a guard interval, the feedback can be integrated more rapidly in the control decisions.

framerate f [Hz] of the stream as well as the length k of the GOP in frames, since those are usually negotiated by the sender and the receiver. The sending rate is therefore simply given by

$$s(t) = \frac{g_s(t)}{k} f \ . \tag{13}$$

The client behavior is then represented in Figure 5. It receives the packet stream and detects potential packet losses. It further reassembles the media bitstream and sends the reconstructed decodable parts of the bitstream to the decoder. At the same time, the receiver counts the number of bits $g_r(t)$ that are received for the current GOP. Periodically, the receiver sends a feedback to the sender where it includes the received size $g_r(t)$ as well as the GOP size $g_s(t - D^F) \geq g_r(t)$ that is read from packet headers. When the feedback eventually reaches the sender, it accurately reconstructs the control decision taken $D^R = D^F + D^B$ time units earlier. Even if the round trip time D^R is arbitrary, the sender can replicate the past control decision and compute the previous sending rate as

$$s(t - D^R) = \frac{g_s(t - D^R)}{k} f. \tag{14}$$

Similarly it reconstructs the rate that is actually received by the client as:

$$r(t - D^B) = \frac{g_r(t - D^B)}{k} f \ . \tag{15}$$

The sender has thus access to all the information necessary to evaluate the congestion signal given in Equation (10). Note that it uses the reference sending rate $s(t-D^R)$ without relying on a stored history of controls nor on exact round trip delay measurements. All the information is rather extracted from the feedback packets and the media stream characteristics that are known at the sender. Furthermore, the information that is transmitted in the feedback packet can be accurately synthesized at each client even when media packets are lost since all the packet headers contain the GOP size information $g_s(t)$. It is therefore sufficient to receive one media packet of the GOP for the client to generate an accurate feedback signal that stabilizes the control system.

Even if such a system is scalable and robust, it does not perform optimally yet due to network latency. Consider for example the scenario depicted in Figure 6(a). The feedback is only sent after a GOP n is completely received. Hence, it is available at the sender at the earliest one round trip time after the last packet of the GOP has been transmitted. At the time the controller has to decide on the sending rate for the next GOP $n + 1$, it does not have access to any feedback yet. It can thus only decide to keep on transmitting at the same sending rate. Even though an accurate feedback becomes available during the transmission of the GOP $n+1$, it can not be incorporated in the control system due to the structure of the video streams. Therefore, it can only affects the sending rate of the GOP $n+2$, which introduces a latency that is almost equal to the duration of one GOP.

In order to avoid such a latency that slows down the convergence of the control system towards a steady-state solution, we propose to increase slightly the actual transmission rate. The data of a GOP with rate $s(t)$ is thus transmitted at a rate $s(t) + \frac{s(t) \cdot RTT}{1-RTT}$, where RTT is the round trip time in the system. It therefore creates a guard-interval between the transmission of two adjacent GOPs n and $n+1$, so that the feedback information about the GOP n is likely to be received on time for controlling the sending of the GOP $n + 1$. Figure 6(b) sketches the improved control sequence. The small increase in the transmission rate permits to adapt the sending rate of the streaming session faster compared to the case illustrated in Figure 6(a).

VI. Simulation Results

A. Setup

We illustrate now the behavior of the distributed rate allocation algorithm with simulation results in different streaming scenarios. We consider the general network topology depicted in Figure 7, where eight different

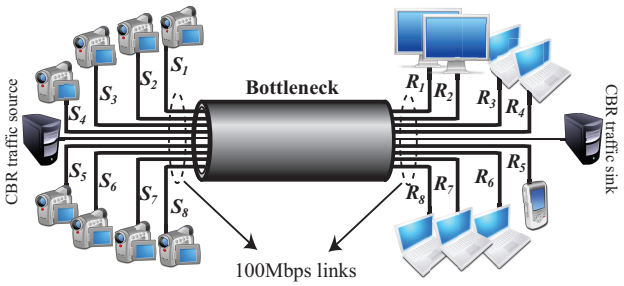


Fig. 7. Simple simulation topology, each source streams SOCCER CIF @ 30Hz

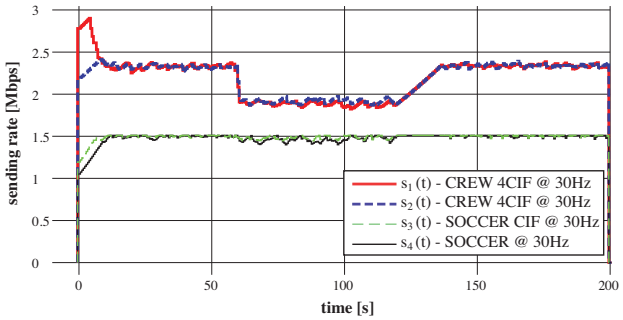


Fig. 8. Resulting sending rates/controls for scenario 1.

sender-receiver pairs connect to a network through high-speed links but share a common bottleneck link. Each sender-receiver pair runs its own rate-control loop independently of the other pairs. It relies only on end-to-end feedbacks and the Utility information relative to the video sequence. In particular, each sender and receiver know neither the capacity of the bottleneck link, nor the number and nature of the other streams that use the same bottleneck link. Finally, we also consider a constant bit rate (CBR) source-sink that also uses the same bottleneck link without any adaptive rate control.

The sequences transmitted by the senders are encoded using the H.264 SVC reference software (JSVM). In the encoding, we constrain each GOP of a given stream to span the same range of encoding rate. Unless otherwise stated, the Utility functions are given by the Rate-Distortion information extracted from the encoded sequences, as depicted in Figure 2. The distribution of the test video sequences between the different sender-receiver pairs is given in Table I for the scenarios considered in our simulations.

The proposed rate allocation algorithm is implemented in the application layer of the NS-2 simulator platform, and controls the rate of the underlying UDP transport protocol. The sender runs the control algorithm using the rate update equation (i.e., Eq. (12)) based on the utility functions of the transmitted stream and the feedback it gathers from the receiver. Unless otherwise stated, the maximum loss rate factor π is set to 0.05 in order to limit the degradations due to loss and the gain factor κ equals 1. Both values have been selected empirically in order to ensure good system performance. The sender then extracts a SVC substream at the target sending rate with help of the tools available in the JSVM software distribution and sends the appropriate packets to the receiver. Upon reception of each packet, the receiver checks for losses and eventually reconstructs a received packet trace for each GOP. It forms a feedback packet based on the information it has received. The video sequence is finally decoded with the packets that have been correctly transmitted.

$S_{1,2} \rightarrow R_{1,2}$	CREW, 4CIF @ 30Hz
$S_{3,4} \rightarrow R_{3,4}$	SOCCER, CIF @ 30Hz
$S_5 \rightarrow R_5$	SOCCER, QCIF @ 15Hz
$S_{6,7} \rightarrow R_{6,7}$	ICE, CIF @ 30Hz
$S_8 \rightarrow R_8$	ICE, CIF @ 30Hz (Priority 1)

TABLE I

DISTRIBUTION OF THE TEST VIDEO SEQUENCES AMONG THE SENDER-RECEIVER PAIRS.

B. Adaptation to changing bottleneck capacity

In the first set of simulations, we analyze the effect of a changing bottleneck capacity, or equivalently the effect of varying background traffic in the Scenario 1. We set the bottleneck bandwidth to 8 Mbps and let senders S_1, S_2, S_3 and S_4 start transmitting simultaneously their respective video streams at time $t = 0$. After 60 seconds, we add a 1 Mbps CBR background traffic stream in the same bottleneck channel, which translates into a sudden bottleneck capacity reduction for the four sender-receiver pairs. The background traffic is switched off again after one further minute.

The resulting sending rates are shown in Figure 8 that illustrates some key properties of the proposed algorithm. Clearly, the algorithm converges quickly to a stable state at the beginning of the transmission, as well as around the changes in background traffic. The stable state corresponds to maximization of the utilities of the different streams. Even though each stream is regulated independently of all the other ones, the same video sequences are allocated the same rate in the stable state. Once the CBR traffic joins the bottleneck, the streams react quickly and

Scenario 1	Mean Y-PSNR at sender	Mean loss at receiver
$S_1 \rightarrow R_1$	33.30 dB	0.65 dB
$S_2 \rightarrow R_2$	33.28 dB	0.75 dB
$S_3 \rightarrow R_3$	36.62 dB	0.52 dB
$S_4 \rightarrow R_4$	36.56 dB	0.49 dB
Scenario 2		
$S_5 \rightarrow R_5$	36.12 dB	0.05 dB
$S_6 \rightarrow R_6$	41.06 dB	0.82 dB
$S_7 \rightarrow R_7$	41.24 dB	0.80 dB
Scenario 3		
$S_5 \rightarrow R_5$	34.01 dB	0.11 dB
$S_6 \rightarrow R_6$	39.99 dB	0.53 dB
$S_8 \rightarrow R_8$	41.92 dB	0.91 dB

TABLE II

PSNR QUALITY FOR THE DIFFERENT STREAMING SCENARIOS [DB].

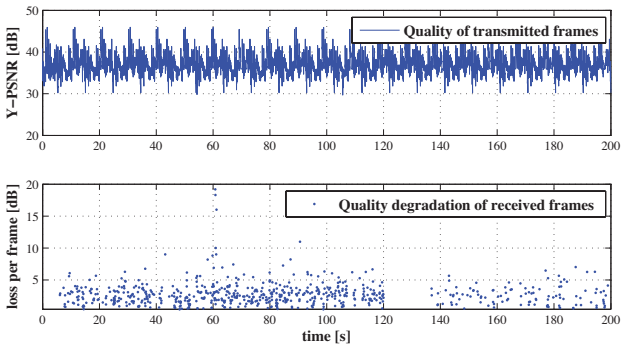


Fig. 9. Results for scenario 1. *Top:* Y-PSNR of each frame the SOCCER CIF@30Hz sequence transmitted by sender S_3. *Bottom:* Quality loss for each frame of the sequence, as received by client R_3.

settle in a new stable state almost immediately. Note that this behavior is very different from the sawtooth behavior seen in classical TCP connections, for example. As the CREW sequences carried by S_1 and S_2 have a lower utility gradient than the SOCCER sequences transmitted by S_3 and S_4, the rates of the latter are hardly affected by the drop in bottleneck capacity. In other words, as the background traffic lowers the bottleneck bandwidth, the distributed control system allocates less rate to the CREW streams, as this results only in a minor overall quality degradation. A rate reduction for the SOCCER streams would result in a larger quality drop. Once the background traffic is switched off, the four streams return rapidly to their original stable rates. The average quality of the streams sent by the senders are reported in Table II (Scenario 1) in terms of Y-PSNR, along with the respective quality reduction due to packet loss during the transmission.

Finally, we show the temporal evolution of the quality for one of the test streams in Figure 9. We report the Y-PSNR quality for each frame of the stream transmitted by S_3. We also compute the quality loss at the receiver by subtracting the Y-PSNR of each received frame from the Y-PSNR of the corresponding frame transmitted by the sender. Unsurprisingly, there is no loss when there is spare bandwidth available on the bottleneck link, i.e. during the first 10 seconds and after the CBR source switches off. There is however more packet loss due to congestion when the background traffic is active (i.e., in the timespan from 60 to 120 seconds). However, the steep gradient of the SOCCER Utility function prevents the rate from dropping. We note that loss cannot be completely avoided even in the steady state, as all the streams simultaneously compete for bandwidth shares until saturation of the bottleneck bandwidth. The small quality degradation resulting from these losses could be avoided by the use of error resilient coding methods.

C. Adaptation to new streams

In Scenario 2 we analyze the allocation of bandwidth resources when streams join and leave the bottleneck channel. We set the bottleneck bandwidth to 2 Mbps,

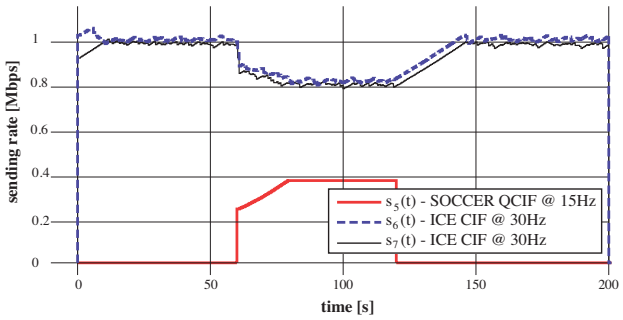


Fig. 10. Resulting sending rates/controls for scenario 2.

and the sources S_6 and S_7 both start transmitting at time $t = 0$. After 60 seconds, the sender S_5 starts streaming during one minute, and then leaves the bottleneck again. The sending rates for this dynamic join/leave scenario are depicted in Figure 10.

It can again be seen that the system converges rapidly into a stable state, where the two equivalent streams get an equal share of the bottleneck capacity. Once the third source initiates its streaming sessions, the sending rates of the ICE sequences are gracefully decreased for S_6 and S_7 in order to adapt to the new situation. The new stream that contains the SOCCER QCIF sequence has a steep utility function and is therefore aggressive in getting shares of the bottleneck bandwidth. The quality of the transmitted streams and the corresponding quality drops due to packet loss are given in Table II (Scenario 2).

In order to illustrate the benefit offered by a slight increase in the transmission rate, we have run the same simulation where the senders however do not implement the guard interval presented in the previous section (see Figure 6). The corresponding sending rates are given in Figure 11. It can be seen that in this case the algorithm has a slower convergence to the steady state due to the delay introduced by late feedbacks. This penalizes the average quality by about 1 dB per stream with respect to the same simulation scenario where the guard interval is used.

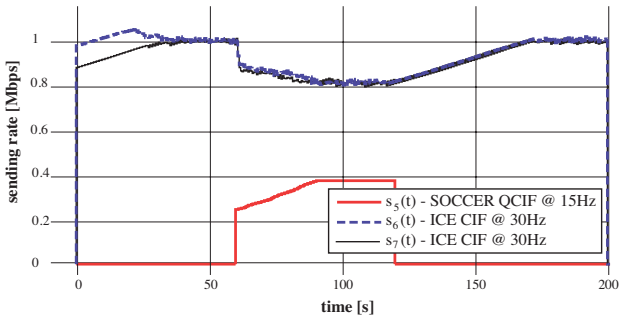


Fig. 11. Resulting sending rates/controls for scenario 2 when the guard interval scheme is not used.

D. Adaptation to priority classes

As stated earlier, the concept of Utility functions is very general and extends beyond the classical characterization

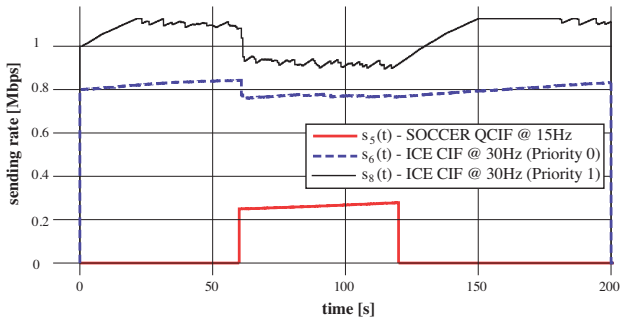


Fig. 12. Resulting sending rates/controls for Scenario 3. Both S_6 and S_8 transmit the same ICE sequence, but using different Utility curves.

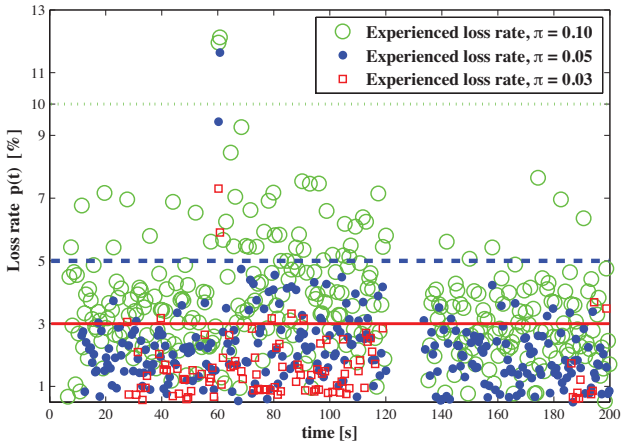


Fig. 13. Results for scenario 3: experienced loss rates $p(t)$, expressed in percentages, at receiver R_8 for different values of π

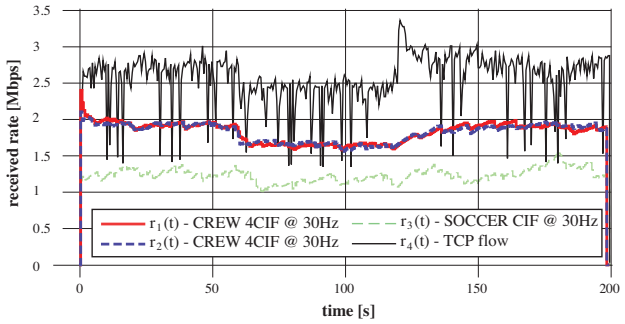


Fig. 14. Received rate in Scenario 1, where the stream from S_4 is replaced by a TCP-regulated flow.

of video streams by their rate-distortion characteristics only. To illustrate this, we consider in Scenario 3 a situation similar to the previous one, but where two streams from servers S_6 and S_8 correspond to the same video content with different priority classes. They are characterized by the 2 Utility curves depicted in Figure 3, which could for example model two clients with differential treatment. The resulting sending rates for this simulation run are shown in Figure 12. They illustrate that the choice of Utility function directly drives the performance of the streaming application, which clearly favor the high priority clients. We report again in Table II (Scenario 3) the average quality of the transmitted streams along with the quality drops due to packet loss. We see that the high priority stream benefits from a 2dB quality gain compared to the low priority stream with the same video content.

Finally, we have run the same scenario several times using different values of π in the rate update Equation (12). Remember that this factor scales all the Utility gradients in the system and should thus bound the loss rate experienced by any stream in the stable state. The result of this simulation is given in Figure 13 where we show the experienced loss (which is equal to the congestion signal $p(t)$), as seen by receiver R_8 for the three cases where π equals 0.03, 0.05 and 0.1 respectively. As expected the packet loss are bounded by π in each of these cases. The parameter π is therefore an essential tool in order to adapt the rate allocation algorithm to a given decoder. A decoder can be characterized by a loss rate that it can cope with while decoding a stream, mostly through the use of error-concealment techniques. By matching π to the maximum tolerable loss rate at the decoder, we therefore ensure that the received stream is decodable when the system is in the stable state.

E. Adaptation to TCP traffic

We finally illustrate the behavior of the rate allocation algorithm when the bottleneck channel is shared with a flow controlled by TCP. We have considered a scenario similar to Scenario 1, but we replace the 4th stream in the system by an FTP data flow, which is regulated by TCP (Tahoe implementation). The results of this experiment are shown in Figure 14, which presents the received rates at each of the clients. The goodput of the TCP flow is shown as stream 4. Interestingly, these results show that our proposed congestion control protocol can coexist with TCP on the same bottleneck channel. Compared to the Scenario 1, the sending rates of the media streams converge slower and show slightly larger oscillations around their stable state values. This is due to the rapid changes in available bandwidth, which is mostly driven by the TCP flow. However, one can observe that the rates are still smooth and that the CREW and SOCCER streams are still handled appropriately, according to their respective Utility functions.

While this simulation does not represent any formal proof of proper distribution of resources between TCP flows and the streams controlled by the algorithm proposed in this paper, it still shows that our algorithm does not starve TCP flows of the network resources. This corresponds to the results presented in [23], which state that the long term behavior of TCP Tahoe is equivalent to the one of a controller that maximizes a Utility function of the form $U(s) = \arctan(s)$. As this is a concave utility function, we expect TCP Tahoe streams to be able to compete with streams regulated by any other concave Utility functions in a stable system. However, the quality offered by the streams controlled by TCP is clearly sub-

optimal; TCP blindly optimizes the share of bandwidth resources, without considering the actual utility of the rate in terms of quality of experience.

Finally, we shall note that our algorithm is not *TCP-friendly* in the sense that the allocated rates yield an average per-flow bandwidth that is not equivalent to the one allocated by a TCP connection. TCP is in general more aggressive and tends to fairly share the average rate of each session, without considering the characteristics of each stream. Our algorithm targets a different objective, which is the effective allocation of the resources in order to maximize the aggregate utility, or equivalently to make the best use of bandwidth resources in terms of application requirements.

VII. Conclusions and Future Work

We have presented a distributed rate allocation algorithm that targets the optimal distribution of bandwidth resources for improving the average quality of service of media streaming applications. We have first extended the framework of Network Utility Maximization with an important stability proof, which states that the algorithm is stable under general concave utility functions and randomly delayed feedback. This result is particularly important for the implementation of rate allocation algorithms in real scenarios. We have then proposed an effective and scalable implementation of the distributed control algorithm with a light-weight application-layer protocol. We have further proposed a few practical utility functions for streaming scalable video sequences. Finally, we have analyzed the behavior of the proposed solution with extensive NS-2 simulations, where we have considered H.264 SVC encoded sequences as an illustration. We have shown that the bandwidth allocation actually respects the constraints imposed by the utility functions and that the system converges quite rapidly to a stable state after changes in the network. The proposed algorithm therefore provides an interesting solution for rate allocation in distributed streaming systems. We plan to study the extension of the proposed framework to step-like utility functions rather than concave functions, in order to provide efficient control solutions for a larger variety of streaming applications.

Acknowledgment

The authors would like to thank Dr Jacob Chakareski and Dr Jari Korhonen for fruitful discussions at the early stages of the work, and Dr Laurent Massoulié for feedbacks on the stability analysis for systems with heterogeneous delays.

References

[1] F. P. Kelly, "Charging and rate control for elastic traffic," *European Transactions on Telecommunications*, vol. 8, pp. 33–27, 1997.

[2] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, no. 3, pp. 237–252(16), Mar 1998.

[3] V. Jacobson, "Congestion avoidance protocol," *ACM SIGCOMM*, pp. 314–329, 1988.

[4] M. H. S. Floyd and J. Padhye, "Equation-based congestion control for unicast applications," *Proceedings of ACM SIGCOMM*, pp. 43–56, Sep 2000.

[5] J. Yan, K. Katrinis, M. May, and B. Plattner, "Media- and tcp-friendly congestion control for scalable video streams," *IEEE Transactions on Multimedia*, vol. 8, no. 2, pp. 196–206, Apr 2006.

[6] Z. Cao and E. W. Zegura, "Utility max-min: an application-oriented bandwidth allocation scheme," *IEEE INFOCOM*, vol. 2, pp. 793–901, Mar 1999.

[7] M. Chiang, S. Zhang, and P. Hande, "Distributed rate allocation for inelastic flows: Optimization frameworks, optimality conditions, and optimal algorithms," *IEEE INFOCOM*, vol. 4, pp. 2679–2690, Mar 2005.

[8] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Transactions on Networking*, vol. 8, no. 5, pp. 556–567, 2000.

[9] W. H. Wang and S. H. Low, "Application-oriented flow control: Fundamentals, algorithms and fairness," *IEEE/ACM Transactions on Networking*, vol. 14, no. 6, pp. 1282–1292, Dec 2006.

[10] Y. Zhang, S. R. Kang, and D. Loguinov, "Delayed stability and performance of distributed congestion control," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, pp. 307–318, Aug 2004.

[11] R. Johari and D. Tan, "End-to-end congestion control for the internet: Delays and stability," *IEEE/ACM Transactions on Networking*, vol. 9, no. 6, pp. 818–832, 2001.

[12] L. Massoulie and J. Roberts, "Bandwidth sharing: Objectives and algorithms," *IEEE INFOCOM*, vol. 3, pp. 1395–1403, Jun 1999.

[13] S. Kunniyur and R. Srikant, "End-to-end congestion control schemes: Utility functions, random losses and ecn marks," *IEEE INFOCOM*, vol. 3, pp. 1323–1332, 2000.

[14] P. Ranjan, R. J. La, and E. H. Abed, "Global stability conditions for rate control with arbitrary communication delays," *IEEE/ACM Transactions on Networking*, vol. 14, no. 1, pp. 94–107, Feb 2006.

[15] R. J. La and V. Anantharam, "Utility-based rate control in the internet for elastic traffic," *IEEE/ACM Transactions on Networking*, vol. 10, no. 2, pp. 272–286, Apr 2002.

[16] Y. Zhang and D. Loguinov, "Local and global stability of symmetric heterogeneously-delayed control systems," *IEEE Conference on Decision and Control (CDC)*, 2004.

[17] T. Alpcan and T. Basar, "A globally stable adaptive congestion control scheme for internet-style networks with delay," *IEEE/ACM Transactions on Networking*, vol. 13, no. 6, pp. 1261–1274, Dec 2005.

[18] J.-W. Lee, R. R. Mazumdar, and N. B. Shroff, "Non-convex optimization and rate control for multi-class services in the internet," *IEEE/ACM Transactions on Networking*, vol. 13, no. 4, pp. 827–840, Aug 2005.

[19] L. Ying, E. Dullerud, and R. Srikant, "Global stability of internet congestion controllers with heterogeneous delays," *American Control Conference*, vol. 4, pp. 2948–2953, Jun 2004.

[20] L. Massoulie, "Stability of distributed congestion control with heterogeneous feedback delays," *IEEE Transactions on Automatic Control*, vol. 47, no. 6, pp. 895–902, Jun 2002.

[21] Y. Zhang, S. R. Kang, and D. Loguinov, "Delay-independent stability and performance of distributed congestion control," *IEEE/ACM Transactions on Networking*, vol. 15, no. 6, Dec 2007.

[22] M. Dai, D. Loguinov, and H. M. Radha, "Rate-distortion analysis and quality control in scalable internet streaming," *IEEE Transactions on Multimedia*, vol. 8, no. 6, pp. 1135–1146, Dec 2006.

[23] J. He, M. Chiang, and J. Rexford, "Can congestion control and traffic engineering be at odds?" *Proceedings of the Global Telecommunications Conference. GLOBECOM '06*, pp. 1–6, 2006.

Jean-Paul Wagner received his M.Sc. in Communication Systems as well as his Ph.D. from the Ecole Polytechnique Fédérale de Lausanne (EPFL) in 2004 and 2008 respectively. His main research topics have been in the domain of distributed streaming of scalable multimedia content. He is now working for a firm of intellectual property attorneys in Luxembourg.

Pascal Frossard received the M.S. and Ph.D. degrees, both in electrical engineering, from the Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, in 1997 and 2000, respectively. Between 2001 and 2003, he was a member of the research staff at the IBM T. J. Watson Research Center, Yorktown Heights, NY, where he worked on media coding and streaming technologies. Since 2003, he has been a faculty at EPFL, where he heads the Signal Processing Laboratory (LTS4). His research interests include image representation and coding, visual information analysis, distributed image processing and communications, and media streaming systems.

Dr. Frossard has been the General Chair of IEEE ICME 2002 and Packet Video 2007. He has been the Technical Program Chair of EUSIPCO 2008, and a member of the organizing or technical program committees of numerous conferences. He has been an Associate Editor of the IEEE TRANSACTIONS ON MULTIMEDIA (2004-), the IEEE TRANSACTIONS ON IMAGE PROCESSING (2010-) and the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (2006-). He is an elected member of the IEEE Image and Multidimensional Signal Processing Technical Committee (2007-), the IEEE Visual Signal Processing and Communications Technical Committee (2006-), and the IEEE Multimedia Systems and Applications Technical Committee (2005-). He has served as Vice-Chair of the IEEE Multimedia Communications Technical Committee (2004-2006) and as a member of the IEEE Multimedia Signal Processing Technical Committee (2004-2007). He received the Swiss NSF Professorship Award in 2003, the IBM Faculty Award in 2005, the IBM Exploratory Stream Analytics Innovation Award in 2008 and the IEEE Transactions on Multimedia Best Paper Award in 2011.

A Framework for Video Network Coding with Multi-generation Mixing

M. Halloush
Yarmouk University/Department of Computer Engineering,
Irbid, Jordan
Email: mdhall@yu.edu.jo

H. Radha
Michigan State University/Department of Electrical and Computer Engineering,
East Lansing MI, USA
Email: radha@egr.msu.edu

Abstract—**Multi-Generation Mixing (MGM) is a generalized approach of Network Coding (NC) that has its improvements when applied in packet loss networks. In this paper we apply MGM network coding in networks communicating video contents. NC has shown viable improvements when applied in packet loss networks. With practical network coding (generation based network coding), packets are grouped in chunks called generations. Generation is the unit of network coding encoding and decoding. The generation grouping of packets is necessary for the practical deployment of network coding. On the other hand it increases the cost of NC losses. NC losses reduce the ability of receiver to decode packets, and hence can severely degrade the quality of recovered video. Video is encoded in layers with Scalable Video Coding (SVC) a base layer and one or more enhancement layers. MGM employs the layering of scalable video to enhance the reliability of communication. MGM provides different levels of reliable communication for the different video layers to improve the overall reliability of video communication. SVC enhancement layers are dependent on lower layers to be recovered. With MGM enhancement layers support the recovery of lower layers. This is done by network encoding lower video layers in higher layers. Through extensive simulations, we show that MGM highly improves the quality of recovered video**

***Index Terms— Scalable video, Network coding, Multi-generation mixing*..**

I. INTRODUCTION

Many video communication applications have emerged through the past few years. Improving the quality of video communicated over packet loss networks is a QoS requirement for a wide range of applications. There are many challenges in delivering high quality video in packet loss networks. Packet loss degrades the ability of receivers to decode video frames. Due to the dependency among video frames, the effect of packet loss can be severe. Packet loss can cause the loss of video frames and all video frames that are dependent on the lost frames.

Network coding [2] has shown promising improvements when applied in packet loss networks [1,3-5,14]. The improvements of NC are in terms of enhanced robustness and bandwidth utilization [6]. Multi-Generation Mixing (MGM) has been proposed as a generalized approach for practical network coding [7, 8]. MGM improves the performance of practical network coding by allowing the mixing among sender packets in a way that improves network coding decodable rates. The improvements achieved by MGM network coding make it a viable approach to improve the quality of video communicated over packet loss networks.

H.264 Scalable Video Coding (SVC) is the scalable extension of the H.264 Advanced Video Coding (AVC). H.264 SVC has been standardized by the Joint Video Team (JVT) [9]. With SVC video is encoded in layers; a base layer and one or more enhancement layers. This allows the decoding of video in different temporal rates, picture sizes, and fidelity. SVC is an attractive solution to many of the challenges faced in video communication systems. With SVC there is no need to encode the video more than once to meet the different requirements of receivers in a heterogeneous environment [10].

Lower SVC video layers are referenced by higher layers which makes the frames of lower video layers necessary for the decoding of higher video layers. Prioritizing the transmission of lower video layers without sacrificing the reliability of communicating higher layers improves video decodable rates. Providing different levels of reliable communication to the different video layers is the goal of applying multi-generation mixing in networks communicating video contents.

With practical network coding, sender packets are grouped in generations. Network encoding/decoding is performed on generations separately. Grouping packets in generations makes a generation the minimum unit of network coding data recovery. In other words if insufficient number of encoded packets were received of a generation the whole generation is lost [11]. From here we can see that depending on the generation size network coding losses can be expensive. Network coding with multi-generation mixing has been proposed to improve NC decodable rates by allowing the cooperative encoding and decoding among generations.

doi:10.4304/jcm.7.3.192-201

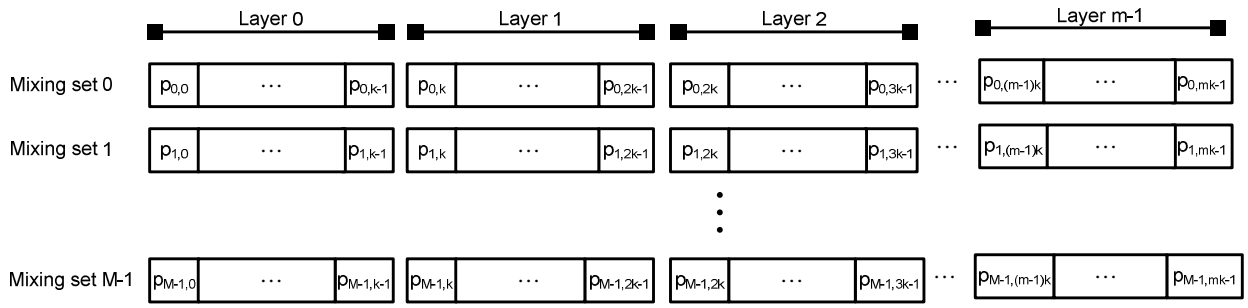


Figure 1: Data partitioning with multi-generation mixing into different layers of priority. Mixing set size is *m*, generation size is *k*.

Multi-generation mixing has been proposed as a generalized approach for practical network coding. With Multi-generation mixing generations are grouped in mixing sets where encoding/decoding is performed among mixing set generations in a particular way such that multiple decoding options at receiver are supported for each generation. Multi-generation mixing supports different levels of reliable communication for the different mixing set generations. In other words the different mixing set generations are communicated with varying degrees of protection against losses. This feature makes MGM suitable for applications where priority transmission is recommended for improved performance.

A special case of MGM is generation based network coding [11] (as we will see shortly). With traditional generation based network coding all sender packets are communicated with the same level of reliability. In other words no layered communication is supported.

In this paper network coding with multi-generation mixing is deployed in networks communicating video contents. The communication of scalable as well as non-scalable video is evaluated under multi-generation mixing. The goal is to show how video communication can benefit from network coding. The rest of the paper is organized as follows. In Section II an overview of network coding with multi-generation mixing is provided. In Section III the unequal protection characteristic of multi-generation mixing is analyzed and evaluated. In Section IV an overview of scalable video structure is presented. In Section V MGM deployment for scalable and non-scalable video is discussed. In Section VI the performance of MGM video network coding is evaluated. Finally, In Section VII the paper is concluded.

II. MULTI-GENERATION MIXING

With MGM, generations are grouped in mixing sets where the size of a mixing set is m generations. Each generation in the mixing set has a position index *l*, where $0 \le l < m$. The position index of a generation indicates its relative position within the mixing set. Figure 1 illustrates the structure of the J^{th} mixing set. In this section we give a brief overview of multi-generation mixing encoding and decoding. Detailed discussion is in [1].

a) MGM Encoding

With traditional generation based network coding, encoding is performed on packets of the same generation. For a generation of size k at least k independent encodings are generated and transmitted.

With MGM, encoding is performed on packets that may belong to different generations in the mixing set. For a node to send a packet of a generation with position index l, that node encodes all the packets it has that are associated with all generations in the same mixing set that have position indices less than or equal to l.

b) MGM decoding

With Generation based network coding, generations are decoded separately. When sufficient number of useful (independent) packets associated with a generation is received, that generation is decodable. On the other hand if insufficient number of independent encodings is received, the generation is un-decodable (lost).

With MGM network coding, packets associated with generation of position index l where $0 \le l < m$ can be successfully decoded in one of two possible scenarios. The first scenario is *Incremental Decoding*. In this scenario all generations with position indices less than l' (i.e., for $0 \le l' < l$) in the same mixing set were decoded and at least k independent MGM packets associated with generation of position index l have been received.

The second scenario is *Collective Decoding*, when incremental decoding is not possible due to receiving insufficient number of independent encodings for one or more mixing set generations. In this case generations are decoded collectively in subsets of mixing set generations. With collective decoding no more than $(l'+1) \cdot k$ independent packets associated with generation of position index l' where $0 \le l' \le l$ can be used in decoding the subset of generations that starts from the first generation with position index zero to the generation with position index l where $0 \le l < m$. This is necessary for successfully formulating a decoding matrix of full rank ($(l+1) \cdot k$) [1, 7].

With MGM each generation supports the recovery of all previous generations in the mixing set. This means that the lower the generation position index the more mixing

Table 1: Guaranteed delivery conditions for generations of size k within mixing sets of sizes m=1, 2, 3 and the corresponding probabilities of successful delivery. For each generation at least k independent packets are sent in addition to extra independent packets to protect against packet loss. The extra packets are independent encodings associated with the transmitted generation.

Mixing set size	**Generation position index**	**Condition for generation recovery**	**Probability of generation successful recovery**
$m=1$	g_0	k_0^+	$p(r_0 \geq k)$
$m=2$	g_0	k_0^+	$p(r_0 \geq k)$
		$k_0^-,(k_0+k_1)^+$	$p(r_0 < k)\cdot p(r_0+r_1 \geq 2\cdot k)$
	g_1	k_0^+,k_1^+	$p(r_0 \geq k)\cdot p(r_1 \geq k)$
		$k_0^-,(k_0+k_1)^+$	$p(r_0 < k)\cdot p(r_0+r_1 \geq 2\cdot k)$
$m=3$	g_0	k_0^+	$p(r_0 \geq k)$
		$k_0^-,k_1^+,(k_0+k_1)^+$	$p(r_0 < k)\cdot p(r_1 \geq k)\cdot p(r_0+r_1 \geq 2\cdot k)$
		$k_0^-,k_1^+,(k_0+k_1)^-,(k_0+k_1+k_2)^+$	$p(r_0 < k)\cdot p(r_1 \geq k)\cdot p(r_0+r_1 < 2\cdot k)$ $\cdot p(r_0+r_1+r_2 \geq 3\cdot k)$
		$k_0^-,k_1^-,(k_0+k_1+k_2)^+$	$p(r_0 < k)\cdot p(r_1 < k)\cdot p(r_0+r_1+r_2 \geq 3\cdot k)$
	g_1	k_0^+,k_1^+	$p(r_0 \geq k)\cdot p(r_1 \geq k)$
		$k_0^-,k_1^+,(k_0+k_1)^+$	$p(r_0 < k)\cdot p(r_1 \geq k)\cdot p(r_0+r_1 \geq 2\cdot k)$
		$k_0^-,k_1^+,(k_0+k_1)^-,(k_0+k_1+k_2)^+$	$p(r_0 < k)\cdot p(r_1 \geq k)\cdot p(r_0+r_1 < 2\cdot k)$ $\cdot p(r_0+r_1+r_2 \geq 3\cdot k)$
		$k_0^-,k_1^-,(k_0+k_1+k_2)^+$	$p(r_0 < k)\cdot p(r_1 < k)\cdot p(r_0+r_1+r_2 \geq 3.k)$
		$k_0^+,k_1^-,(k_0+k_1+k_2)^+,(k_1+k_2)^+$	$p(r_0 \geq k)\cdot p(r_1 < k)\cdot p(r_0+r_1+r_2 \geq 3\cdot k)$ $\cdot p(r_1+r_2 \geq 2\cdot k)$
	g_2	k_0^+,k_1^+,k_2^+	$p(r_0 \geq k)\cdot p(r_1 \geq k)\cdot p(r_2 \geq k)$
		$k_0^-,k_1^-,(k_0+k_1+k_2)^+$	$p(r_0 < k)\cdot p(r_1 < k)\cdot p(r_0+r_1+r_2 \geq 3\cdot k)$
		$k_0^-,k_1^+,(k_0+k_1)^+,k_2^+$	$p(r_0 < k)\cdot p(r_1 \geq k)\cdot p(r_0+r_1 \geq 2k)\cdot p(r_2 \geq k)$
		$k_0^-,k_1^+,(k_0+k_1)^-,(k_0+k_1+k_2)^+$	$p(r_0 < k)\cdot p(r_1 \geq k)\cdot p(r_0+r_1 < 2\cdot k)$ $\cdot p(r_0+r_1+r_2 \geq 3\cdot k)$
		$k_0^+,k_1^-,(k_1+k_2)^+$	$p(r_0 \geq k)\cdot p(r_1 < k)\cdot p(r_0+r_2 \geq 2\cdot k)$

set encoded packets transmitted that carry information from that generation. From here we can see that MGM supports the unequal protection of mixing set generations depending on the generation position within the mixing set. Also we can see that generation based network coding is a special case of MGM when mixing set size is one (m=1). In this case there is only one generation in the mixing set and hence no inter-generation mixing is supported [11].

MGM network coding improves the performance of practical network coding. This comes on the cost of increased network coding computational overhead. The computational overhead of MGM is investigated in [1].

III. MGM UNEQUAL PROTECTION

MGM allows inter-generation mixing within a mixing set to improve generation's decodable rates. An encoded packet associated with a generation carries information from its generation as well as all generations that have lower position indices in the mixing set. The lower generation position index the larger number of encoded packets transmitted that carry that generation information and hence the higher reliability of generation delivery [1]. Consequently, generation position index can be considered as a priority level for that generation. More specifically each generation in the mixing set has a priority level such that generations with lower position indices have higher levels of protection against losses

than succeeding generations (with higher position indices) in the same mixing set. Among the different mixing sets, generations with the same position index are in the same priority level. Figure 1 illustrates the different priority levels of generations in different mixing sets.

In Figure 1 the first priority level consists of generations with position index zero in consecutive mixing sets. The second priority level consists of generations with position index one in consecutive mixing sets and so on. From here we can see that the number of priority levels equals the size of the mixing set.

For different mixing set sizes and for each generation within the mixing set, Table 1 summarizes the different conditions of recovering (decoding) sender generations and the probability of having each condition satisfied. In Table 1, $(k_0 + \cdots + k_l)^+$ indicates that at least $(l+1)\cdot k$ independent packets of the $(l+1)$ generations have been received. On the other hand $(k_0 + \cdots + k_l)^-$ indicates that less than $(l+1)\cdot k$ independent packets of the $(l+1)$ generations have been received. A generation in the table is recovered if any of its recovery conditions is satisfied. The entries in Table 1 satisfy the incremental/collective MGM decoding conditions explained previously.

In Table 1, for each generation g_i in a mixing set of size m there is at least one entry that shows the condition of delivery of that generation. For example, for mixing set size two (m=2) and for the generation with position index zero (g_0), k_0^+ is the condition of delivery of that generation. k_0^+ indicates that at least k independent packets associated with generation of position index zero (g_0) necessary to recover that generation have been received. If this condition is not satisfied, it is still possible to recover g_0 if the second condition $(k_0^-, (k_0 + k_1)^+)$ is satisfied. $k_0^-, (k_0 + k_1)^+$ indicates that less than k independent packets associated with generation g_0 were received (first condition is not satisfied) and the overall number of independent packets received of the two mixing set generations (g_0 and g_1) is greater than $2k$ and hence the two generations are collectively decodable. This is possible since more than k encoded packets associated with each generation in the mixing set can be sent.

The fourth column in the table is the probability of the corresponding guaranteed recovery condition. In the fourth column r_l is the number of independent packets received that are associated with generation of position index l (g_l). For the same previous example (m=2 and g_0), $p(r_0 \geq k)$ is the probability that the receiver receives at least k independent packets of generation with position index zero (g_0). With losses characterized by binomial distribution, let p be the probability packet loss. The probability of successful delivery of the first generation is:

$$P_s = P_{k_0^+} \tag{1}$$

$$P_s = \sum_{j=k}^{n} \binom{n}{j} \cdot (1-p)^j \cdot p^{n-j} \tag{2}$$

For a generation of size k, the sender transmits $n = k + (R \cdot k)$ packets, where R is the portion of extra independent packets that are used to protect the generation against losses.

For the second guaranteed recovery condition, $p(r_0 < k) \cdot p(r_0 + r_1 \geq 2 \cdot k)$ is the probability of receiving less than k independent packets of the first generation multiplied by the probability of receiving an overall number of independent packets of the first two generations (g_0, g_1) that is at least $2k$, which is sufficient for the collective recovery of the two generations. In this case:

$$P_s = P_{k_0^-(k_0+k_1)^+} \tag{3}$$

$$P_s = (1 - \sum_{j=k}^{n} \binom{n}{j} \cdot (1-p)^j \cdot p^{n-j}) \cdot (\sum_{j=2k}^{2n} \binom{2 \cdot n}{j} \cdot (1-p)^j \cdot p^{n-j}) \tag{4}$$

Figure 2 shows how the different generations (g_0, g_1) within a mixing set of size two (m=2) differs by the probability of successful delivery. The first generation (g_0) has a higher probability of successful delivery; since it is protected by the succeeding generation that has higher position index (g_1).The second generation (g_1) is the last generation in the mixing set.

Figure 3 shows how the different generations (g_0, g_1, g_2) within a mixing set of size three (m=3) differs by the probability of successful delivery. The first generation (g_0) has the highest probability of successful delivery; since it is protected by the two succeeding generations that have higher position indices (g_1, g_2).The second generation (g_1) has higher probability of successful delivery than the third generation (g_2) (g_1 is protected by g_2 which is the last generation in the mixing set).

Figure 5 compares the average probability of generation successful delivery for mixing sets of sizes one, two and three (m=1, 2, 3). For a mixing set of size m, the probability of successful delivery is averaged over all mixing set generations. This figure shows that generally, by increasing the size of a mixing set MGM achieves an improvement in the probability of successful delivery.

Figure 8 compares the probability of successful delivery for the last generations of mixing sets of sizes one, two and three (m=1, 2, 3). All these generations are last generations in there mixing sets; they are not protected by any other generations. These generations achieve close probabilities of successful delivery. This means that although MGM enhances the reliability of delivering generations of lower position indices, it does not degrade the reliability of delivering generations of higher position indices. For mixing sets of sizes two and three (m=2, 3) the last generation in these mixing sets achieves a decodable rate that is close to the case of the single

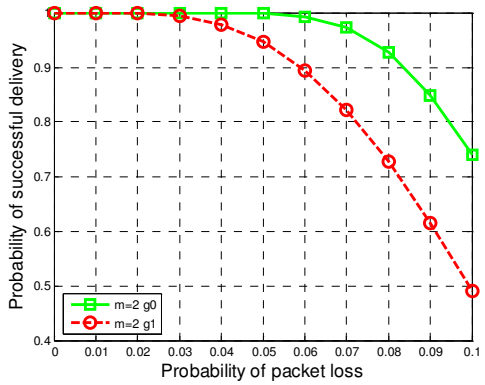


Figure 2: Probability of successful delivery for the two generations of a mixing set of size two (m=2). k=50, R=0.1.

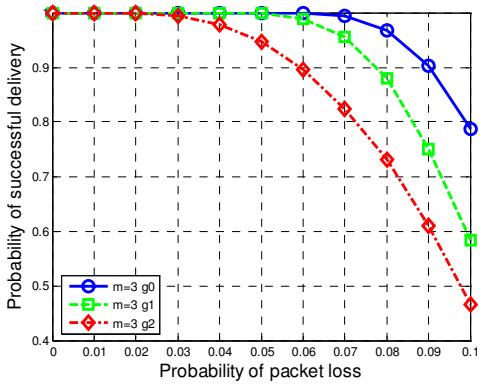


Figure 3: Probability of successful delivery for the three generations of a mixing set of size three (m=3). k=50, R=0.1.

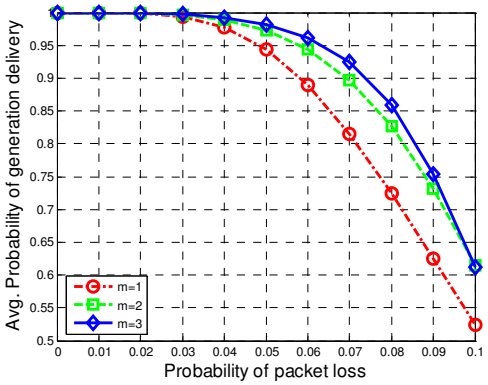


Figure 4: Average probability of successful delivery of generations in mixing sets of different sizes (m=1, 2, 3). k=50, R=0.1.

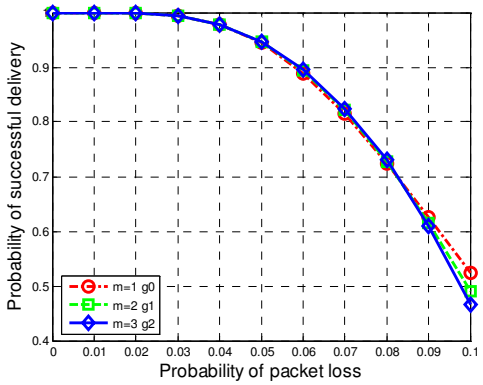


Figure 5: Probability of successful delivery for the last generations in mixing sets with sizes (m=1, 2, 3). k=50, R=0.1.

generation of mixing set of size one (m=1), which is the case of generation based network coding (no multi-generation mixing).

IV. SVC STRUCTURE

By using scalable video coding, video is encoded in layers a base layer and one or more enhancement layers. Video layers support the decoding of video in varying reconstruction quality. Higher video layers enhance the video of base layer. Enhancement is in terms of temporal resolution, spatial resolution, and fidelity.

Scalable video stream consists of sub-streams that allow the decoding of video with a reconstruction quality that depends on the decoded sub-streams. From received sub-streams video can be recovered in different frame rates (temporal scalability), picture size (spatial scalability), and quality (fidelity). Supporting video recovery from received sub-streams has many benefits in a video communication system, this appears in a heterogeneous environment of a multicast scenario where a set of receivers request the same video content with different frame rate, size and quality. With SVC video is encoded once with sub-streams from which the receivers can decode video with requested quality.

With SVC video there is dependency among coded video frames. Intra and inter layer video frames dependencies are among frames of the same layer or different layers respectively. The loss of a video frame can cause the inability to decode other video frames in the same layer or among the different layers.

With SVC the coded video stream is organized in NAL (Network Abstraction Layers) units. A NAL unit consists of a header and a payload. The header is one byte that specifies the type of payload of the NAL unit. Each NAL unit is identified by a tuple (D, T, Q). (D, T, Q) tuple specifies the Dependency ID (D), Temporal ID (T) and Quality ID (Q) of the NAL unit [10, 12].

NAL units identified by (0, *, 0) constitute the base layer of the video stream. * represents all possible levels of temporal resolution. NAL units of the base layer are the most important since they are referenced by NAL units of other video layers and they don't reference any other layers. Figure 6 shows the dependency among NAL units of the different video layers. Spatial, temporal and SNR resolutions increase in the arrow direction.

The loss of one NAL unit will cause the inability to

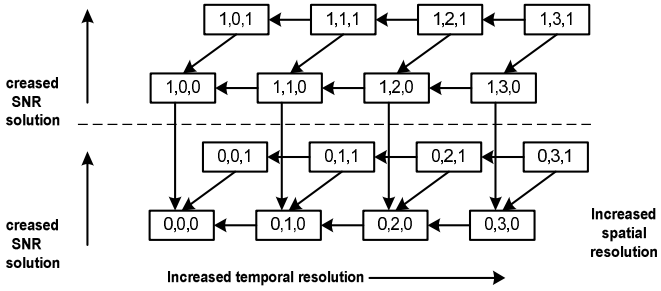


Figure 6: NAL units dependency among different video layers.

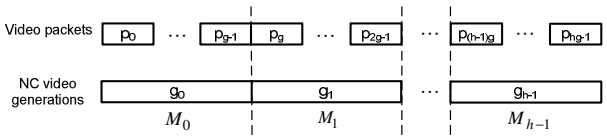


Figure 7: Non-scalable video packets grouped in generations for network encoding. Mixing set size is one (m=1), this is the case of generation based network coding.

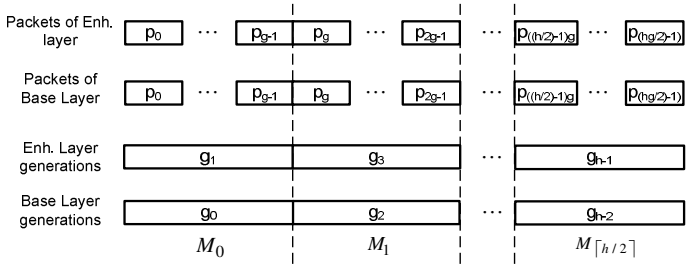


Figure 8: Scalable video of two layers. Packets of each layer are grouped in generations that have the same position index in consecutive mixing sets. Mixing set size is two (m=2).

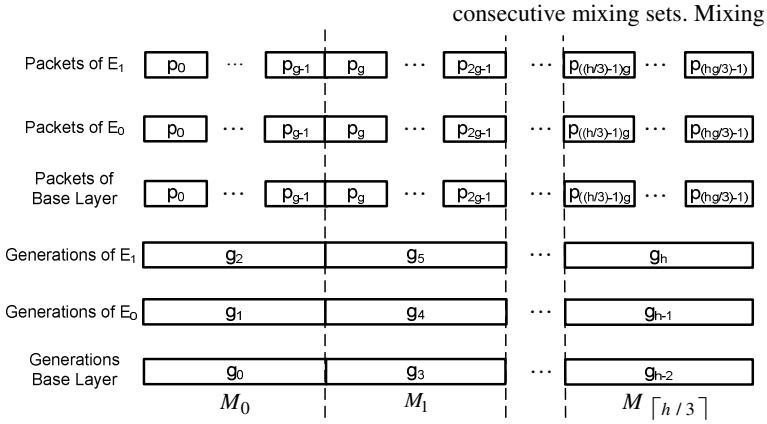


Figure 9: Scalable video of three layers. Packets of one layer each grouped in generations that have the same position index in consecutive mixing sets. Mixing set size is one (m=3).

decode other NAL units. For example a NAL unit with (D, T, Q) = (1, 1, 0) cannot be decoded due to its dependency on NAL unit (D, T, Q) = (0, 1, 0) that was lost or not decoded. This means a NAL unit that is either lost or not decoded (due to its dependency on a non-decodable NAL unit) will spread to have more NAL units un-decodable.

Due to the importance of lower layers to decode higher video layers, it will be of high advantage to prioritize the communication in a way that provides higher levels of reliable communication to lower video layers. This is the goal of applying MGM on scalable video; enhancing the reliability of communicating lower video layers without sacrificing the reliability of communicating higher layers. Next, we will discuss how to apply MGM network coding on video contents.

V. MGM FOR SCALABLE AND NONSCALABLE VIDEO

In packet loss networks, video is communicated in packets. NAL units are packettized in packets of limitted size. To apply practical network coding on video, video packets are grouped in generations. Hence a generation is a consecutive sequence of video packets. To apply multi-generation mixing, video generations are mapped to mixing sets. Hence a mixing set is a consecutive sequence of generations. Multi-generation mixing is applied by mapping video generations to mixing sets of size equals the number of video layers. We will focus on three scenarios. The first is non-scalable video with generation based network coding which is MGM with mixing set size one (m=1). The second scenario is scalable video of two layers where MGM has mixing set size two (m=2). The third scenario is scalable video of three layers where MGM has mixing set size three (m=3). In Figures 7, 8 and 9, g_x is the generation with index x. P_y is the packet with index y.

For the scenario of non-scalable video, video packets are grouped in generations that are mapped to mixing sets where the size of mixing set is one (m=1). In this case there is no inter-generation mixing and hence this is the case of traditional generation based network coding. Figure 7 shows the grouping of non-scalable video packets in generations that constitute mixing sets of size one.

On the other hand, for the scenario of scalable video of two layers, video generations are mapped to mixing sets of size two (m=2). The first generation in consecutive mixing sets (generations with position index zero) consist of packets of video base layer. Second generation in consecutive mixing sets (generations with position index one) consist of packets of first enhancement video layer. In this case packets of base layer (first generations in consecutive mixing sets) are provided with higher level of reliable communication than packets of enhancement layer (second generations in consecutive mixing sets). This is because of encoding first mixing set generation in the second which allows the collective decoding of first mixing set generations as was explained previously. Figure 8 shows the grouping of video packets in generations that constiute mixing sets of size two.

For the case of scalable video of three layers, video generations are mapped to mixing sets of size three (m=3). Generations with position index zero in consecutive mixing sets consist of video packets of base

layer. Generations with position index one in consecutive mixing sets consist of video packets of first enhancement layer. Generations with position index two in consecutive mixing sets consist of video packets of second enhancement layer. Figure 9 shows the grouping of video packets in generations that constiute mixing sets of size three.

Next, we evaluate video network coding using extensive simulations.

VI. PERFORMANCE EVALUATION

In this Section we evaluate the performance of practical network coding in video networks and with extensive simulations. Foreman video sequence of 300 video frames is encoded in one, two and three CGS layers so that a particular bit rate is targeted for each layer. In the scenario of single layer video (non-scalable) the target bit rate is 300Kbps. For the scenario of video of two layers, each layer is encoded with a target bit rate of 150 Kbps. And finally for the scenario of three video layers each layer is encoded with a target bit rate of 100 Kbps. Each layer is encoded with five temporal levels with 1.875, 3.75, 7.7, 15 and 30 fps Table 2 summarizes the stream layers bit rates and overall PSNR. Video sequence is encoded, decoded packetized and evaluated using JSVM 9.4 [13]. The packetized video streams are communicated using a round based network simulator that was developed for the purpose of evaluating practical network coding (Multi-generation mixing and traditional generation based network coding which is a special case of MGM).

For the network topology, 400 nodes are distributed randomly in an area of 20×20. In each 1×1 unit area there is a randomly positioned node that can communicate directly with all nodes within a radius of 1.5. The network is of broadcast nature. At each round of simulation, only one node transmits within the transmission radius. Receiving nodes in a particular round are candidate senders in the following rounds. senders in the following round are selected so that there is no collision among their transmission. Source is a node at one corner of the topology and receiver is selected randomly to be close to the other corner. Each node transmits an encoded packet for every useful encoded packet received.

A large number of simulations were done where performance was evaluated using different generation sizes. The quality of video at the selected receiver is evaluated at the same time we evaluated network coding decodable rates achieved by all nodes to show the improvement of achieving efficient spread of useful data across communicating nodes.

In Figures 10, 11, and 12, m=1 is the case of single layer video and generation based network coding. m=2 is the case of scalable video of two layers and multi-generation mixing with mixing set size two. m=3 is the case of scalable video of three layers and multi-generation mixing with mixing set size three.

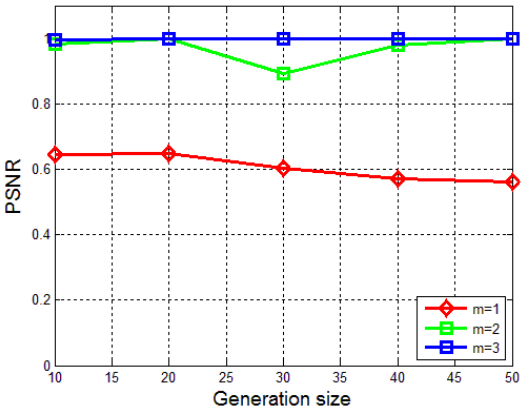


Figure 10: PSNR of video communicated under network coding, m is mixing set size. m=1 is the case of generation based NC for non-scalable video. m=2 is MGM for scalable video of two layers. m=3 is MGM for scalable video of three layers.

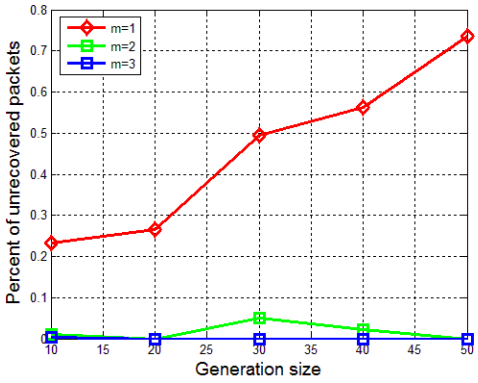


Figure 11: Percent of unrecovered packets at receiver. , m is mixing set size. m=1 is the case of generation based NC for non-scalable video. m=2 is MGM for scalable video of two layers. m=3 is MGM for scalable video of three layers.

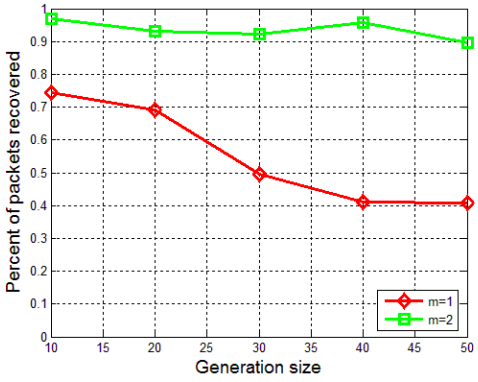


Figure 12: Ratio of network coding decodable packets received with generation based, and MGM m=2 to that received with MGM m=3, averaged over all nodes. This shows the overall network performance in decoding packets in comparison with MGM, m=3.

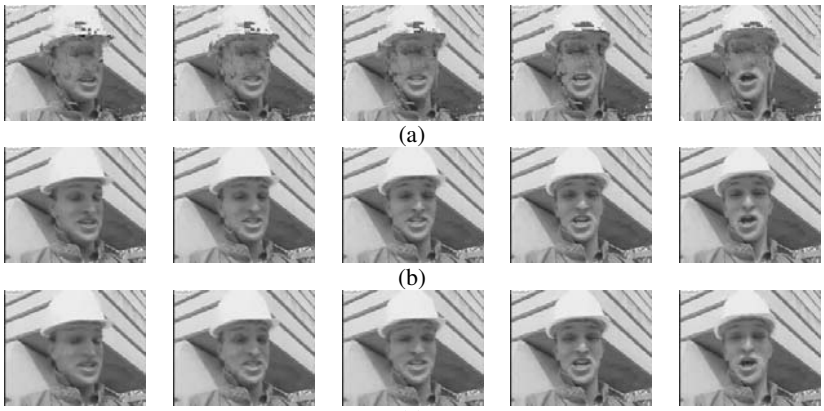


Figure 13 :Five consecutive frames (78-82) of Foreman video sequence. (a) Generation based NC (MGM with m=1) applied on video sequence of single layer. (b) MGM with m=2 applied on video sequence of two layers. (c) MGM with m=3 applied on video sequence of three layers. Generation size k=10.

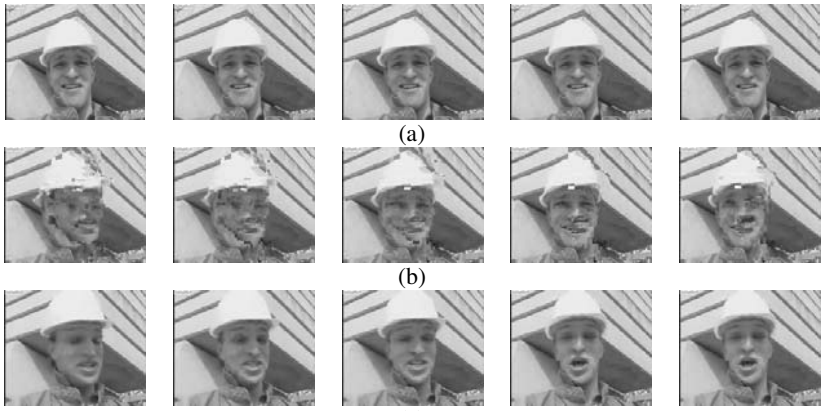


Figure 14: Five consecutive frames (78-82) of Foreman video sequence. (a) Generation based NC (MGM with m=1) applied on video sequence of single layer. (b) MGM with m=2 applied on video sequence of two layers. (c) MGM with m=3 applied on video sequence of three layers. Generation size k=30.

PSNR values in Figure 10 are relative to those in Table 2. More precisely, for single layer video, PSNR values of PSNR in Figure 10 are relative to the maximum PSNR value of 34.5 (PSNR for single layer video). For scalable video of two layers, PSNR values in the figure are relative to the maximum PSNR value of 32.9 (PSNR for video of two layers). For scalable video of three layers, PSNR values in the figure are relative to the maximum PSNR value of 30.7 (PSNR for video of three layers). The goal is to show the ability of network coding in supporting the recovery of maximum video quality at receiver node for each of the three scenarios.

For the selected receiver Figure 10 shows the PSNR of recovered video for the three scenarios of Figures 7, 8, and 9. When MGM is applied to scalable video of three layers (m=3) almost full video quality is achieved at receiver. It is higher than scalable video of two layers. For the scenario of non-scalable video we notice a poor quality of recovered video. This is due to the expensive losses of generation based network coding (explained previously).

Table 2: Bit rates and Y-PSNR for 300 video frames, SVC Foreman sequence.

Number of Video Layers	Bit Rate (Kbps)			Y-PSNR
	Layer1	Layer 2	Layer 3	
Single layer	305.3	-	-	34.5
Two Layers	145.96	301.07	-	32.9
Three Layers	95.59	198.28	295.91	30.7

Figure 11 shows the percent of unrecovered packets at receiver for the three scenarios. Results shown in Figure 11 lead to the PSNR values shown in Figure 10, and follow the same justification. As seen in Figure 11 the cases of m=3 and m=2 looks similar to the cases of m=3 and m=2 of Figure 10.

The case of m=1 in Figure 11 differs from that in Figure 10. In Figure 11 and for m=1, the increase in the percent of unrecovered packets is higher than the resulted decrease in PSNR shown in Figure 10. The reason for that is the deployment of video concealment technique upon the loss of video frames. Each lost frame is replaced by the last frame decoded correctly. This leads to the lowered effect of frame loss on the PSNR values of recovered video.

In Figure 12 the percent of decoded packets of m=1 and m=2 relative to m=3 is evaluated. The percent of decoded packets is evaluated over all topology nodes for different

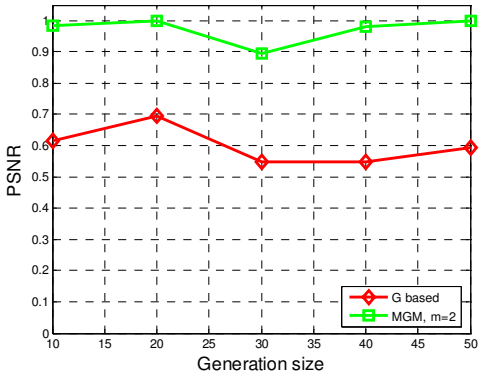


Figure 15: PSNR with multi-generation mixing and generation based for Foreman SVC sequence of two layers.

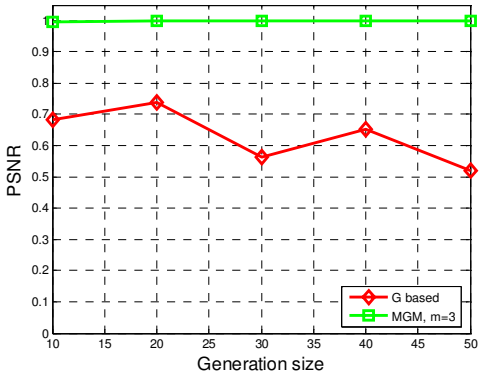


Figure 16: PSNR with multi-generation mixing and generation based for Foreman SVC sequence of three layers.

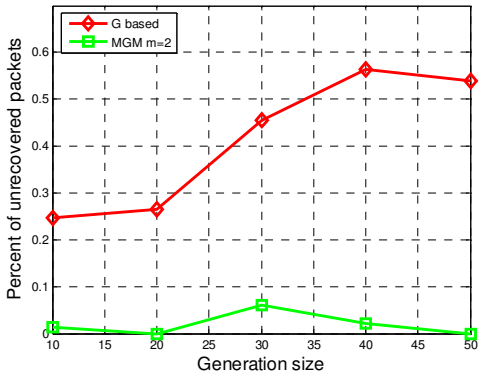


Figure 17: Percent of unrecovered packets at the selected receiver. With MGM m=2 an improved decodable rate is achieved.

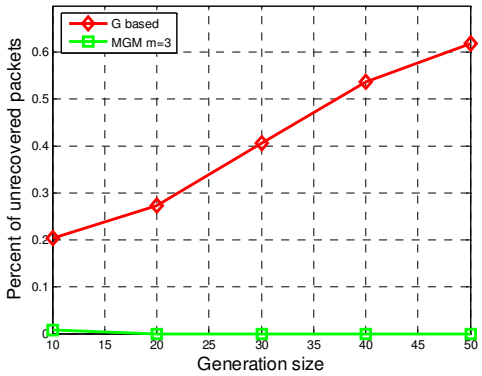


Figure 18: Percent of unrecovered packets at the selected receiver. With MGM m=3 almost full decodable rate is achieved.

generation sizes. This gives a clear idea about the improvements achieved by multi-generation mixing in increasing the spread of useful packets among network nodes. As shown in the figure for MGM with m=2, over 90% of MGM, m=3 decodable rate is achieved. On the other hand with Generation based network coding (m=1) the decodable rate is between 40% - 75% of that for MGM with m=3.

In Figures 13 and 14 the visual effect of loss on the recovered frames of Foreman video sequence is shown. In both figures (a) is for generation based network coding applied on video encoded in a single layer, (b) is for MGM with m=2 applied on video encoded in two layers, and (c) is for MGM with m=3 applied on video encoded in three layers.

In Figure 13 losses incur distortion on the recovered pictures (frames of a). On the other hand we notice correct recovery of the pictures when MGM is applied on two and three layerd video (frames of b and c).

In some scenarios MGM with m=2 losses degrade the quality of recovered video (as shown in Figures (10, 11, and 12). Figure 14 shows a scenario in which no frames are decoded with generation based network coding (frames of a). The five consecutive frames of (a) are copies of the last correctly decoded frame. Substiuting the last correctly received frame in the place of lost frames is the loss concealment approach used. At the same time as shown in (b) losses incur distortion on the recoverd frames when MGM with m=2 is applied. On the other hand (c) shows the correct decoding of the five consecutive frames when MGM with m=3 is applied.

Now we apply generation based network coding to scalable video and evaluate the quality of recovered video. More specifically, for video of two layers we apply generation based network coding (G-based) as well as MGM with mixing set size two (MGM, m=2). For video of three layers we apply generation based network coding (G-based) as well as MGM with mixing set size three (MGM, m=3). The goal here is to show that the improvement in video quality is mainly due to the deployment of MGM.

Generation based network coding is applied on video encoded in two and three layers in Figures 15-18. Video packets are grouped into generations without distinguishing between the different video layers. In other words all video packets have the same level of reliable communication supported by generation based network coding.

In Figures 15 and 16 MGM achieves major improvements in the quality of recovered video at receiver. In Figures 15 and 16 the improvement of MGM with m=2 (Figure 15) and m=3 (Figure 16) is over generation based network coding (G-based) applied on scalable video of two layers (Figure 15) and three layers (Figure 16). These improvements are achieved over the different generation sizes (generation size from 10 to 50). We note that MGM with m=3 (Figure 16) achieves almost full video quality at receiver. On the other hand

for MGM with *m*=2 (Figure 15) the quality of recovered video is degraded in some scenarios (generation size 30). This indicates that by increasing the size of mixing set reliable video delivery is assured. Hence the larger the mixing set the more reliable video communication.
Figures 17 and 18 show the percent of unrecovered packets at receiver. With generation based NC (G-based), as the generation size increases there is an increase in the percent of unrecovered packets and hence a degradation in the quality of recovered video (as shown in Figures 15 and 16). The reason behind this is the increased cost of network coding losses as the generation size increases. On the other hand with MGM the percent of unrecovered packets is very low due to the supported inter-generation mixing.

VII. CONCLUSION

In this paper network coding with multi-generation mixing was applied and evaluated in networks communicating video contents. Results showed major improvements in the quality of recovered video when multi-generation mixing is applied. The improvments in video quality is achieved due to the improved network coding decodable rates of multi-generation mixing.

References

[1] M. Halloush, and H. Radha, "Network Coding with Multi-generation Mixing: A Generalized Framework for Practical Network Coding," *IEEE Transactions on Wireless Communications,* vol. 10, pp. 466 - 473, December, 2010.

[2] R Ahlswede, N Cai, S Li, and R Yeung, "Network Information Flow," *IEEE Transactions in Information Theory,* July 2000.

[3] Abhinav Kamra, Vishal Misra, Jon Feldman, and Dan Rubenstein, "Growth codes: maximizing sensor network data persistence," *SIGCOMM Comput. Commun. Rev.,* vol. 36, pp. 255-266 2006

[4] C Gkantsidis, and P Rodriguez, "Network Coding for Large Scale Content Distribution," *INFOCOM, Miami,* 2005.

[5] S. Katti, D. Katabi, W. Hu Hariharan, and R. Medard, "The Importance of Being Opportunistic:Practical Network Coding for Wireless Environments," *Allerton,* 2005.

[6] C Fragouli, J Le Boudec, and J Widmer, "Network coding: an instant primer," *Computer Communication Review, ACM SIGCOMM,* Jan 2006.

[7] M. Halloush, and H. Radha, "Network Coding with Multi-generation Mixing," *CISS,* 2008.

[8] M. Halloush, and H. Radha, "Network Coding with Multi-generation Mixing: Analysis and Applications for Video Communication," *ICC,* 2008.

[9] "ISO/IEC 14496-10:2003 Information technology – Coding of audiovisual objects – Part 10: Advanced Video Coding."

[10] Heiko Schwarz, Detlev Marpe, and Thomas 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, SEPTEMBER 2007.

[11] P Chou, Y Wu, and K Jain, "Practical network coding," *Allerton Conference on Communication, Control, and Computing, Monticello, IL,* October 20, 2003.

[12] Jânio M. Monteiro, Carlos T. Calafate, and Mário S. Nunes, "Evaluation of the H.264 Scalable Video Coding in Error Prone IP Networks," *IEEE TRANSACTIONS ON BROADCASTING,* vol. 54, NO. 3, SEPTEMBER 2008.

[13] J. Vieron, M. Wien, H. Schwarz, ", "JSVM-9.4 Software, Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG," 2007.

[14] S. Mirshokraie and M. HefeedaLive, "Peer-to-Peer Streaming with Scalable Video Coding and Network Coding", In Proc. of ACM Multimedia Systems (MMSys'10), pp. 123-132, Scottsdale, AZ, February 2010. Slides

Mohammed Halloush received the B.S. degree from Jordan University of Science and Technology, Irbid, Jordan in 2004, the M.S. and the Ph.D. degrees in Electrical Engineering from Michigan State University, East Lansing, MI, USA in 2005, 2009 respectively.
Currently he is an Assistant professor in the department of Computer Engineering at Yarmouk University, Irbid Jordan. His research interests include network coding, multimedia communications, wireless communications and networking.

Hayder Radha received the B.S. degree (with honors) from Michigan State University (MSU) in 1984, the M.S. degree from Purdue University, West Lafayette, IN, in 1986, and the Ph.M. and Ph.D. degrees from Columbia University, New York, in 1991 and 1993, respectively, all in electrical engineering. Currently, he is a Professor of electrical and computer engineering (ECE) at MSU, the associate Chair for Research and Graduate Studies of the ECE Department, and the Director of the Wireless and Video Communications Laboratory at MSU. His current research areas include wireless communications and networking, sensor networks, network coding, video coding, stochastic modeling of communication networks, and image and video processing. He has more than 100 peer-reviewed papers and 30 U.S. patents in these areas.
Prof. Radha serves on the Editorial Board of IEEE Transactions on Multimedia and the Journal on Advances in Multimedia. He is an elected member of the IEEE Technical Committee on Multimedia Signal Processing and the IEEE Technical Committee on Image and Multidimensional Signal Processing (IMDSP). He is a recipient of the Bell Labs Distinguished Member of Technical Staff Award, the A&T Bell Labs Ambassador Award, the AT&T Circle of Excellence Award, the MSU College of Engineering Withrow Distinguished Scholar Award for outstanding contributions to engineering, and the Microsoft Research Content and Curriculum Award.
He is also a recipient of National Science Foundation (NSF) Theoretical Foundation, Network Systems, and Cyber-Trust awards.

Time-Shifted Streaming in a Tree-Based Peer-to-Peer System

Jeonghun Noh
ASSIA inc., Redwood City, CA, USA
Email: jnoh@assia-inc.com

Bernd Girod
Information Systems Laboratory, Department of Electrical Engineering
Stanford University, CA, USA
Email: bgirod@stanford.edu

***Abstract*— We propose a peer-to-peer (P2P) streaming system that multicasts live video as well as provides the live video as Video-on-Demand (VoD) during the live session. The proposed system allows users to individually pause and resume a live video stream, and to view video that was streamed before the particular user joined the session. Since live video content naturally results in many users watching it online at the same time, the P2P concept is leveraged to reduce server load.**

We extend our Stanford Peer-to-Peer Multicast (SPPM) protocol, originally designed for live video multicast, to support playback control. To achieve this in a P2P fashion, peers store received video packets in their local buffer and forward them at a later time when requested by other peers, which is called *time-shifted streaming*. To understand interaction between live and time-shifted traffic, we analyze video availability of the proposed system, which refers to how many peers possess video contents of a particular position. Motivated by the analysis, we propose *fast prefetching* and a parent selection scheme suitable for fast prefetching in order to further reduce server load by disseminating video quickly. Fast prefetching is possible because the relaxed transfer deadline of time-shifted streams and peers' extra uplink bandwidths are exploited. With simulations and mathematical analysis, it is shown that fast prefetching not only alleviates the requirement of server bandwidth by expediting video dissemination, but also mitigates video disruption due to peer churn.

***Index Terms*— P2P streaming, video-on-demand, time-shifted streaming, playback control**

I. INTRODUCTION

Two primary forms of video streaming are live streaming and video-on-demand (VoD). Live streaming often implies multicast communications as many users concurrently watch video. Video-on-demand, on the other hand, has been achieved by unicast communications because the temporal correlation among video segments users watch tends to be low. In this paper, we propose a P2P streaming system that allows viewers to individually pause a live video stream, rewind (even to contents prior to their time of joining), and jump forward (to contents beyond the current point of playback), which allows for the VoD of a live stream. As viewers watch the same content at a different time in a rather short time scale, the proposed P2P system exploits high temporal correlation among users' viewing points in time. The potential benefits from high temporal correlation among user arrival times for P2P VoD systems are well studied in [1]. In the proposed system, peers store received video in their local buffer. If a peer rewinds or jumps forward to content not present in its local buffer, the required video segment is retrieved from other peers. The video, buffered at some peer(s), is then streamed to the requesting peer asynchronously with respect to the live stream emanating from the server. To keep track of the video contents at peers, the peer database at the source peer is augmented with additional information of peers. The protocol is changed to be able to handle additional fields, such as a video position or a time stamp. This is called *time-shifted streaming*.

As mentioned earlier, the concept of time-shifting in a live session is closely related to asynchronous video streaming in VoD systems. VoD systems allow users to select and watch video content on demand. With playback control, also known as *trick mode*, users can also pause a video stream, rewind to an arbitrary position, and fast forward. Unlike live video, video files requested on-demand are pre-encoded and stored at video servers. If the available uplink bandwidth at the sender exceeds the video bitrate, the user can receive video faster than the play-back speed [2]. Traditionally VoD services were provided by servers. For instance, Chaining [3] and Patching [4] employ media servers and IP multicast, where IP multicast is used to reduce server workload. The authors of [5] study trade-off between the system profit and user experiences in a near-VoD system. In [6], patching and batching of server-based VoD service are augmented by taking into consideration heterogeneity in receivers.

As in server-based live streaming, servers deployed for VoD services may fail in the presence of a large population of users. Thus, many P2P-based VoD systems have been proposed to achieve scalable services. In order to achieve low workload on servers, the resources in the P2P network are leveraged for efficiently taking

Jeonghun Noh was with Stanford University, Stanford, CA 94305 USA. Manuscript received February 15, 2011; revised July 15, 2011; accepted September 30, 2011.

doi:10.4304/jcm.7.3.202-212

advantage of uplink bandwidth and storage contributed by peers [7]–[10]. For VCR-like operation (temporal random access), either distribution overlay is combined with look-up overlay [7], [8], or a separate look-up overlay is built to avoid a central look-up service [11]–[13]. In [9], [14], [15], viewers are able to watch a pre-recorded video from the beginning of the content. Dynamic Skip List [8] and oStream [7] provide users random temporal access to video content. The mesh-based P2P VoD systems are studied in [16]–[18]. The authors of [15] exploit the extra uplink bandwidths and peer storages by downloading video faster than playback speed. In [19], a user-log based prefetching algorithm is proposed. PPB, Peer-to-Peer Batching, combines peer-to-peer and IP multicast by using IP multicast for batching while a peer-to-peer network is used to provide the initial portion of video to new peers [20]. In [21], the authors propose a tree-based system that constructs dynamic multicast trees, called J-tree. The J-tree structure explicitly distinguishes the live multicast tree and playback trees that are implicitly built in our system. DRPSS, presented in [22], also supports live and time-shifted streaming in a single system based on the Distributed Hash Table overlay. P2TSS [23] supports time-shifted streaming in a live broadcast. In P2TSS, peers have two separate buffers; a buffer for local video playback and a shared buffer for serving time-shifted streams to other peers. The contents of the shared buffer do not change once the buffer becomes full of the received video. Unlike P2TSS, our system requires a single buffer to store the incoming video stream regardless of its shift in time (e.g., live or time-shifted stream).

The remainder of the paper is structured as follows. Section II introduces the Stanford Peer-to-Peer Multicast (SPPM) protocol [24] . In Section III, SPPM is extended to support time-shifted streaming. Section IV presents the analysis of video availability, which represents how many peers possess video contents of a particular position. In Section V, we describe fast prefetching to improve video availability. Fast prefetching allows peers to disseminate video faster. To facilitate video dissemination with fast prefetching, a new parent selection algorithm, called *maximum throughput*, is proposed and its effects are demonstrated with extensive simulations in Section VI.

II. Stanford Peer-to-Peer Multicast

Stanford Peer-to-Peer Multicast (SPPM) is a P2P system that achieves low-latency and robustness in live video multicast. Like in [25], [26], SPPM adopts the push approach for low-latency data dissemination. The push approach requires a persistent route among peers. This route is built by connecting a pair of peers using unicast connections, such as TCP or UDP. Since unicast connections are established on top of the IP layer, the collection of the unicast connections is often called an *overlay*. Each data path, a chain of peers in the overlay, starts from the server. On the data path, all intermediate peers act as both parents and children; peers that relay video packets are the *parents* of the peers that receive the packets. Peers that receive video packets are the *children* of the peers that relay the packets. When a parent peer can relay packets to more than one child peer, then the path naturally evolves from a chain to a tree structure, with the server being the root and peers being intermediate nodes and bottom nodes. Note that the desired distribution structure becomes a spanning tree because no packet needs to visit the same peer more than once.

Fig. 1 illustrates an example overlay in which two complementary trees are constructed among a small group of peers. Typically, we use 4 to 8 trees, and peer groups might be much larger [27]. As peers join the system, the trees are incrementally constructed in a distributed manner. When a new peer contacts the video source, the video source replies with session information, such as the number of multicast trees and the video bit rate. It also sends a list of candidate parents randomly chosen from the table of participating peers it maintains. The new peer then probes each candidate parent to know about their current status. After receiving probe replies, the best candidate parent is selected and contacted for each tree by minimizing the height of the distribution tree. Once the selected candidate parent accepts the attachment request, a data connection is established between the parent and the new peer. After data transmission starts, each child peer periodically sends *hello* messages to their parents. When a peer leaves the system ungracefully, its parents detect it by observing consecutive missing *hello* messages, and stop forwarding video to the child. The departing peer's children notice that neither video packets nor a response to *hello* messages arrive. Each abandoned child then initiates procedure to connect to a different parent node in the tree.

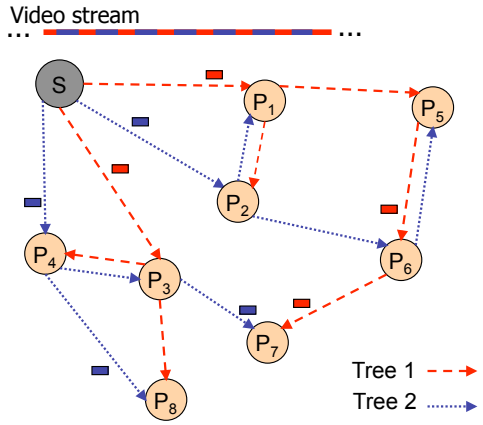


Figure 1. A peer-to-peer overlay consisting of two complementary distribution trees. The encoded video is packetized and split into disjoint substreams, one for each distribution tree.

To provide the best video quality despite congestion and peer churn, SPPM employs sender-driven packet scheduling [28] in conjunction with receiver-driven retransmission requests [29]. Acting as a relay, each peer schedules the next packet to forward by comparing the importance of packets in its output queue. As a receiver,

each peer evaluates the importance of missing packets by computing the expected video quality improvement if the corresponding frame is received before its playout deadline. Retransmission requests of selected missing packets are then sent to alternative parents. This NACK-based retransmission is advantageous as packet error rate is often not stationary over time nor uniform among peers. Since packet losses typically occur in rare bursts, such feedback error control is generally superior to forward error control. Feedback implosion, prohibiting feedback error control in network-layer multicast, is not an issue for a P2P overlay because retransmissions are handled locally between a parent and a child peer.

III. Time-Shifted Streaming

In this section, SPPM is extended to allow users to control their playback in the time domain [30]. With playback control, also known as *trick mode* in video-on-demand systems, peers can individually pause a live video stream, rewind to an arbitrary position (even to contents prior to their time of joining), and fast forward until the latest position of the live video. In the extended SPPM, peers cache the received video in their local buffer. For pause/resume, the locally cached video can be played back at a later time. When a peer rewinds or jumps forward to content not present in its local buffer, the requested video segment, buffered at some peer(s), is streamed to the peer asynchronously unlike the live stream being multicast to peers. This is called *time-shifted streaming*. As we will see later, SPPM supports both the live stream and time-shifted streams while keeping the system's scalability.

While extending the SPPM system for time-shifted streaming, the fundamental architecture of the system is kept intact so that live video multicast is seamlessly supported. In this section, we describe only the system extensions added to the existing architecture.

A. Preliminaries

As in live stream multicast, the video server (source peer) starts to stream a live video starting at time 0. Let $V(x)$ denote a video frame associated with position x in time, $x \geq 0$. The live video emanating from the server at time t is thus represented by $V(t)$. A time-shifted stream is represented by $V(t-d)$ with delay d, $0 < d \leq t$, at time t. Suppose that a new peer or an existing peer requests video $V(x)$ at time t. If the peer requests the live video $V(t)$, the peer is called *LS* peer. If the peer requests a time-shifted stream $V(t-d)$, $0 < d \leq t$, then the peer is called *TS* peer. For LS peers, the live video is immediately relayed from peer to peer. For TS peers, the live stream must be delayed by time d somewhere in the system, either at the server or at the peers. Delayed streaming, or *time-shifted streaming*, is achieved by storing past video packets in peers' buffers while they watch the video[1] and transmitting them when requested.

The time-video plot in Fig. 2 illustrates the trajectories of the live stream, Peer 1 (LS peer), and Peer 2 (TS peer). The trajectories indicate the video contents stored at the peers over time. The vertical axis represents the time stamp of video frames (a time stamp is the time when a video frame is encoded at the server). In Fig. 2, Peer 1 requests the live video $V(t_1)$ at time t_1. Its video trajectory overlaps the live stream trajectory. At time t_2, Peer 2 requests the video starting from $V(x_2)$. Since the requested video is stored at Peer 1, Peer 2 can obtain the video delayed by $t_2 - x_2$ from Peer 1.

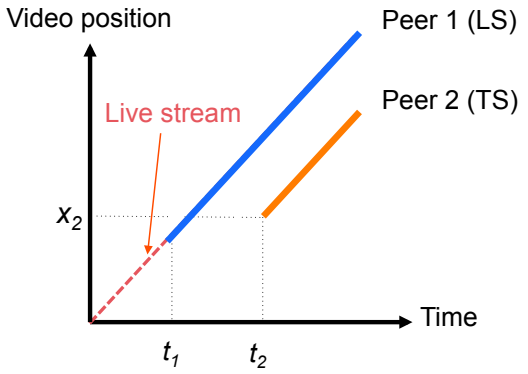


Figure 2. Video trajectories of live video, Peer 1, and Peer 2. Peer 1, an LS peer, requests the live video. Peer 2, a TS peer, requests the video delayed by $t_2 - x_2$.

B. Peer Database

In the system, peers collectively form a distributed video storage. To facilitate the use of the videos stored by peers, locating video among peers is essential. To locate the video of a particular position, the video server maintains a database comprising each peer's identifier, the time the video is requested, the time stamp of the initial video frame, and the time stamp of the latest video stored in the peer's buffer. The latest time stamp is estimated by the server based on each peer's initial video position and request time. Estimation by the server reduces the amount of control traffic between the server and peers by truncating the number of buffer state updates. To prevent the estimation from deviating too far from the actual buffer states, peers may report their buffer state intermittently (see Section V-A).

C. Video Request and Connection Set-up

When a new peer joins or an existing peer seeks a video of a particular position, it sends a query requesting a certain video position to the server. The server refers to the peer database it maintains. It returns a list of randomly chosen parent candidates whose video buffer may contain the requested (possibly time-shifted) video. The peer then probes each parent candidate to learn about the candidates' current status, including time stamps of the stored video and available bandwidth. After the peer receives probe replies, it selects and requests for each

[1] A peer shares its storage only when it participates in a session. For a typical two hour-long video stream encoded at 600 kbps, about 540 MB of storage is required. This is a moderate space requirement considering today's typical personal computers.

multicast tree the best parent candidate that will accept it as a child. We discuss parent selection criteria in detail in Section V. Once the parent candidates accept the video request, data connections are established between the parents and the child.

D. Asynchronous Video Transmission

Connections established for time-shifted streaming are different from connections that constitute the live video multicast trees. In the latter case, packets are immediately relayed to the next hop in the multicast tree, whereas in the former, packets are asynchronously relayed over the time-shifted streaming connection. Multiple complementary trees are the basis of data dissemination in SPPM. As LS peers, TS peers receive packets from multiple parents. Parents asynchronously relay video packets to their TS peers by ensuring that a packet travels along the tree designated to itself.

A parent peer dedicates a streaming window for each TS child peer. Fig. 3 shows a streaming window of a TS child that requests video from position x_1 at time t_1. The lower end of the window, denoted by W_L, is the oldest video packet the parent can transmit. Since packets that are past their display time (called the play-out deadline) are not decoded at the child, the value of W_L is set to the minimum of the initial video position in time and of the time stamp of the video frame displayed at the child. The upper end of the window, denoted by W_U, corresponds to the time stamp of the latest video packet the parent can send. The value of W_U is selected in such a way that the child's downlink is not congested by the parent, and the child does not suffer from buffer underflow. The parent regularly searches for packets whose time stamps are between W_L and W_U in its video buffer. In addition, only packets that belong to the parent's tree ID are selected to construct the corresponding substream.

To avoid duplicate transmission, the history of transmitted packets is maintained for a finite period of time. Selected packets are marked with their play-out deadline and put into the network output queue. The output queue sorts video packets according to their play-out deadline. Therefore, urgent packets, such as retransmitted packets or live video packets, are sent earlier than packets with a relaxed deadline, such as packets in time-shifted streams.

At a child peer, packets that belong to different substreams may arrive out-of-order. Once received, packets are ordered according to their positions in the video stream before being passed to the video decoder.

E. Recovery from Disconnect

After video transmission starts, each child peer periodically sends *hello* messages to its parents. When a peer leaves the system ungracefully, its parents detect this by observing consecutive missing *hello* messages and stop forwarding video to the child. The departing peer's children notice that neither video packets nor responses to *hello* messages arrive. Each abandoned child then initiates a recovery procedure to connect to a different parent node in the tree. During the recovery procedure, LS peers request the latest portion of the video in order to keep end-to-end transmission delay as low as possible. TS peers, on the other hand, request the next position of the video they received if the position was ahead of their playback position. Otherwise, TS children request the playback position of the video.

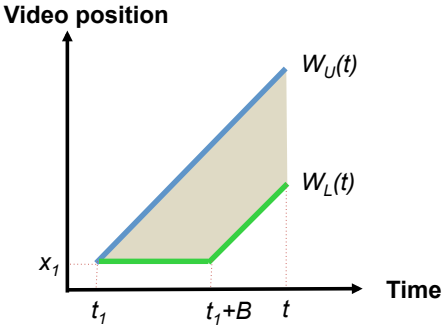


Figure 3. A streaming window for a TS child that requests video from position x_1 at time t_1. W_L is its lower end and W_U is its upper end. B is the pre-roll delay of play-out at the child. Packets whose time stamps fit in the streaming window are transmitted unless they have been sent already.

IV. Analysis of Video Availability

This section presents the analysis of the availability of time-shifted video segments among peers, which is closely associated with system scalability. High video availability lessens the need for making a connection to the server, thereby reducing server load. The analysis also sheds light on the interaction between live and time-shifted streams. Suppose that peers join the system according to a Poisson process $N(t)$ with rate λ [31]. Their lifetime follows an exponential distribution[2] with parameter μ ($1/\mu$ is the average peer lifetime). We assume that the probability that the next peer is an LS peer follows a Bernoulli process with parameter α. When a TS peer requests a video, its video access pattern is assumed to be uniform. This assumption allows us to be free from any particular video content and it provides the worst-case study because overlaps between peers' buffered contents are minimized, which results in minimizing the video availability. Based on this assumption, the position x is selected, at time t, uniformly between 0 and t. We further assume that a peer always receives the video it requests, by connecting either to another peer or to the video server.

Let $M(t, x; \lambda, \mu, \alpha)$ denote the number of peers possessing the video at position x at time t. To analyze the

[2] A peer lifetime is known to follow a long-tail distribution, such as the Zipf distribution [27], [32], [33]. However, the exponential distribution is still a popular choice for both analysis and simulation study because it reasonably approximates the actual distribution. More importantly, this assumption allows the system analysis to be tractable, and the system behavior can be predicted by observing only a few primary system parameters.

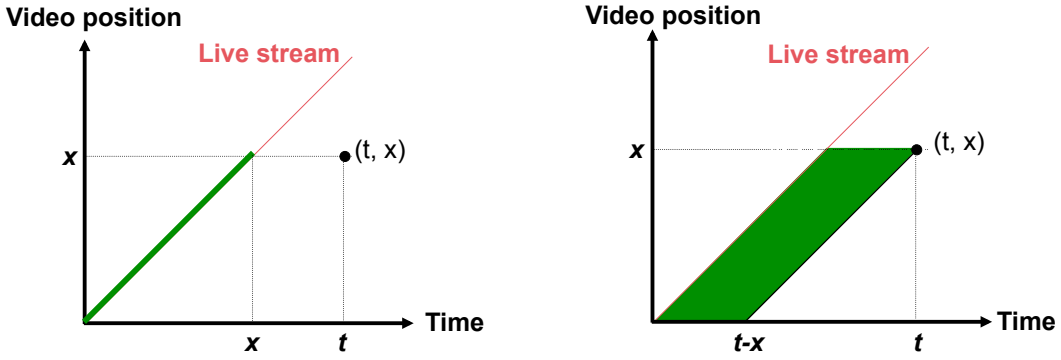


Figure 4. Initial video positions and request times for LS peers (Left) and TS peers (Right) that possess video at position x at time t. Left: An LS peer whose arrival time is earlier than x has video at position x at time t. Right: A TS peer whose request time and initial video position are within the area surrounded by $(0,0)$, $(t-x,0)$, (t,x), and (x,x) on the time-video plot has video at position x at time t.

availability of video contents among peers, we compute the expected value of $M(t,x;\lambda,\mu,\alpha)$, given by

$$\begin{aligned} &E\{M(t,x;\lambda,\mu,\alpha)\} \\ &= \sum_{k=1}^{\infty} k \Pr\{M(t,x;\lambda,\mu,\alpha)=k\} \\ &= \sum_{k=1}^{\infty} k \sum_{i=k}^{\infty} \Pr\{M(t,x;\cdot)=k|N(t)=i\} \Pr\{N(t)=i\}, \end{aligned} \tag{1}$$

where $N(t)$ represents the number of peers that have arrived at the system between 0 and time t. Given $N(t)=i$, since $N(t)$ is a Poisson process, peers' arrival times $S_1, \ldots, S_i$, considered unordered random variables, are distributed independently and uniformly in the interval $[0,t]$ [34]. This allows us to study each peer's behavior independently regardless of peers' join sequence. Consider a peer that joins the system between 0 and t. Let p denote the probability that the peer is still present and possesses video at position x at time t. Then, $E\{M(t,x;\lambda,\mu,\alpha)\}$ in (1) can be rewritten as

$$\begin{aligned} &E\{M(t,x;\lambda,\mu,\alpha)\} \\ &= \sum_{k=1}^{\infty} k \sum_{i=k}^{\infty} \binom{i}{k} p^k (1-p)^{i-k} e^{-\lambda t} \frac{(\lambda t)^i}{i!} \\ &= \sum_{k=1}^{\infty} \frac{1}{(k-1)!} \left(\frac{p}{1-p}\right)^k e^{-\lambda t} \sum_{i=k}^{\infty} \frac{((1-p)\lambda t)^i}{(i-k)!} \\ &= \sum_{k=1}^{\infty} \frac{1}{(k-1)!} \left(\frac{p}{1-p}\right)^k e^{-\lambda t} e^{(1-p)\lambda t} ((1-p)\lambda t)^k \\ &= e^{-p\lambda t} \sum_{k=1}^{\infty} \frac{(p\lambda t)^{k-1}}{(k-1)!} p\lambda t \\ &= p\lambda t. \end{aligned} \tag{2}$$

We compute p by considering the cases of TS peer arrivals and LS peer arrivals separately. Suppose that a peer arrives at time s, $s \le t$. If it is a TS peer, then it contains video at position x at time t only when its arrival (or request) time and its requested video position are inside the area surrounded by $(0,0)$, $(t-x,0)$, (t,x), and (x,x) on the time-video plot in Fig. 4. One can easily check this by drawing a video trajectory (in parallel with the live stream trajectory) from any point inside the area until time t and examine whether the trajectory passes or touches the horizontal line drawn from $(0,x)$ to (t,x). Note that the video is not available above the live stream trajectory. Let $g(s;t,x)$ denote the probability that a TS peer joining at time s contains the video x at time t. The value of $g(s;t,x)$ is 1 when $0 \le s \le (t-x)$, $(t-x)/s$ when $(t-x) < s \le x$, and $(t-s)/s$ when $x < s < t$, respectively. Based on the exponential distribution of peer lifetime, the probability that the peer is still present at time t is $e^{-\mu(t-s)}$. Then, the probability that a TS peer that contains video at x at time t, $l(t,x;\mu)$, is

$$\begin{aligned} l(t,x;\mu) &= \int_0^t e^{-\mu(t-s)} g(s;t,x) \frac{1}{t} ds \\ &= \frac{e^{-\mu t}}{t} \left\{ \int_0^{t-x} e^{\mu s} ds + (t-x) \int_{t-x}^{x} \frac{e^{\mu s}}{s} ds \right. \\ &\quad \left. + \int_x^t e^{\mu s} \frac{t-s}{s} ds \right\}. \end{aligned} \tag{3}$$

When an LS peer arrives at time s, it possesses video at position x at time t only when $s \le x$. The set of the pairs of the live video position and arrival time is the line segment from $(0,0)$ to (x,x) shown on the left side of Fig. 4. Then, by incorporating the likelihood of a peer's departure before time t, the probability that an LS peer that contains video at x at time t is expressed as

$$\int_0^x e^{-\mu(t-s)} \frac{1}{t} ds = \frac{e^{-\mu t}(e^{\mu x}-1)}{\mu t}. \tag{4}$$

Finally, p can be written as

$$\begin{aligned} p &= \Pr\{L^c\} \Pr\{x \text{ is possessed at time } t | L^c\} + \\ &\quad \Pr\{L\} \Pr\{x \text{ is possessed at time } t | L\} \\ &= (1-\alpha) l(t,x;\mu) + \frac{\alpha e^{-\mu t}(e^{\mu x}-1)}{\mu t}, \end{aligned} \tag{5}$$

where L is the event that an LS peer joins the system. By

inserting (5) into (2), we rewrite (2) as

$$E\{M(t,x;\lambda,\mu,\alpha)\} = (1-\alpha)\lambda tl(t,x;\mu) + \alpha\frac{\lambda}{\mu}e^{-\mu t}(e^{\mu x}-1), \quad (6)$$

where $(1-\alpha)\lambda tl(t,x;\mu)$ corresponds to the expected number of TS peers that possess video at position x at time t and $\alpha\frac{\lambda}{\mu}e^{-\mu t}(e^{\mu x}-1)$ corresponds to the expected number of LS peers that possess x at time t.

In (6), the video availability is predicted to increase linearly as the peer's join rate λ increases. This result demonstrates that the system is self-scalable. The availability of the early portion of the video continues to decrease as $t \to \infty$, which indicates that time-shifted streams have to be provided by the video server at a later time. In Fig. 5, the video availability is predicted by evaluating (6) with λ = 1.125 joins per second, $1/\mu$ = 120 seconds, and α = 0.5. In the figure, the video near the live stream is available at many peers, the majority of which are LS peers. As time progresses, more video frames become available and the overall video availability decreases.

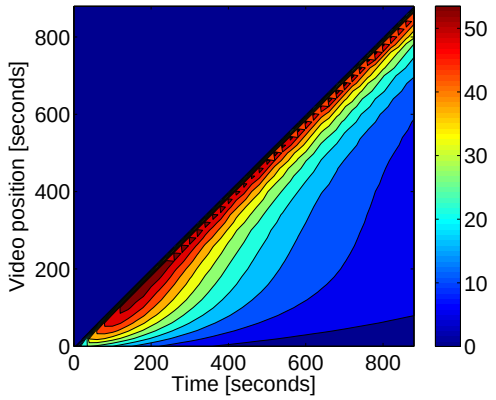


Figure 5. Video availability on the time-video plot. The value at (t,x), predicted by the model, represents the expected number of peers available for video at position x at time t (λ = 1.125 joins per second, $1/\mu$ = 120 seconds, and α = 0.5).

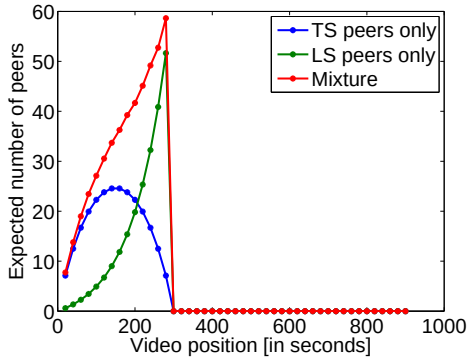


Figure 6. Model: video availability at 300 seconds. (λ = 1.125 joins per second, $1/\mu$ = 120 seconds, and α = 0.5)

Figs. 6 and 7 illustrate the expected video availability at 300 and 900 seconds, respectively. In these figures, the contribution from LS peers and TS peers is separately shown. The expected number of TS peers having video at position x at time t peaks when $x = t/2$, whereas the expected number of LS peers at time t peaks at the live video position of $x = t$ and its value is $\lim_{t\to x}\alpha\frac{\lambda}{\mu}e^{-\mu t}(e^{\mu x}-1) \approx \alpha\lambda/\mu$. Although the videos stored at either LS or TS peer have high overlaps at time 300s, the overlap dramatically decreases at time 900s. This result implies that at a later time of the session more TS peers have to connect to the server, which was already observed in Fig. 5.

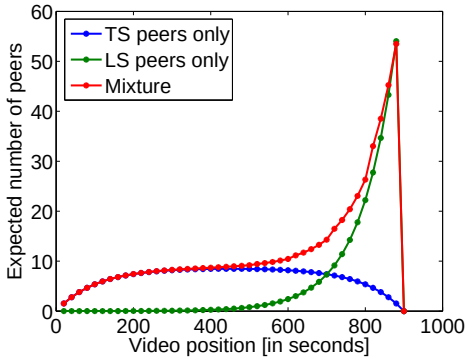


Figure 7. Model: video availability at 900 seconds. (λ = 1.125 joins per second, $1/\mu$ = 120 seconds, and α = 0.5)

V. Fast Prefetching

In the previous section, we have observed that the video availability decreases over time. Server load is highly affected by video availability because higher video availability allows more peers to obtain video from other peers, thus reducing the number of connections made to the server. In this section we propose *fast prefetching*, which allows peers to download video more rapidly. Fast prefetching exploits the relaxed transfer deadline of time-shifted streams when compared to the live stream. As an input device such as a video camera generates a video signal, the live video is immediately encoded and transmitted to peers. On the other hand, time-shifted streams are the past portion of the video, already encoded and stored somewhere in the system. Fast prefetching also exploits the extra uplink bandwidth of peers.

A. Improving Video Availability

Suppose a TS child receives a time-shifted stream from its parent. When its parent has sufficient uplink bandwidth, the stream can be pushed to the child faster than playback speed. Since fast prefetching facilitates video dissemination among peers, video availability can be improved accordingly. Fast prefetching also improves the video playback quality of TS peers. With fast prefetching, peer buffers grow faster than playback speed, which reduces the urgency for acquiring later packets. When a peer is disconnected due to failure of its parent, it reconnects to the overlay. Since the peer has more video frames to play back until it resumes to receive later video,

it experiences statistically less video disruption due to buffer underflow.

Since fast prefetching is based on the relaxed transfer deadline of time-shifted streams, it does not benefit LS peers. When LS peers are disconnected from the overlay, they often experience buffer underflow because the play-out deadline of a video packet is kept small in live streaming. To compensate for a tight play-out deadline, LS peers connect to peers that can minimize the number of logical hops to the source [35]. Using this criterion, LS peers attempt to reduce disruptions in receiving the video when their parent fails or leaves unexpectedly.

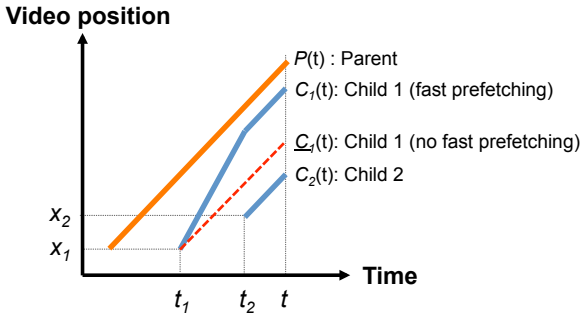


Figure 8. An example of fast prefetching for TS peers. The parent peer can serve two children concurrently. At time t, the excess download at Child 1 due to fast prefetching is $C_1(t) - \underline{C}_1(t)$.

The effects of fast prefetching are illustrated in Fig. 8. In this illustration, the uplink of the parent is assumed to be twice the video bitrate. $P(t)$ is the video trajectory of the parent. After Child 1 connects to its parent, it prefetches the video faster than the playback speed until Child 2 connects to the same parent. The trajectories of the downloaded video for Child 1 and 2 are depicted as $C_1(t)$ and $C_2(t)$, respectively. If fast prefetching is not employed, Child 1 receives video at the playback speed, which is marked by $\underline{C}_1(t)$. Since fast prefetching enables peers to download video faster than at the playback speed, the peer database's estimation on the peers' buffer state may deviate from the actual buffers. To correct mismatches incurred at the database, peers intermittently report their buffer state to the video server.

B. Selecting Parents that Maximize Prefetching

LS peers choose their parents according to the number of hops to the server, which is certainly not optimal for TS peers because the video disruption is less likely due to fast prefetching. To maximize prefetching, we propose a new parent selection algorithm for TS peers, which estimates video download for each parent candidate. When Parent Candidate i is probed by a TS peer that requests the video from position x, the parent candidate reports (1) the download rate q_i from its current parent, (2) the number of bits cached at Parent Candidate i and requested by the TS peer (e.g., a part of the requested substream and beyond the position x), denoted by l_i, and (3) the maximum upload rate it can offer to the TS peer, denoted by r_i. In Fig. 9, q_i, r_i, and l_i reported by Parent Candidate i are depicted over time. The Y-axis shows the video position in bits. Note that the rates and the buffer amount were adjusted to reflect the number of trees. As each multicast tree delivers a $1/k$ fraction of the video bitstream, q_i, r_i, and l_i were multiplied by k for the illustration (k is the number of trees).

After probing the candidates, the TS peer computes the amount of prefetched video that it expects to receive time D after connecting to each parent candidate: if the peer's download catches up with its parent's download before time D has elapsed, the expected download is $D \cdot q_i + l_i$. Otherwise, the expected download is $D \cdot r_i$ (see Fig. 9). The value of D is chosen to be sufficiently small, as q_i and r_i may change over time. The best candidate k is chosen according to the rule

$$k = \arg\max_i \min\left(D \cdot q_i + l_i, D \cdot r_i\right). \tag{7}$$

The video server is selected as a parent only when there is no peer possessing the requested video or when no peer that possesses the requested video has available bandwidth. It should also be noted that the server does not participate in fast prefetching in order to minimize its uplink bandwidth usage.

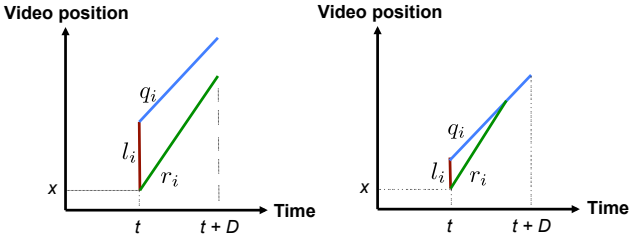


Figure 9. Expected download at a child peer after time D when a child peer joining at time t connects to parent candidate i. Left: child buffer lags behind parent buffer at time $t + D$. Right: child buffer catches up with parent buffer. (q_i: download rate of Parent Candidate i from its parent, l_i: buffered video in bits at Parent Candidate i, r_i: upload rate from Parent Candidate i to the child peer.)

C. Uplink Bandwidth Partitioning

To support fast prefetching, peers partition their available uplink bandwidth to children as follows:

(Step 1) Allocate the tree bitrate $r(= R/k)$ to every LS child. R is the video bitrate and k is the number of trees.

(Step 2) Allocate r to every TS child who completed fast prefetching (the child's latest video time stamp is the same as the parent's latest video time stamp).

(Step 3) Compute $\tilde{r} = U/n$, $\tilde{r} \geq r$. U is the remaining uplink bandwidth after Step 2, and n is the number of TS children who have not been assigned bandwidth in the previous steps. If some TS children have a smaller downlink capacity than $\tilde{r}$, the bandwidth allocated to them is curtailed in order not to overwhelm them. The remaining bandwidth is evenly allocated to the remaining TS children.

The algorithm is illustrated in Fig. 10. Note that the server performs no bandwidth partitioning because it supports no fast prefetching.

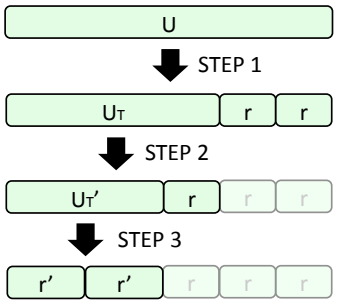


Figure 10. Uplink bandwidth partitioning. In Step 1, LS children are assigned the tree bitrate. In Step 2, TS children that have completed fast prefetching are assigned the tree bitrate. In Step 3, TS children that can perform fast prefetching are assigned the remaining uplink bandwidth evenly. Every peer except the server regularly performs bandwidth partitioning.

VI. Experimental Results

In the experiments in the ns-2 simulator [36], 150 peers join and leave repeatedly with an average lifetime of 120 seconds. We used the *Mother and Daughter* sequence, encoded at $R = 420$ kbps, using H.264/AVC with an intraframe period of 15 frames. Two B frames were encoded between anchor frames. The video server's uplink bandwidth was set to $20R$. $2R$ was allocated to LS peers, and the rest of the server's uplink bandwidth was allocated to TS peers. The peer uplink bandwidth was set to $2R$ and the download bandwidth was set to $4R$. On joining at time t, half of the peers watched the live stream. The rest of them watched the stream shifted by a random delay ranging from 0 to t.

First, we examine the system without fast prefetching. Fig. 11 is the plot of the experimental results for video availability among peers. The value associated with the pair $(t,\ x)$ represents the average number of peers that possess video at position x at time t. The result closely matches the video availability predicted by the model in Section IV and shown in Fig. 5. Fig. 12 shows the video availability for the model and the experiments in terms of the number of TS peers that possess video at position x $(0 \leq x \leq 900)$ at time 900s (no LS peers were present). It is notable that video availability peaks at 450s, the half point of the video available at 900s, under the assumption of random video access. As with Fig. 5 and 11, the video availability predicted by the model closely matches the averaged video availability of the experiments. As the session proceeds, only a small neighborhood of the video near the live stream is cached by the majority of the peers. The extent of the neighborhood is determined by several factors, one of which is the average peer lifetime. The expected number of peers that cache a past portion continues to decrease over time, as expected from the analysis.

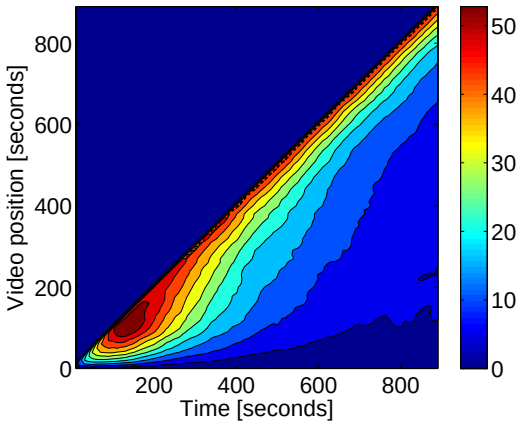


Figure 11. Video availability averaged over 100 simulation runs. (λ = 1.125 joins per second, $1/\mu$ = 120 seconds, and α = 0.5 are used.)

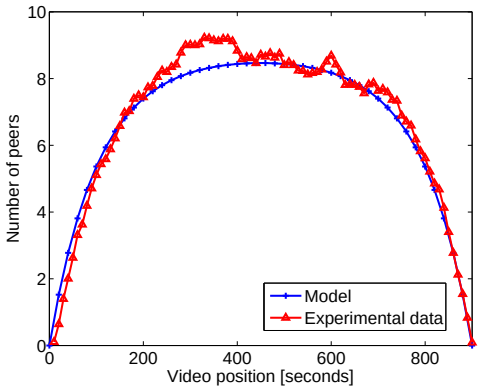


Figure 12. Video availability at 900 seconds. Averaged over 100 simulation runs. Only TS peers (α = 0) joined the system at the rate of λ = 1.125 joins per second. $1/\mu$ = 120 seconds.

Next, we study the system behavior when fast prefetching is employed. In Fig. 13, the average number of direct children of the server is plotted for three different cases:

- No fast prefetching is performed.
- Fast prefetching with parent selection *minimum hops*. For both TS and LS peers, parents that minimize the number of logical hops to the server are selected.
- Fast prefetching with parent selection *maximum throughput*. The parent selection algorithm described in Section V-B is used (D is set to 10 seconds).

The graphs show that with fast prefetching, fewer connections directly to the server are established because peers find their requested video among other peers more often than without fast prefetching. The proposed parent selection algorithm further reduces the server load, compared to the case where parent selection *minimum hops* was used.

As the session progresses, more TS peers connect to the server. This causes peers to experience less peer churn, leading to higher video quality at the cost of server

bandwidth. Since only few TS peers need to connect to the video server at the early stage of a session, we evaluate video quality for the first 900 seconds. When there are missing or late packets that do not arrive in time, the last fully decoded frame is displayed until the next I frame arrives. Table I summarizes average video quality and server load. The average PSNR is computed at the end of the simulation by taking the average of the PSNR value of the video frames displayed at all peers. For missing frames, copy error concealment (display of the previously decoded frame) is used. The table indicates that fast prefetching achieves higher or similar video quality with reduced server load. We observe that fast prefetching with parent selection *maximum throughput* outperforms no fast prefetching by 2 dB for TS peers with about 40% sever load reduction. In Fig. 13, the server load remains low at the early stage of the session because TS peers can easily find peers to connect to. Toward the end of the session, in contrast, more TS peers need to receive video from the video server as fewer peers are found to cache video.

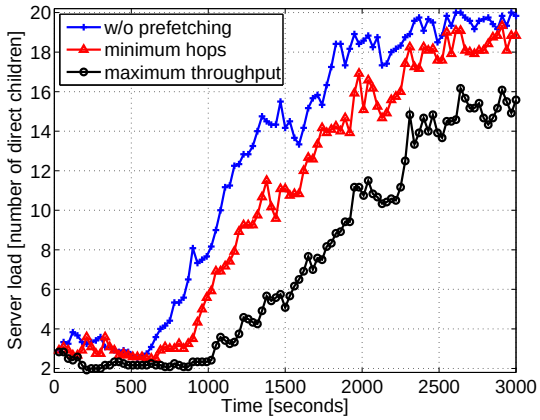


Figure 13. Comparison of server load over time (averaged over 10 simulation runs).

TABLE I.
EFFECTS OF FAST PREFETCHING (COMPUTED OVER THE FIRST 900 SECONDS). VIDEO QUALITY IS REPRESENTED BY THE AVERAGE PSNR OF PEERS. SERVER LOAD IS THE AVERAGE NUMBER OF DIRECT CHILDREN OF THE SERVER. *Hops* AND *throughput* ARE PARENT SELECTION CRITERIA.

	Parent selection criterion	Average PSNR LS peers [dB]	Average PSNR TS peers [dB]	Server load [num. of children]
Without prefetching	hops	40.6	40.4	3.7
With prefetching	hops	41.0	42.2	2.9
	throughput	41.5	42.3	2.2

Finally, we investigate the sensitivity of fast prefetching against D. Recall that D denotes a *look-ahead time*, the time a TS peer looks ahead to compute the expected download. In the experiment, we had 75 TS peers with uplink speed of $2R$, with 60 seconds of average lifetime and 6 seconds of average off-time. Simulations ran for 900 seconds and the average download speed at which a peer received video was averaged over 20 simulations. Fig. 14 shows the average download rate, which is normalized by the video bitrate. We observe that the average download peaks near 0 and decreases as D increases, indicating that large D may overestimate expected downloads because selected parents may depart the system. However, the download speed is not very sensitive to the value of D, showing only a 3% drop from the peak value for $1 \leq D \leq 20$. The download rate becomes nearly flat as D approaches the average peer lifetime. It is worth noting that when $D \approx 0$, the maximum-throughput algorithm in Eq. 7 can be viewed as a maximum-download-rate algorithm; when $D \approx 0$ and l_i is moderate, $\min(D \cdot q_i + l_i, D \cdot r_i)$ approximates to $D \cdot r_i$, thereby $\arg\max_i \min(D \cdot q_i + l_i, D \cdot r_i) \approx \arg\max_i r_i$.

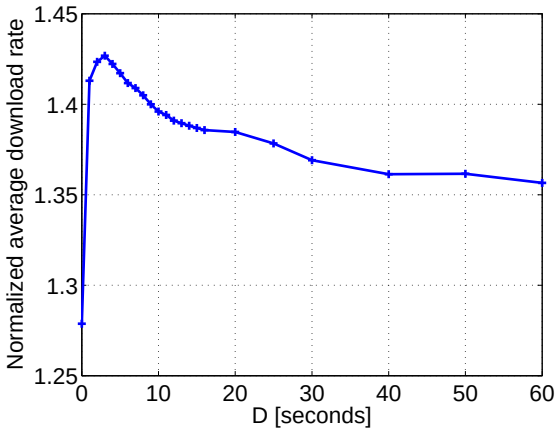


Figure 14. Normalized average download rate with respect to D, look-ahead time.

VII. CONCLUSIONS

In this paper, a tree-based P2P live streaming system like SPPM is shown to support time-shifted streaming without any fundamental changes in the system architecture. To understand the distribution of video segments among peers, we modeled the average number of peers that possess the required video segments cached in their buffers, defined as video availability. The model and the experiments revealed that few peers request time-shifted streams from the server at the early stage of the session because they can connect to peers who have been caching the live stream, and that the server load increases as the session progresses since a smaller number of peers are likely to have the requested time-shifted streams. The video availability improves more significantly when fast prefetching is combined with the maximum throughput parent selection. This shows that the overlay may look very different depending on what objectives are pursued in the overlay construction.

In this work, we considered the continuous bitstream consisting of past video packets stored in peers' local storage. One can devise more sophisticated caching algorithms, which may allow multiple chunks of video that are not necessarily continuous in bitstream. Server bandwidth allocation is also an important research problem. We have found that time-shifted streaming often requires more server bandwidth than live streaming because peers that request past portion of video need to connect to the server if the requested video chunk is not available from other peers. Thus, the server needs to divide its finite uplink bandwidth between live streaming and time-shifted streaming dynamically in order to maximize the overall video quality of peers.

REFERENCES

[1] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon, "I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system," in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, ser. IMC '07. New York, NY, USA: ACM, 2007, pp. 1–14.

[2] Y. Shen, Z. Shen, S. Panwar, K. Ross, and Y. Wang, "On the design of prefetching strategies in a peer-driven video on-demand system," *IEEE Intern. Conf. Multimedia and Expo*, pp. 817 – 820, Jul. 2006.

[3] S. Sheu, K. Hua, and W. Tavanapong, "Chaining: a generalized batching technique for video-on-demand systems," in *IEEE Proc. Intern. Conf. Multimedia Computing and Systems*, Ottawa, Canada, Jun. 1997, pp. 110 – 117.

[4] K. A. Hua, Y. Cai, and S. Sheu, "Patching: A multicast technique for true video-on-demand services," in *Proc. ACM Intern. Conf. Multimedia*, University of Bristol, United Kingdom, 1998, pp. 191–200.

[5] F. A. T. S.-H. G. Chan, "Tradeoff between system profit and user delay/loss in providing near video-on-demand service," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 11, no. 8, pp. 916–927, Aug. 2001.

[6] B. Qudah and N. Sarhan, "Towards scalable delivery of video streams to heterogeneous receivers," in *Proc. ACM Intern. Conf. Multimedia*, Oct. 2006, pp. 347 – 356.

[7] Y. Cui, B. Li, and K. Nahrstedt, "oStream: Asynchronous streaming multicast in application-layer overlay networks," *IEEE Journ. on Selected Areas on Comm.*, vol. 22, no. 1, pp. 91 – 106, 2004.

[8] D. Wang and J. Liu, "Peer-to-peer asynchronous video streaming using Skip List," *Proc. of IEEE International Conference on Multimedia and Expo*, 2006.

[9] Y. Guo, K. Suh, J. Kurose, and D. Towsley, "P2Cast: peer-to-peer patching scheme for VoD service," in *Proc. Intern. Conf. World Wide Web*. ACM Press New York, NY, USA, May 2003, pp. 301–309.

[10] T. Do, K. Hua, and M. Tantaoui, "Robust video-on-demand streaming in peer-to-peer environments," *Computer Communications*, vol. 31, no. 3, pp. 506–519, Feb. 2008.

[11] H. Chi, Q. Zhang, and S. Shen, "Efficient search and scheduling in P2P-based media-on-demand streaming service," *IEEE Journ. on Selected Areas on Comm.*, vol. 25, no. 1, pp. 119–130, Jan. 2007.

[12] W. Yiu, X. Jin, and S. Chan, "VMesh: distributed segment storage for peer-to-peer interactive video streaming," *IEEE Journ. on Selected Areas on Comm.*, vol. 25, no. 9, pp. 1717–1731, 2007.

[13] Y. Guo, K. Suh, J. Kurose, and D. Towsley, "Directstream: A directory-based peer-to-peer video streaming service," *Computer Communications*, Jan. 2008.

[14] T. Do, K. A. Hua, and M. Tantaoui, "P2VoD: providing fault tolerant video-on-demand streaming in peer-to-peer environment," vol. 3, June 2004, pp. 1467—1472.

[15] C. Huang, J. Li, and K. W. Ross, "Peer-assisted vod: Making internet video distribution cheap," Feb. 2007.

[16] X. Xu, Y. Wang, S. Panwar, and K. W. Ross, "A peer-to-peer video-on-demand system using multiple description coding and server diversity," *Intern. Conf. Image Processing*, vol. 3, pp. 1759 – 1762, Oct. 2004.

[17] Z. Liu, Y. Shen, S. Panwar, K. W. Ross, and Y. Wang, "Efficient substream encoding and transmission for P2P video on demand," in *IEEE Proc. Intern. Packet Video Workshop*, Nov. 2007, pp. 143 – 152.

[18] S. Annapureddy, S. Guha, C. Gkantsidis, and D. Gunawardena, "Exploring VoD in P2P swarming systems," in *IEEE Proc. Intern. Conf. Computer Comm.*, Anchorage, AK, Jan. 2007, pp. 2571 – 2575.

[19] C. Zheng, G. Shen, and S. Li, "Distributed prefetching scheme for random seek support in peer-to-peer streaming applications," in *Proc. ACM Intern. Conf. on Multimedia, P2PMMS Workshop*, Singapore, Nov. 2005, pp. 29 – 38.

[20] K. M. Ho, W. F. Poon, and K. T. Lo, "Video-on-demand systems with cooperative clients in multicast environment," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 19, no. 3, pp. 361 – 373, Mar. 2009.

[21] Weihui, J. Xu, and C. Qing, "Dynamic multicast tree to support time-shifting in real-time streaming," in *International Conference on Intelligent Computing and Cognitive Informatics*, Los Alamitos, CA, USA, 2010, pp. 251–254.

[22] X. Hou and R. Ding, "Research of providing live and time-shifting function for structured P2P streaming system," in *Proceedings of the 2010 International Conference on Machine Vision and Human-machine Interface*, Washington, DC, USA, 2010, pp. 239–242.

[23] S. Deshpande and J. Noh, "P2TSS: Time-shifted and live streaming of video in peer-to-peer systems," in *Proc. of IEEE International Conference on Multimedia and Expo*, 2008, pp. 649–652.

[24] J. Noh, P. Baccichet, F. Hartung, A. Mavlankar, and B. Girod, "Stanford Peer-to-Peer Multicast (SPPM) – overview and recent extensions," *Proc. of International Picture Coding Symposium (PCS), Chicago, Illinois, USA, invited paper,*, May 2009.

[25] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai, "Distributing streaming media content using cooperative networking," in *Proc. of ACM NOSSDAV, Miami Beach, FL*, Miami, FL, May 2002, pp. 177 – 186.

[26] Y. Chu, S. Gao, and H. Zhang, "A case for end system multicast," in *Proc. ACM Sigmetrics*, Santa Clara, CA, 2000, pp. 1 – 12.

[27] J. Noh, P. Baccichet, and B. Girod, "Experiences with a large-scale deployment of Stanford Peer-to-Peer Multicast." Seattle, WA: IEEE Proc. Intern. Packet Video Workshop, May 2009, pp. 1 – 9.

[28] E. Setton, J. Noh, and B. Girod, "Congestion-distortion optimized peer-to-peer video streaming," *International Conference on Image Processing (ICIP), Atlanta, Georgia*, pp. 721 – 724, Oct. 2006.

[29] ——, "Low-latency video streaming over peer-to-peer networks," *IEEE Intern. Conf. Multimedia and Expo*, pp. 569 – 572, Jul. 2006.

[30] J. Noh, A. Mavlankar, P. Baccichet, and B. Girod, "Time-shifted streaming in a peer-to-peer video multicast system," in *Proc. of IEEE Global Communications Conference (GLOBECOM 2009)*, Honolulu, Hawaii, USA, Nov./Dec 2009.

[31] B. Chang, L. Dai, Y. Cui, and Y. Xue, "On Feasibility of P2P on-demand streaming via empirical VoD user behavior analysis," in *Proc. Intern. Conf. Distributed Computing Systems Workshops*, Beijing, China, 2008, pp. 7–11.

[32] Y. Chu, A. Ganjam, T. Ng, S. Rao, K. Sripanidkulchai, J. Zhan, and H. Zhang, "Early experience with an Internet broadcast system based on overlay multicast," in *USENIX Annual Technical Conference*, June 2004, pp. 347 – 356.
[33] B. Li, S. Xie, G. Y. Keung, J. Liu, I. Stoica, H. Zhang, and X. Zhang, "An empirical study of the Coolstreaming+ system," *IEEE Journ. on Selected Areas on Comm.*, vol. 25, no. 9, pp. 1627–1639, Dec. 2007.
[34] S. Ross, *Stochastic processes*, 2nd ed., 1996.
[35] P. Baccichet, J. Noh, E. Setton, and B. Girod, "Content-aware P2P video streaming with low latency," in *IEEE Proc. Intern. Conf. Multimedia and Expo*, Beijing, China, 2007, pp. 400 – 403.
[36] "The Network Simulator - NS-2," http://www.isi.edu/nsnam/ns, seen on June 10, 2010.

Jeonghun Noh received an M.S. degree in Electrical Engineering from Seoul National University, Korea, in 1999, and a Ph.D. in Electrical Engineering from Stanford University, CA, in 2010, respectively.

He is currently a senior systems engineer at ASSIA inc., Redwood City, CA, USA. His research interests include low-latency data distribution, peer-to-peer streaming, mobile video streaming, and broadband access networks. He has authored or co-authored over 17 conference papers and one journal paper, and holds two US patents granted and seven US patents under review. He worked as a summer research intern at Sharp Laboratories of America, Camas, WA, in 2006 and 2007. He worked as a consultant at Dyyno Inc., Palo Alto, CA, in 2008. He collaborated with Deutsche Telekom Laboratories in Los Altos, CA, and in Berlin, Germany, in 2008-2010. He is a recipient of the ACM Mobimedia Best Student Paper Award in 2009 and the IEEE CCNC Best Presentation Award in 2008. Dr. Noh is a member of IEEE, IEEE ComSoc, and IEEE MMTC.

Bernd Girod is Professor of Electrical Engineering and (by courtesy) Computer Science in the Information Systems Laboratory of Stanford University, California, since 1999. Previously, he was a Professor in the Electrical Engineering Department of the University of Erlangen-Nuremberg. His current research interests are in the areas of video compression, networked media systems, and image-based retrieval. He has published over 450 conference and journal papers, as well as 5 books, receiving the EURASIP Signal Processing Best Paper Award in 2002, the IEEE Multimedia Communication Best Paper Award in 2007, the EURASIP Image Communication Best Paper Award in 2008, as well as the the EURASIP Technical Achievement Award in 2004. As an entrepreneur, Professor Girod has been involved with several startup ventures, among them Polycom, Vivo Software, 8x8, and RealNetworks.

He received an Engineering Doctorate from University of Hannover, Germany, and an M.S. Degree from Georgia Institute of Technology. Prof. Girod is a Fellow of the IEEE, a EURASIP Fellow, and a member of the German National Academy of Sciences (Leopoldina).

Network friendly transmission control for progressive download over TCP

Hiroyuki Hisamatsu
Department of Computer Science, Osaka Electro–Communication University, Shijonawate, Japan
Email: hisamatu@isc.osakac.ac.jp
Go Hasegawa
Cybermedia Center, Osaka University, Toyonaka, Japan
Email: hasegawa@cmc.osaka-u.ac.jp
Masayuki Murata
Graduate school of Information Science and Technology, Osaka University, Suita, Japan
Email: murata@ist.osaka-u.ac.jp

Abstract— **Video streaming services using Transmission Control Protocol (TCP) as a transport layer protocol—represented by YouTube—are becoming increasingly popular and, accordingly, have come to account for a significant portion of Internet traffic. TCP is greedy; that is, it tries to exhaust the entire bandwidth. Thus, video streaming over TCP tends to unnecessarily take bandwidth from competing traffic.**

In this paper, we first investigate the data transfer mechanisms of the current video streaming services using TCP and show that they perform data transfer at much higher rates than the video playback rate. We then propose a new transfer mechanism for video streaming over TCP, one that controls the data transfer rate based on the network congestion level and the amount of buffered video data at the receiver. Simulation results show that the proposed mechanism has two characteristics lacked by current video streaming over TCP, specifically (1) a low frequency of buffer underflow at the receiver and (2) a lack of excessive bandwidth "stealing" from competing traffic.

Index Terms— **Transmission Control Protocol (TCP), Congestion Control, Video Streaming, YouTube**

I. INTRODUCTION

With the rapid increase in network bandwidth, video streaming services have gained popularity in recent years. A number of these services utilize User Datagram Protocol (UDP) as a transport layer protocol. Also, such leading video players as Windows Media Player [1] and Real Player [2] utilize UDP if it is available. Because UDP does not conduct congestion control or retransmit packets discarded in the network, UDP-based applications are able to adjust their data transfer rate. However, UDP-based communications are often intercepted by firewalls and/or Network Address Translations (NATs), creating environments where such video streaming services cannot be offered. Transmission Control Protocol (TCP), on the other hand, can easily bypass firewalls and NATs. For this reason, most of the current video streaming services, such as YouTube [3] and nicovideo [4], support TCP-based streaming.

It has been pointed out, however, that TCP is not suitable for video streaming due to its congestion control [5]. When TCP detects a packet loss, it markedly shrinks the congestion window size, causing the TCP transfer rate to decrease significantly. Since constant-rate data transmission is desirable for video streaming, this behavior is not suitable for such a service. Furthermore, TCP, because of its greedy nature, tries to exhaust the entire bandwidth when the network bandwidth is larger than the video playback rate. Thus, video streaming over TCP has the problem that TCP increases its transfer rate regardless of the video playback rate and unnecessarily takes the bandwidth from other competing traffic.

In this paper, we address the problem of video streaming over TCP. First, we investigate the characteristics of the data transfer mechanisms of current video streaming services using TCP and show that they perform data transfer at a much higher rate than the video playback rate. We then propose an application-based data transfer mechanism that resolves this issue. Our proposed mechanism acquires TCP state variables to estimate the level of network congestion and the buffered video data size at the receiver, and then controls data transfer at an application layer according to the network congestion level and the buffered video data size at the receiver. By simulation experiments, we show that the proposed mechanism (1) decreases the frequency of buffer underflow at the receiver and (2) avoids excessively taking bandwidth from competing traffic. Note that the current mechanism of TCP video streaming fails to achieve either.

Much research has been conducted on new transport layer protocols for video streaming [6]–[12]. For instance, transport layer protocols proposed in [6]–[8] do not, relative to TCP, rapidly increase/decrease the congestion window size, yet remain TCP-friendly. In [9], the authors proposed a new transport layer protocols that conducts the TCP-friendly rate control. These protocols [6]–[9],

Manuscript received February 15, 2011; revised June 15, 2011; revised October 15, 2011; accepted December 9, 2011.

doi:10.4304/jcm.7.3.213-221

however, still do not tackle the issue in video streaming over TCP, caused by greedy nature of TCP.

Moreover, in [10]–[12], the authors propose modifications to TCP to maintain the data transfer rate requested by upper-layer applications. For instance, in [11], [12], the authors propose mechanisms to stabilize TCP throughput by concealing packet loss from TCP by installing a Forward Error Correction (FEC) layer in the lower part of a transport layer at both the sender and receiver. However, the redundancy of FEC causes these mechanisms to transmit excessive traffic to the network.

Also presented are application layer protocols for media streaming [13], [14]. Real Time Streaming Protocol (RTSP) [13] is used by Windows Media Player and Real Player, and generally uses Real-time Transport Protocol (RTP) [15] as its lower layer protocol. RTP is used for real-time applications, and although usage over TCP is supported, it almost always utilizes UDP for its transport layer protocol. In addition to unicast/multicast network services, RTSP provides operations, such as playback, pause, and record, for audio or video streaming clients.

Real Time Messaging Protocol (RTMP) [14] is a proprietary protocol of Adobe Systems and is not wholly open. While, like RTSP, RTMP is independent of its underlying protocol, it uses TCP as its transport layer protocol. It provides operations such as playback, pause, and seek for clients and splits the message into chunks and transfers them. Chunk size is determined by the server according to the data size requested by the clients. Currently, both RTSP and RTMP only offer operations for viewing and listening to streamed media; they do not directly control the data transfer, instead relying upon their lower protocols. Thus, they do not resolve the problem of streaming traffic unnecessarily taking away bandwidth from competing traffic. As mentioned above, although there are a great number of studies on video streaming, to the best of our knowledge, there is currently no study that address the issue of having too much throughput unnecessarily in video streaming over TCP.

The reminder of this paper is organized as follows. First, in Section II, we investigate the characteristics of the data transfer mechanisms of current video streaming services using TCP. In Section III, we explain our data transfer control mechanism. We then evaluate the performance of our proposed mechanism in Section IV. Finally, in Section V, we present a conclusion and discuss future works.

II. Investigation of current Video Streaming Mechanisms using TCP

In this section, we investigate the currently utilized video streaming services using TCP and clarify the features and problems of their data transfer mechanisms. Specifically, we examined YouTube and nicovideo.

In the investigation, we observed data transfer in video streaming at a packet level using `tcpdump` at a receiver. The receiver is located at Osaka Electro-Communication University in Shijonawate-shi, Osaka, Japan. We acquired video sequences from servers thought to be in Japan (although both YouTube and nicovideo do not disclose the location of their servers, we conclude that the servers investigated here are located in Japan from the results of `traceroute`). We measured 10 video sequences on YouTube, and nicovideo, respectively, and ran five measurements for each video sequence. The playback time of the YouTube video sequences was 10 [m] and the quality was 1080p (the playback rate was about 3.60 [Mbit/s]). The playback time of the nicovideo video sequences was about 10 [m] and the playback rate was about 500 [Kbit/s].

We disabled the TCP delayed Acknowledgement (ACK) option since we wanted to observe typical TCP behaviors. Moreover, we configured the advertising window size, which is the buffer size of the receiver TCP, to 224 [KByte], a size sufficiently large to avoid throughput limitations.

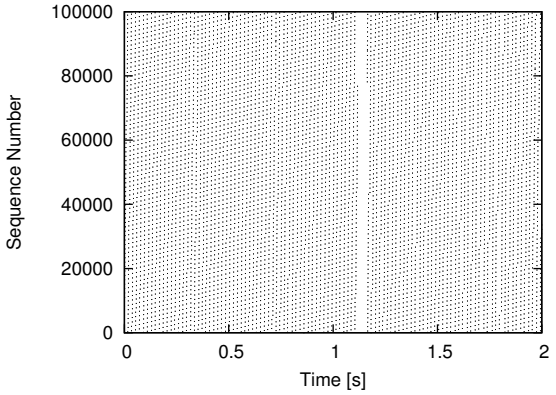
A. YouTube

From the observation results, we found that YouTube utilizes two mechanisms for video data transfer. Moreover, by conducting five time experiments against the same video sequence from the same server, we could not determine the detailed conditions for selecting one of those two mechanisms. We found that the data transfer behavior of YouTube is independent of available bandwidth between the server and the receiver. We confirmed these characteristics by experiments under various bandwidth limitations on the link connected to the receiver. We refer to these two mechanisms below as mechanism(i) and mechanism(ii), respectively.

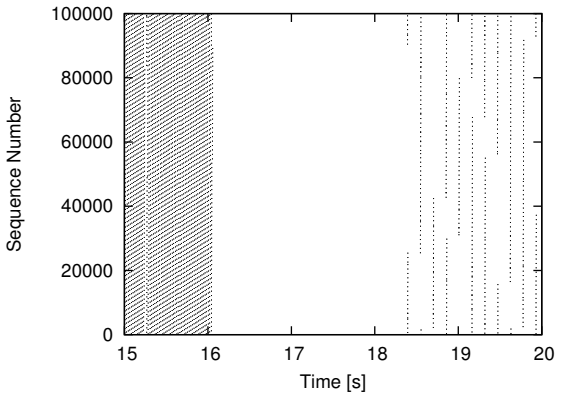
Mechanism(i) has two phases: a first and a second phase. Figure 1(a)–(c) shows the receiving timing of data packets in the first phase, at phase transition, and in the second phase, respectively. The y axis of the graphs represents the byte-count sequence number (mod 100,000) of received TCP data packets. The x axis represents the time, which is zero when the first data packet is received.

In mechanism(i), a server transmits roughly 80 [MByte] data in the first phase, whereupon it interrupts data transfer. The average transfer rate in the first phase is about 43.4 [Mbit/s], which is excessively high compared to the video playback rate. Some of the short interruptions in packet transmission apparent in Fig. 1(a) are not caused by packet losses; instead, the server may stop packet generation for some reasons. We believe that YouTube conducts such a greedy, high-rate data transmission at the beginning of the transfer to buffer sufficient amount of video data at the receiver.

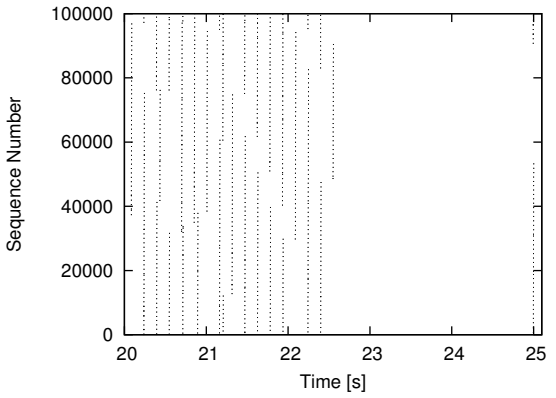
Several seconds after the first phase has finished, YouTube shifts to the second phase. In the second phase, YouTube transmits data and then switches over to discontinuous data spurts that continue until finishing the data transfer. The server sends between 32 [KByte] and 128 [KByte] of data within one round-trip time duration. Then, once the server sends a total of about 2.20 [MByte] data, it interrupts the data transfer for several seconds. The



(a) First phase



(b) Phase transmission



(c) Second phase

Figure 1: YouTube data packet receiving: mechanism(i)

average transmission rate at the time of data transfer in the second phase was about 6.13 [Mbit/s], which is also larger than the video playback rate.

With regards to mechanism(i), we find that YouTube transmits data at a higher rate than the video playback rate in both the first and the second phase. The reason for the decreased transfer rate in the second phase, we presume, is that the receiver is thought to have buffered enough video data in the first phase. Moreover, throughout experiments at different bandwidth limitations, we found that the data transfer size and rate are largely independent

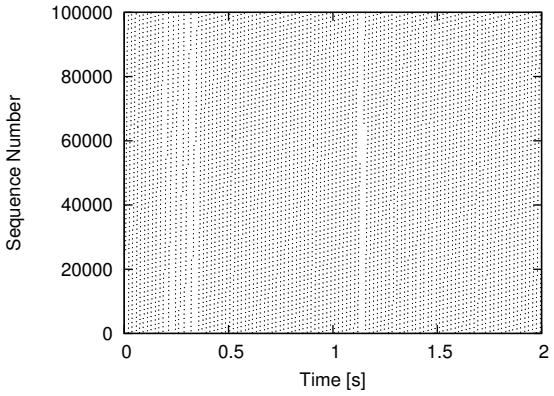


Figure 2: YouTube data packet receiving: mechanism(ii)

of the network bandwidth. We believe that the server dose not "care" the network status but rather the such differences in transfer sizes/rates are attributable to server conditions, such as CPU load, and number of TCP connections to connect.

Next, we explain mechanism(ii) of YouTube. In mechanism(ii) of YouTube, unlike in mechanism(i) of YouTube, we found that there is no special control. The server sends video data at an extremely high rate from the beginning to the end of the video sequence. The average data transfer rate is about 45.1 [Mbit/s], which is much higher than the video playback rate. From these observations, it is apparent that the data transfer mechanism for transmits video data at a rate far beyond what is necessary.

B. nicovideo

The video data transfer mechanism of nicovideo is differentiated by level of user privileges. Figure 3 shows the receiving timing of data packets for normal users. In data transfer for normal users, the server first transmits the 512 [KByte] of data at the beginning. It then transmits the 32 [KByte] chunks of data until finishing the data transfer. The average transfer rate for normal users is 583.7 [Kbit/s], which is slightly larger than the video playback rate. Moreover, the measurements under different bandwidths at the receiver reveal that for service for normal users, the data size transmitted at the beginning and chunks that follow is constant within each.

On the contrary, for premium users, the server sends video data at an extremely high rate from the beginning to the end of the video sequence. The average transfer rate for premium users is 52.0 [Mbit/s], which is 100-times larger than the video playback rate. From these observations, it is apparent that the data transfer mechanism for premium users transmits video data at a rate far beyond what is necessary. From the experiments with changing bandwidth limitation at the receiver, we also found that the data transfer behavior of nicovideo is independent of the available bandwidth between the sender and the receiver.

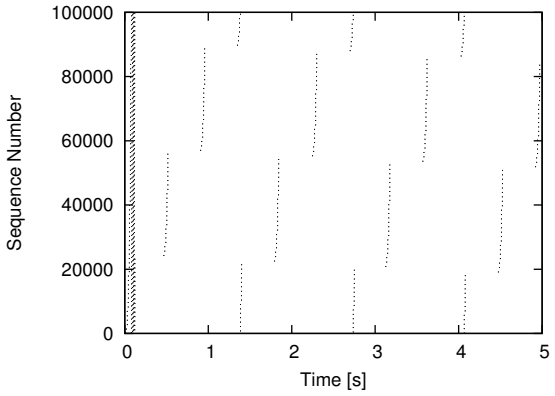


Figure 3: nicovideo data packet receiving: normal users

C. Summary of results

We found that both of YouTube and nicovideo (premium users) transfer video data at the extremely high rate relative to the video playback rate. This is good from the viewpoint of enabling continuous playback. However, it is a serious problem from the viewpoint of needless competition with other applications.

It is more desirable to transmit a small chunk of data for every interval, such as the case with transmissions for nicovideo normal users. However, optimal chunk size and chunk transmitting interval are strongly dependent on the network. Video streaming with such fixed parameters may overflow/underflow network capacity and produce failures in video playback at the receiver. Actually, the playback did stop frequently during our measurements under nicovideo normal users conditions.

III. Non bandwidth intrusive video streaming over TCP

In this section, we propose a new data transfer mechanism for video streaming over TCP that transfers video data at the minimum required rate and dose not unnecessarily deprive the other competing traffic of bandwidth. It does this by transmitting a sufficient amount of data such that buffer underflows do not occur at the receiver.

A. Outline of our Data Transfer Mechanism

The proposed mechanism is assumed to be installed as an application program at the sender and receiver. Moreover, the sender-side application requires the mechanism to acquire TCP state variables from a TCP connection transmitting video data. It is easily possible to acquire TCP state variables by using web100 [16] in the case of Linux or interactive TCP [17] for FreeBSD. Note that we do not modify any TCP operations, including the congestion control mechanism. We can expect that service providers would accept this modification. Moreover, since our proposed mechanism based on the end-to-end mechanism, we do not need to introduce it to all servers at once, but we can introduce it in a stepwise manner. Therefore, we believe that deployment of our mechanism for the large scale service such as YouTube is not so difficult.

TABLE I.
TCP state variables to acquire

Time	Variables
once in one RTT	congestion window size TCP slow-start threshold smoothed RTT retransmission timer value maximum segment size
immediately	occurrence of a packet loss packet loss detection method

The receiver-side application notifies the sender-side application of the amount of buffered video data (b_{dst}) per one round-trip time (RTT) using the TCP connection for data transfer. The sender-side application, while considering the network congestion level and the buffered video data size at the receiver, calculates the application-level window size ($apwnd$) and the amount of video data to send in the next RTT to avoid buffer underflow and playback interruption. Control of the proposed mechanism operates in the unit of one RTT.

B. TCP State Variables to acquire

In the proposed mechanism, the sender-side application acquires the TCP state variables to estimate the network congestion level. Specifically, the sender application acquires the current congestion window size (cwnd), smoothed RTT (srtt), retransmission timer value (RTO), slow-start threshold (ssthresh), and maximum segment size (MSS). These variables are obtained at every RTT. In addition, when a packet loss is detected at TCP, the sender-side application is immediately informed of the occurrence of the packet loss and its detection method: reception of three duplicate ACKs (TD-ACKs) and occurrence of a timeout (TO). We summarize the TCP state variables that the sender-side application acquires in Tab. I.

C. Congestion Level Estimation

We estimate the number of packets queued throughout the network as an index of the network congestion level. Henceforth, we use the network congestion level in the sense of the number of packets queued over the entire network. Therefore, the unit of congestion level is a packet. Based on the mechanism in TCP Vegas [18], the network congestion level $cl(i)$ in ith RTT is calculated as

$$cl(i) = \frac{baseRTT(i-1)}{MSS} \times \left(\frac{apwnd(i-1)}{baseRTT(i-1)} - \frac{apwnd(i-1)}{RTT(i-1)} \right) \quad (1)$$

where $apwnd(i-1)$, $baseRTT(i-1)$, $RTT(i-1)$ and MSS are the application-level window size of the proposed mechanism, the minimum srtt, the srtt, and the maximum TCP segment size, respectively. $apwnd(i-1)$ is the variable of our mechanism and our mechanism

knows its value. $baseRTT(i-1)$, $RTT(i-1)$, and MSS are informed by TCP.

D. Target Value of Video Data to be buffered

We first introduce $b_{ret}(i)$, $b_{dup}(i)$ and $b_{TO}(i)$. Here, $b_{ret}(i)$, $b_{dup}(i)$ and $b_{TO}(i)$ are the amount of video data to be buffered to avoid buffer underflow when one packet loss is detected by TD-ACKs, that when two or more packet losses are detected by TD-ACKs, and that when packet losses are detected by a TO, respectively.

$b_{ret}(i)$ is the amount of video data buffered at the receiver to avoid buffer underflow when a packet loss occurs, it is detected by TD-ACKs, and no additional packet loss occurs until the cwnd recovers to catch up with the video rate. Using the TCP congestion window size $cwnd(i-1)$ and the video playback rate $rate$, both are informed by TCP, $b_{ret}(i)$ in ith RTT is given by

$$b_{ret}(i) = (3 + \max(nwnd(i-1) - \lfloor cwnd(i-1)/2 \rfloor, 0)) \times RTT_{max}(i-1) \cdot rate - MSS \sum_{k=\lfloor cwnd(i-1)\ 2 \rfloor}^{nwnd(i-1)} k \quad (2)$$

where $nwnd(i)$ is the congestion window size to achieve the video playback rate. $nwnd(i)$ is given by

$$nwnd(i) = RTT_{max}(i) \cdot rate/MSS \quad (3)$$

$RTT_{max}(i)$ is given by

$$RTT_{max}(i) = \overline{RTT}(i) + \gamma RTT_{std}(i) \quad (4)$$

where $\overline{RTT}(i)$ is the exponential weighted moving average of the RTT with weight β, and $RTT_{std}(i)$ is the standard deviation of the RTT. The reason for not using the current RTT is to take account of the RTT fluctuation.

$b_{dup}(i)$ is the amount of video data buffered at the receiver to avoid the buffer underflow when a packet loss occurs, it is detected by TD-ACKs, and additional packet losses occurs until the cwnd recovers to $nwnd(i)$. Taking into account the worst case, we consider a situation whereby the window size is reduced to one by occurring two or more packet losses. $b_{dup}(i)$ is given by

$$b_{dup}(i) = (3 + nwnd(i-1) - 1)\, RTT_{max}(i-1) \cdot rate - MSS \sum_{k=1}^{nwnd(i-1)} k. \quad (5)$$

$b_{TO}(i)$ is the amount of video data buffered at the receiver to avoid buffer underflow when packet losses are detected by a TO. $b_{TO}(i)$ is given by

$$b_{TO}(i) = \Big\{ RTO(i-1) + \big(4 + \lfloor log_2 ssthresh(i-1) \rfloor + nwnd(i-1) - 2^{\lfloor log_2 ssthresh(i-1) \rfloor}\big) RTT_{max}(i-1) \Big\} rate - \Big(\sum_{k=0}^{\lfloor log_2 ssthresh(i-1) \rfloor} 2^k + \sum_{k=2^{\lfloor log_2 ssthresh(i-1) \rfloor}}^{nwnd(i-1)} k \Big) MSS \quad (6)$$

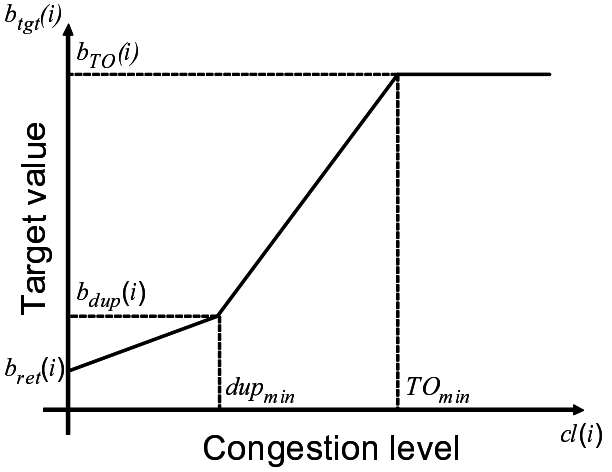


Figure 4: Calculation of the target values for video data

where $RTO(i-1)$ is the retransmission timer value and $ssthresh(i-1)$ is the threshold value for TCP slow-start, both are informed by TCP.

We then introduce dup_{min} and TO_{min}. Here, dup_{min} and TO_{min} are the threshold values to calculate the amount of video data should be buffered to avoid receiver-side buffer underflow. When the sender-side application is notified of an occurrence of a packet loss by TCP, it records the current network congestion level (cl) and the packet loss detection method. From cls and the detection methods that were recorded, the sender-side application calculates dup_{min} and TO_{min}, with dup_{min} and TO_{min} being the lower bound of 95% confidence interval of cls when packet losses are detected by TD-ACKs, and that when packet losses are detected by a TO, respectively. We calculate dup_{min} and TO_{min} from cls of the last α [h].

A sender-side application calculates the amount of video data should be buffered to avoid receiver-side buffer underflow. In ith RTT, the amount of data that a receiver-side application should buffer in order to play back video continuously, $b_{tgt}(i)$, is given by the following function of $cl(i)$, and Fig. 4 illustrates this function.

$$b_{tgt}(i) = \begin{cases} \frac{b_{dup}(i) - b_{ret}(i)}{dup_{min}} cl(i) + b_{ret} \\ \qquad (0 \le cl(i) < dup_{min}) \\ \frac{b_{TO}(i) - b_{dup}(i)}{TO_{min} - dup_{min}} (cl(i) - dup_{min}) + b_{dup}(i) \\ \qquad (dup_{min} \le cl(i) < TO_{min}) \\ b_{TO}(i) \qquad (TO_{min} \le cl(i)) \end{cases} \quad (7)$$

When $cl(i)$ satisfies $0 \le cl(i) < dup_{min}$, we assume that network packet losses do not happen very often. In this case, in order to avoid buffer underflow, the proposed mechanism buffers video data in preparation for a packet loss. As $cl(i)$ increases, the network is considered to be in an increasingly dangerous state; packet losses more serious than a single packet loss are considered likely to occur. Here, we simply increase the amount of video data

to be buffered linearly to $cl(i)$.

When $cl(i)$ satisfies $dup_{min} \leq cl(i) < TO_{min}$, the network is considered to be in a congestion level at which packet loss detection by TD-ACKs may occur. In this case, the proposed mechanism buffers video data in preparation for a packet loss detected by TD-ACKs and additional packet losses until the cwnd recovers to $nwnd(i)$. Moreover, the amount of video data to be buffered increases linearly to $cl(i)$, similarly to the case in which $0 \leq cl(i) < dup_{min}$ is satisfied.

When $cl(i)$ satisfies $TO_{min} \leq cl(i)$, we assume that the network is at a serious congestion level where a TO may happen. Here, we believe that enough video data should be buffered to avoid a buffer underflow even should a TO occur. However, we do not target a network with a congestion status so severe that packet losses occur repeatedly after a TO until the cwnd recovers $nwnd(i)$. Here, we resign ourselves to the observation that video streaming of the video playback rate within such a network simply cannot be performed at a rate specified by the application.

Since we cannot estimate the network congestion level until the first packet loss occurs, the target value $b_{tgt}(i)$ is set equal to $b_{TO}(i)$ until the first packet loss occurs. We also configure the initial value of dup_{min} and TO_{min} to be sufficiently large.

E. Controlling Transfer Rate

The proposed mechanism controls data transfer in a sender-side application. This is performed by adjusting the amount of video data $apwnd(i)$ passed from an application to TCP in one RTT. By using $b_{dst}(i)$ and Eqs. (4)(7), $apwnd(i)$ is given by

$$apwnd(i) = \min\left(\max(RTT_{max}(i-1) \cdot rate + b_{tgt}(i) - b_{dst}(i), MSS), 2apwnd(i-1)\right). \quad (8)$$

The proposed mechanism determines the amount of data passed to TCP in one RTT based on the difference between $b_{tgt}(i)$, the target value of video data to be buffered, and $b_{dst}(i)$, the amount of buffered video data at the receiver. The minimum of $apwnd(i)$ accords with TCP minimum window size. It also acts to prevent resetting of the TCP congestion window. Furthermore, in order to keep from increasing the transfer rate too rapidly, the maximum value is set at the twice of $apwnd(i-1)$, which is based on TCP slow-start phase. We summarize the notations used in this section in Tab. II. In the table, $x(i)$ means the value of x in the ith RTT.

IV. Performance Evaluation by Simulation Experiments

Using ns-2 simulator [19], we ran extensive simulation experiments at the packet level to confirm the effectiveness of our proposed mechanism. Figure 5 shows the network model for the experiments, where five TCP connections for video streaming, TCP connections for File Transfer Protocol (FTP), and one UDP flow share the single bottleneck link having a capacity of 100 [Mbit/s]. The arrival of UDP packets follows a Poisson process, with an average arrival rate of 10 [Mbit/s]. For comparison purposes, we utilized two kinds of YouTube-like transfer mechanisms, to which we refer below as YouTube-like(i) and YouTube-like(ii). Both of YouTube-like(i) and YouTube-like(ii) operate based on measurement results like those described in Section II. YouTube-like(i) has two phases such as in mechanism(i) in Subsection II. In the first phase, 80.0 [MByte] of video data is transferred. Then, YouTube-like(i) stops sending for 2.0 [s], after which it enters the second phase. It then sends video data in on-off fashion, where transmission of 2.2 [MByte] data (it is divided into chunks of 128 [KByte] and one chunk is transmitted in one RTT) followed by a 1.5 [s] pause are repeated. In contrast, YouTube-like(ii) transfers video data without any control in an application layer. In simulation experiments, we use the following parameters for the proposed mechanism: $\alpha = 1$ [h], $\beta = 0.2$, and $\gamma = 1$.

In simulation experiments, the UDP flow and the TCP connection(s) for FTP begin data transfer when a simulation starts to create background traffic and competing traffic in the network. At 30 [s], TCP video streaming connections start data transfer. The playout delay of video

TABLE II.
Definition of Notations

$cwnd(i)$	congestion window size
$apwnd(i)$	application-level window size
$nwnd(i)$	congestion window size to achieve a video playback rate
$ssthresh$	threshold value for TCP slow-start
$cl(i)$	congestion level
dup_{min}	lower bound of 95% confidence interval of a congestion level when packet losses are detected by TD-ACKs
TO_{min}	lower bound of 95% confidence interval of a congestion level when packet losses are detected by a TO
$b_{dst}(i)$	amount of buffered video data at a receiver
$b_{tgt}(i)$	target value of video data to be buffered
$b_{ret}(i)$	amount of video data to be buffered to avoid buffer underflow by single TD-ACKs
$b_{dup}(i)$	amount of video data to be buffered to avoid buffer underflow by double or more TD-ACKs
$b_{TO}(i)$	amount of video data to be buffered to avoid buffer underflow by a TO
$RTT(i)$	smoothed RTT
$baseRTT(i)$	minimum smoothed RTT
$RTT_{max}(i)$	maximum RTT
$RTO(i)$	retransmission timer value

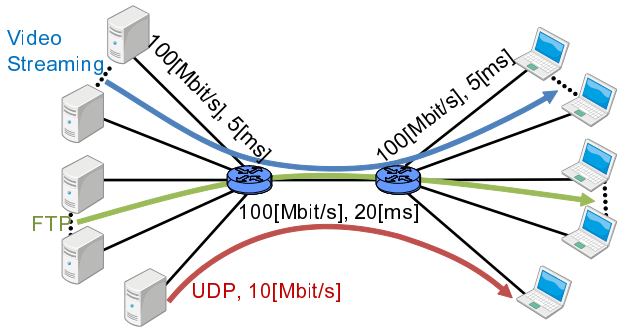


Figure 5: Simulation model

streaming at the receiver is 5 [s] and the packet size is 1500 [Byte]. The simulation finishes when the transfer of all video data is completed. We use the simulation result (excluding the first 30 [s] of the simulation time) to calculate such performance metrics as average throughput, average packet loss probability and average underflow time experienced by the receiver-side application. The video sequence has 270 [MByte], the playback rate is 3.6 [Mbit/s], and the playback time is 600 [s], which is equivalent to YouTube's 1080p video.

Figure 6 exhibits simulation results as a function of the number of TCP connections. Figures 6(a)–(d) show (a) the average throughput of TCP connections for video streaming, (b) the average total buffer underflow time per video streaming connection, (c) the average packet loss probability in the network during the experiments, and (d) the average total throughput of TCP connections for FTP, respectively. In Fig. 6(a), we see that the YouTube-like(ii) transfer rate is much higher than the video playback rate, while the transfer rates under the proposed mechanism and under YouTube-like(i) are almost equal rate to the video playback rate. Note that it is not necessary to transfer video data at a much higher rate than the video playback rate (as in the case with YouTube-like(ii)) to maintain continuous video playback.

Figure 6(b) shows that buffer underflow does not occur often under the proposed mechanism and YouTube-like(ii). On the other hand, under YouTube-like(i), we found that as the number of TCP connections for FTP increases, the buffer underflow time becomes large. This is because YouTube-like(i) controls video data transfer independently of network status. In this case, YouTube-like(i) dose not obtain sufficient throughput and so buffer underflow occurs when YouTube-like(i) competes with other TCP connections.

We next focus on the packet loss probability in the network. Figure 6(c) shows that the packet loss probability of the proposed mechanism is lower than that of YouTube-like(ii). This is because YouTube-like(ii) performs video data transfer without any controls. Thus, TCP tries to exhaust the entire bandwidth due to its greedy nature. Therefore, the network congestion level of YouTube-like(ii) is higher than that of the proposed mechanism, which controls the data transfer rate in accordance with congestion level and video playback status at the receiver. Conversely, the packet loss probability of the proposed mechanism is higher than that of YouTube-like(i). This can be explained as follows. When the network congestion level becomes high, the proposed mechanism increases the target value for video data to be buffered at the receiver. In other words, the propose mechanism acts to increase the data transfer rate. On the other hand and as mentioned above, YouTube-like(i) transmits packets independently of the network congestion status. Consequently, the packet loss probability of the proposed mechanism is higher than that of YouTube-like(i). Note that YouTube-like(i) experiences buffer underflow for long durations, especially when network congestion is serious, as a price to pay for low packet loss probability.

Finally, we focus on the total throughput of background traffic. From Fig. 6(d), we see that the total throughput of background traffic when using the proposed mechanism is higher than that when using YouTube-like(ii). This is because YouTube-like(ii) transfers video data at a high rate regardless of the video playback rate. As the result, the video streaming connections steal the bandwidth from background traffic. On the other hand, the total throughput of background traffic when using YouTube-like(i) is higher than that when using the proposed mechanism. As explained earlier, the proposed method increases the number of packets sent to the network when network congestion gets worse. Thus, the video transfer rate becomes large, driving down the throughput of competing traffic. Throughout the simulation experiments, we find that the proposed mechanism is effective in (1) suppressing the occurrence of the buffer underflow (2) avoiding unnecessarily diversion of bandwidth background traffic. Note that the current video streaming mechanisms do not have these two characteristics.

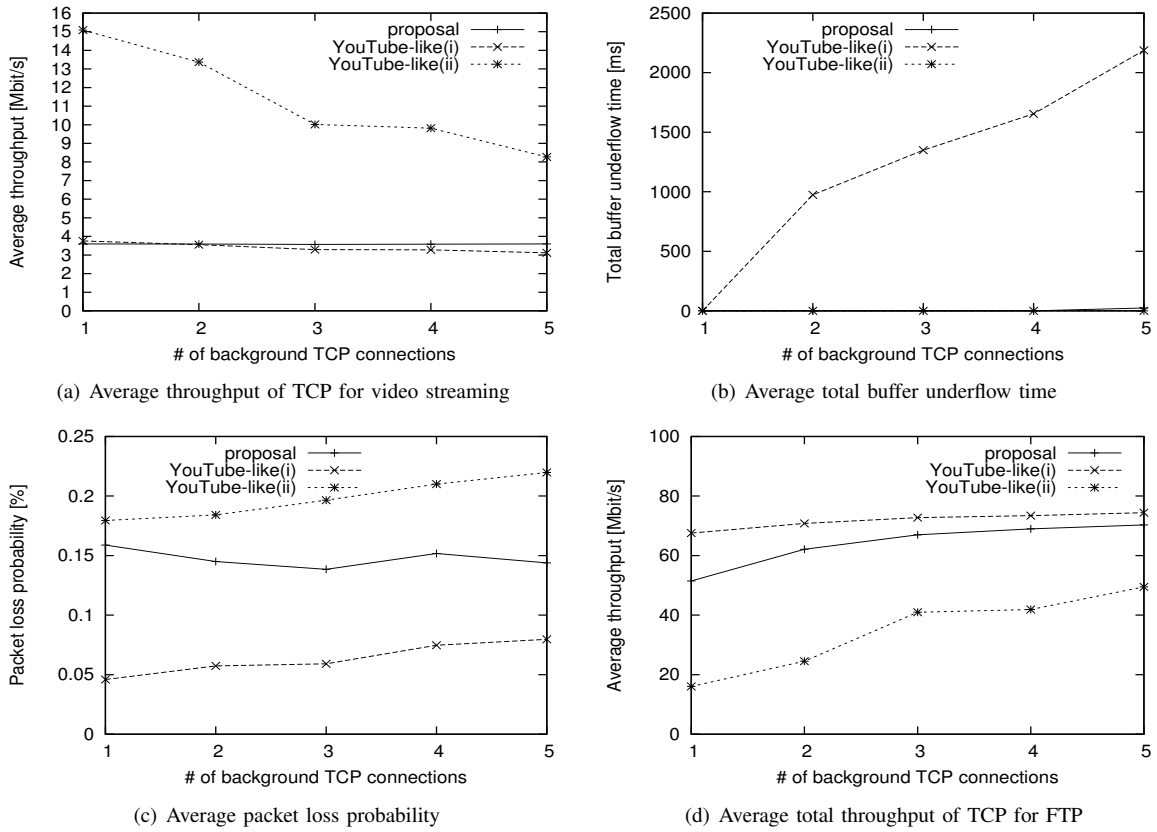
V. Conclusion and Future Works

In this paper, we have proposed a non bandwidth-intrusive video streaming mechanism and have shown its effectiveness by simulation. First, we have investigated data transfer mechanisms of the current video streaming services using TCP. Our investigation has shown that video streaming over TCP is currently performed at a much higher rate than the video playback rate. We have proposed a new data transfer mechanism to resolve this problem. The proposed mechanism transfers video data without unnecessarily taking the bandwidth from competing traffic. In simulation experiments, we have shown that proposed mechanism suppresses the occurrence of buffer underflow and dose not unnecessarily divert bandwidth from background traffic.

As future works, it is important to evaluate the performance of the proposed mechanism in a real network. We also plan to extend the proposed mechanism so that it can be operate solely by a sender-side application. Moreover, it would be interesting to consider what type of data transfer control is preferred for the network, where an increase in transfer rates for continuous video playback acts to worsen the network congestion and exacerbates video playback interruptions.

References

[1] "Microsoft Windows Media," available at http://www.microsoft.com/windows/windowsmedia/.
[2] "RealPlayer," available at http://www.real.com/.
[3] "YouTube," available at http://www.youtube.com/.
[4] "nicovideo," available at http://www.nicovideo.jp/.
[5] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications," in *Proceedings of ACM SIGCOMM 2000*, Aug. 2000, pp. 43–56.



(a) Average throughput of TCP for video streaming

(b) Average total buffer underflow time

(c) Average packet loss probability

(d) Average total throughput of TCP for FTP

Figure 6: Simulation results

[6] L. Cai, X. Shen, J. Pan, and J. Mark, "Performance analysis of TCP-friendly AIMD algorithms for multimedia applications," *IEEE/ACM Transactions on Networking*, pp. 339–355, Apr. 2005.

[7] F. Yang, Q. Zhang, W. Zhu, and Y.-Q. Zhang, "End-to-end TCP-friendly streaming protocol and bit allocation for scalable video over wireless Internet," *IEEE Journal on Selected Areas in Communication*, pp. 777–790, May 2004.

[8] S. L. Bangolae, A. P. Jayasumana, and V. Chandrasekar, "TCP-friendly congestion control mechanism for an udp-based high speed radar application and characterization of fairness," in *Proceedings of the The 8th International Conference on Communication Systems*, 2002, pp. 164–168.

[9] Y.-G. Kim, J. Kim, and C.-C. J. Kuo, "TCP-friendly Internet video with smooth and fast rate adaptation and network-aware error control," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 256–268, Feb. 2004.

[10] Y. Zhu, A. Velayutham, O. Oladeji, and R. Sivakumar, "Enhancing TCP for networks with guaranteed bandwidth services," *ACM Computer Networks: The International Journal of Computer and Telecommunications Networking*, vol. 51, no. 10, pp. 2788–2804, July 2007.

[11] T. Tsugawa, N. Fujita, T. Hama, H. Shimonishi, and T. Murase, "TCP-AFEC: An adaptive FEC code control for end-to-end bandwidth guarantee," *in Proceedings of 16th International Packet Video Workshop (PV 2007)*, pp. 294–301, 2007.

[12] B. Libæk and O. Kure, "Protecting scalable video flows from congestion loss," in *Proceedings of the 2009 Fifth International Conference on Networking and Services*, Apr. 2009, pp. 228–234.

[13] H. Schulzrinne, A. Rao, and R. Lanphier, "Real time streaming protocol (RTSP)," *Request for Comments (RFC) 2326*, Apr. 1998.

[14] "Real-time messaging protocol (RTMP) specification," available at http://www.adobe.com/devnet/rtmp.html.

[15] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," *Request for Comments (RFC) 3550*, Jan. 2003.

[16] "The web100 project," available at http://www.web100.org/.

[17] "iTCP –interactive transport protocol–," available at http://www.medianet.kent.edu/itcp/main.html/.

[18] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "TCP Vegas: New techniques for congestion detection and avoidance," in *Proceedings of ACM SIGCOMM '94*, Oct. 1994, pp. 24–35.

[19] "The network simulator – ns2," available at http://www.isi.edu/nsnam/ns/.

Hiroyuki HISAMATSU received M.E. and Ph.D. from Osaka University, Japan, in 2003 and 2006, respectively. He is currently an Associate Professor of Department of Computer Science, Osaka Electro-Communication University. His research work is in the area of performance evaluation of TCP/IP networks. He is a member of IEEE and IEICE.

Go HASEGAWA received the M.E. and D.E. degrees in Information and Computer Sciences from Osaka University,

Japan, in 1997 and 2000, respectively. From July 1997 to June 2000, he was a Research Assistant of Graduate School of Economics, Osaka University. He is now an Associate Professor of Cybermedia Center, Osaka University. His research work is in the area of transport architecture for future high-speed networks and overlay networks. He is a member of the IEEE.

Masayuki MURATA received the M.E. and D.E. degrees in Information and Computer Science from Osaka University, Japan, in 1984 and 1988, respectively. In April 1984, he joined Tokyo Research Laboratory, IBM Japan, as a Researcher. From September 1987 to January 1989, he was an Assistant Professor with Computation Center, Osaka University. In February 1989, he moved to the Department of Information and Computer Sciences, Faculty of Engineering Science, Osaka University. In April 1999, he became a Professor of Cybermedia Center, Osaka University, and is now with Graduate School of Information Science and Technology, Osaka University since April 2004. He has more than five hundred papers of international and domestic journals and conferences. His research interests include computer communication network architecture, performance modeling and evaluation. He is a member of IEEE, ACM and IEICE. He is a chair of IEEE COMSOC Japan Chapter since 2009. Also, he is now partly working at NICT (National Institute of Information and Communications Technology) as Deputy of New-Generation Network R&D Strategic Headquarters.

Characterizing Locality-aware P2P Streaming

Jian Zhao, Chuan Wu
Department of Computer Science
The University of Hong Kong
Hong Kong, S. A. R.
Email: {jzhao, cwu}@cs.hku.hk

Abstract— **Peer-to-peer (P2P) live streaming systems have been increasingly popular and successful in today's Internet, which provide large collections of video channels to millions of users at low server costs. The large volumes of P2P streaming traffic are exceeding those incurred by BitTorrent-like file sharing applications, threatening huge traffic relay cost to the Internet service providers (ISPs). There have recently emerged proposals advocating locality-aware P2P streaming protocol design, which aim to constrain streaming traffic within ISP boundaries and to alleviate traffic relay cost to the Internet Service Providers (ISPs). Nevertheless, there is a lack of in-depth understanding on the impact of such a locality-aware design on P2P streaming performance. Taking an analytical approach, we model the relation between streaming performance and traffic locality in P2P live streaming systems, in order to acquire useful insights for designing high-performance locality-aware real-world systems. We use end-to-end streaming delays as the performance metric for live streaming, and the number of copies of the live streams imported into an ISP to evaluate the volume of inter-ISP traffic. Considering multiple ISPs at different bandwidth levels, we characterize the generic relationship between the volume of inter-ISP traffic and the streaming performance; we then analyze the traffic volume when the best streaming performance is achieved and the streaming performance when minimum inter-ISP traffic is incurred. Our models and analyses provide intriguing insights on the design of effective locality-aware peer selection protocols and server deployment strategies across multiple ISPs. We also evaluate our models and theoretical results with large-scale simulations under realistic settings.**

Index Terms— **P2P Live Streaming, Traffic Localization, Performance Modeling, End-to-End Delay**

I. Introduction

Peer-to-peer (P2P) live streaming applications have thrived in today's Internet, (*e.g.*, PPLive [1], SopCast [2], Zattoo [3]), bringing thousands of live channels to millions of users at low server cost. As compared to P2P file sharing, the mutual exchanges of streams among peers may incur persistent and more intense inter-ISP traffic at all times, due to the delivery of live content in real time. Therefore, P2P streaming traffic has increasingly become a major source incurring traffic relay cost to the ISPs [4], risking ISPs' packet filtering and rate throttling in the near future [5].

Manuscript received February 15, 2011; revised June 15, 2011; accepted October 15, 2011.

This work was supported in part by the University of Hong Kong under Small Project Funding, and the Research Grants Council of Hong Kong under RGC General Research Fund (Ref: HKU718710E).

doi:10.4304/jcm.7.3.222-231

To prevent the fate of traffic filtering, a number of locality-aware P2P streaming designs have been proposed, which connect peers to nearby neighbors in the same AS or ISP, in order to reduce inter-ISP traffic [6], [7], [8]. P4P [6] advocates collaboration between P2P applications and ISPs, where ISPs provide necessary network information for a P2P streaming application to make localized peer selection decisions. Picconi *et al.* [7] propose a two-tier mesh overlay structure, where a highly clustered primary overlay is constructed for local stream propagation and the secondary inter-cluster links are used when necessary for global stream propagation, to minimize inter-ISP traffic. The recent work of Magharei *et al.* [8] bears some similarity, where an inter-ISP scheduling algorithm over an ISP-level overlay ensures that each ISP receives all substreams of a video, and an intra-ISP scheduling scheme further delivers all substreams to all internal peers. Though locality-aware protocol designs are present, an in-depth understanding on the relationship between traffic localization and P2P streaming performance is still lacking: Will streaming performance be affected when inter-ISP traffic is cut? If so, what is the quantitative relationship between inter-ISP traffic volume and P2P streaming performance? The answers to these questions are critical for P2P streaming protocol design to achieve optimal operations on traffic localization and streaming QoS provisioning, which we seek to address with extensive theoretical analysis in this paper.

A number of work have indeed been done on theoretical analysis of P2P live streaming applications [9], [10], [11], [12], [13], [14], [15], [16]. Kumar *et al.* [9] and Liu [10] have studied the maximum sustainable streaming rate and the minimum delay bound of mesh-pull based P2P streaming protocols. Liu *et al.* [11] and Chen *et al.* [13] investigate the performance bounds for minimum server load, maximum streaming rate, and minimum tree depth under different peer selection constraints for tree-push based streaming protocols. Different chunk selection strategies are analyzed by Zhou *et al.* [14] on their impact on startup latency and streaming continuity in mesh-pull based streaming systems. Bonald *et al.* [16] compare several tree-push based streaming protocols in terms of the optimality of achieved streaming rates and delays. However, none of the above work has considered locality-aware streaming protocols.

In this paper, we analytically explore the relationship between streaming performance and inter-ISP traffic in

mesh-based P2P live streaming systems, in order to derive useful insights for designing high-performance locality-aware P2P streaming systems. We use end-to-end streaming delays as the performance metric for live streaming, and quantify the amount of inter-ISP traffic with the number of copies of the live streams imported into each ISP. Considering multiple ISPs at different bandwidth levels, we characterize the generic relationship between the volume of inter-ISP traffic and the streaming performance, as well as analyze the traffic volume when the best streaming performance is achieved and the streaming performance when minimum inter-ISP traffic is incurred. Our models and analyses provide useful insights on the design of effective locality-aware peer selection protocols and server deployment strategies across multiple ISPs, which achieve desired goals on inter-ISP traffic minimization or streaming performance optimization. We also evaluate the effectiveness of our models with large-scale empirical studies under realistic settings.

The remainder of the paper is organized as follows. We present our P2P streaming system model in Sec. II and characterize the relationship between traffic localization and streaming performance in Sec. III. We perform empirical study in Sec. IV, further discuss related work in Sec. V, and conclude the paper in Sec. VI.

II. System Model

We consider a locality-aware mesh-based P2P live streaming system, where each peer retrieves the live stream of a video channel by exchanging available chunks of the stream with its neighbors.[1] There is one tracker server, which has full information of online peers and assigns neighbors to each peer. A live stream consists of consecutive chunks of the live video, and each chunk corresponds to one unit time of playback. The streaming rate of the live stream is R, equaling its playback bitrate. There are a total number of N peers distributed in M ISPs in the system, with N_i peers in ISP i. Let u_{pi} denote the average upload bandwidth of peers in ISP i, $i = 1, \ldots, M$, given as multiples of the streaming rate R, *i.e.*, $u_{pi}R$ is the average peer upload capacity in bps. The total bandwidth of the streaming server (referred to as *server capacity* hereinafter) deployed in the system is u_s, given as multiples of the streaming rate R as well. Let u_{si} be the server capacity deployed in ISP i, with $u_{si} \geq 0, i = 1, \ldots, M$, and $\sum_{i=1}^{M} u_{si} = u_s$.

We make the following assumptions on the streaming system: The upload bandwidth at peers constitutes the bandwidth bottleneck, while the download bandwidth of each peer is not. Peers in ISP i ($i = 1, \ldots, M$) have an upload bandwidth level around the average u_{pi},[2] and the average peer upload bandwidth among the ISPs satisfy $u_{p1} \geq u_{p2} \geq \ldots \geq u_{pM}$. A peer in ISP i has C_i active neighbors, and the larger peer upload capacity is in an ISP, the more neighbors each peer in the ISP can have. Specifically, we assume the average link delay on intra-ISP links in all the ISPs is the same $t_h = \frac{C_i - 1}{u_{pi}}$ ($i = 1, \ldots, M$). Let $\tau_{j,i}$ denote the delay on inter-ISP links from ISP j to ISP i, with $\tau_{j,i} \geq t_h$. Servers deployed in ISP i directly serve at most u_{si} peers at any given time, *i.e.*, a peer that directly streams from a server can download at the rate of R. All servers distributed in different ISPs have the entire live stream, and they pump out the same chunk at the same time to peers directly connected to them.

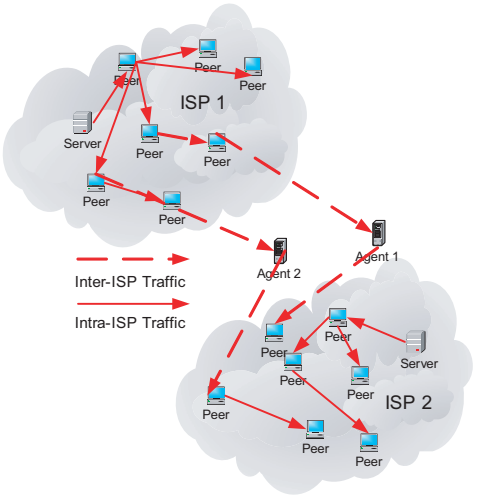


Figure 1. Agent model to characterize inter-ISP traffic.

Let $K_{j,i}$ denote the number of copies of live streams that peers in ISP i retrieve from ISP j. For better characterization of the inter-ISP traffic, we consider there are $K_{j,i}$ *virtual agents*, each downloading a copy of the live stream from ISP j and forwarding the chunks in the stream to peers in ISP i. We note that the agents are imaginary, modeled to characterize the number of live streams disseminated across ISP boundaries, and the chunks of a live stream are indeed downloaded by (possibly) different peers in ISP i from ISP j. An illustration of the agent model is given in Fig. 1. With this agent model, we introduce the simplification that an integer number of streams are downloaded from one ISP to another, while the derived insights can be readily applied to the general case with fractional streams.

The amount of inter-ISP traffic from ISP j to ISP i ($i \neq j$) is $K_{j,i}R$. In a real-world streaming system, whether a packet incurs intra- or inter-ISP traffic can be judged based on the source IP address carried in the packet. There are existing tools to tell the ISP belonging of IP addresses, *e.g.*, ASFinder in the CoralReef suite [17].

In live streaming systems, peers need to play chunks of the live stream at similar paces. When a new chunk is pumped out from the server, it should be distributed to all peers as soon as possible. This deadline driven dissemination is different from the rarest first strategy in file sharing systems. Given the time sensitivity of live

[1] We focus on one video channel in our analysis and our analytical results can be readily extended to a system with multiple channels, since each peer typically only participates in one live streaming channel.

[2] We note that this is a reasonable simplification of real-world ISPs, as an ISP typically provides the same type of access network.

TABLE I.
IMPORTANT NOTATIONS

M	number of ISPs
N	total number of peers in the system
N_i	number of peers in ISP i
R	streaming bitrate
C_i	number of active neighbors at a peer in ISP i.
u_s	overall actual server capacity
u_{si}	actual server capacity deployed in ISP i
u_{pi}	average peer upload bandwidth in ISP i
$\mathscr{U}_s$	total effective server capacity
$\mathscr{U}_{si}$	effective server capacity in ISP i
t_h	delay on intra-ISP links
$\tau_{j,i}$	delay on inter-ISP links from ISP j to ISP i
$h_{j,i}^{(l)}$	hop count at which agent l pulls a chunk from ISP j to ISP i
$\tilde{u}_{j,i}^{l}$	effective server capacity brought by agent l into ISP i from ISP j
$\bar{u}_{i,j}^{l'}$	effective server capacity branched off from ISP i to ISP j by agent l'
$K_{j,i}$	number of agents downloading streams from ISP j to ISP i
$\mathscr{H}_{j,i}$	the set of hops at which agents pull chunks from ISP j to ISP i
H_i	min. number of dissemination hops for all peers in ISP i to receive a chunk
D_i	minimum end-to-end delay for all peers in ISP i to receive a chunk

streaming, we evaluate the streaming performance in the system as the end-to-end chunk dissemination delay, *i.e.*, the time slipped from when the server pumps out a chunk to the time when all peers have received this chunk.

We summarize important notations in Table I for ease of reference.

III. CHARACTERIZING THE IMPACT: TRAFFIC LOCALIZATION VS. STREAMING PERFORMANCE

We start by giving a lemma that provides theoretical bounds for minimum-delay chunk dissemination inside one ISP, to be used in our analysis.

Lemma 1. *Consider a P2P live streaming system with N peers over an ISP, where the server capacity is u_s, upload bandwidth of each peer is u_p, and the maximum number of neighbors each peer has is C. The minimum number of dissemination hops needed for all peers to receive each chunk is*

$$H = 1 + \lceil log_C \frac{N}{u_s} \rceil.$$

The minimum end-to-end delay for all peers to receive the chunk is:

$$D = 1 + \frac{C-1}{u_p} \cdot \lceil log_C \frac{N}{u_s} \rceil. \quad (1)$$

Proof: For mesh-based P2P streaming system, Liu [10] has shown that minimum end-to-end delay is achieved when each chunk is disseminated using a snow-ball approach, which can be applied to our case as follows: In the first hop, the server sends the chunk to u_s peers; in the following hop h ($h > 1$), each of the peers who have received the chunk during the previous $h-1$ hops further distributes the chunk to $C-1$ neighbors (excluding the one she has received the chunk from), who do not own the chunk yet (while the server continues distributing new chunks to u_s peers during this process).

According to this approach, the total number of peers that the chunk reaches after the 1st hop is u_s, that after 2 hops is $u_s + u_s(C-1) = u_sC$, that after 3 hops is $u_sC + u_sC(C-1) = u_sC^2$. The total number after h hops can be derived as u_sC^{h-1}. Considering the total number of peers is N, we derive the total number of peers that have received the chunk after h hops ($h \geq 1$), counting from the time the server pumps out the chunk, is

$$min\{N, u_sC^{h-1}\}.$$

The minimum number of dissemination hops for all peers to receive the chunk, H, can be derived by letting $u_sC^{H-1} \geq N$. Therefore,

$$H = 1 + \lceil log_C \frac{N}{u_s} \rceil.$$

Based on our assumptions that each chunk corresponds to 1 unit playback time and that any peer directly downloading from the server gets a streaming rate of R, we know that the first hop of the chunk dissemination from the server takes 1 unit time. Each other hop of the chunk dissemination from a peer to another takes $\frac{1}{u_p/C-1} = \frac{C-1}{u_p}$ time units. We can then calculate the minimum delay for all peers to receive the chunk as

$$D = 1 + \frac{C-1}{u_p} \cdot \lceil log_C \frac{N}{u_s} \rceil.$$

□

We now analyze the relationship between the volume of inter-ISP traffic and the minimum end-to-end delay in a multi-ISP P2P streaming system. We first use a two-ISP case to illustrate the idea of our model, and then present our model for the general multi-ISP case.

A. Two-ISP Case: An Illustration of Modeling Methodology

For ease of illustration, we assume server capacity u_s is only deployed in ISP 1, and thus there is only inter-ISP traffic from ISP 1 to ISP 2. We also simplify notations: Let K denote the number of virtual agents who download live streams from ISP 1 to ISP 2, with corresponding inter-ISP traffic KR. The delay of links between the two ISPs is τ. Suppose agent l ($1 \leq l \leq K$) downloads a chunk (any chunk in the live stream) at hop $h^{(l)}$ (counting from the time the server pumps out the chunk) from a peer in ISP 1, who has received the chunk previously. and we assume $1 \leq h^{(1)} \leq h^{(2)} \leq ... \leq h^{(K)}$ without loss of generality. Let $\mathscr{H} = \{h^{(1)}, \ldots, h^{(K)}\}$. We next calculate the minimum end-to-end chunk dissemination delay in each ISP. The maximum between the two is the minimum chunk dissemination delay in the entire system.

ISP 1. Let H_1 be the minimum number of dissemination hops for all peers in ISP 1 to receive a chunk. To achieve minimum end-to-end dissemination delay in ISP 1, the snow-ball algorithm discussed in Lemma 1 can

be applied, which distributes a chunk to $u_s C_1^{H_1-1}$ peers within H_1 hops in the ISP, if no copies of the chunk branch off as pulled by agents into ISP 2. However, if agent l does pull a copy of the chunk from ISP 1 at hop $h^{(l)}$, there will be a reduction of $C_1^{H_1-h^{(l)}}$ peers in ISP 1 that could receive the chunk within the H_1 hops of dissemination, derived as follows: If one peer in ISP 1 received the chunk rather than agent l at hop $h^{(l)}$, it could have further distributed the chunk to C_1-1 peers in hop $h^{(l)}+1$, and then all C_1 peers could have further sent the chunk to $C_1(C_1-1)$ peers at hop $h^{(l)}+2$, and so on. The total number of peers that could have received the chunk from hop $h^{(l)}$ to hop H_1 is therefore $1+C_1-1+C_1(C_1-1)+\ldots+C_1^{H_1-h^{(l)}-1}(C_1-1)=C_1^{H_1-h^{(l)}}$.

In the case that K agents each retrieve one copy of the chunk at hop $h^{(1)},\ldots,h^{(k)}$, respectively, the number of peers in ISP 1 that can receive the chunk in H_1 hops becomes $u_s C_1^{H_1-1}-\sum_{l=1}^{K} C_1^{H_1-h^{(l)}}$. Therefore, the minimum number of hops for all peers in ISP 1 to receive the chunk in this case, H_1, can be derived by letting $u_s C_1^{H_1-1}-\sum_{h\in\mathscr{H}} C_1^{H_1-h}\geq N_1$. We derive

$$H_1 = 1+\lceil log_{C_1}\frac{N_1}{u_s-\sum_{h\in\mathscr{H}} C_1^{1-h}}\rceil.$$

Again, the first hop of chunk dissemination from the server takes 1 time unit, and each other hop in ISP 1 takes $t_h=\frac{C_1-1}{u_{p1}}$ time units. The minimum end-to-end delay for all peers to receive the chunk in ISP 1 is

$$D_1 = 1+t_h\cdot\lceil log_{C_1}\frac{N_1}{u_s-\sum_{h\in\mathscr{H}} C_1^{1-h}}\rceil. \qquad (2)$$

Comparing (2) with (1), we see that the effective server capacity used to serve peers in ISP 1 is indeed

$$\mathscr{U}_{s1} = u_s-\sum_{h\in\mathscr{H}} C_1^{1-h}.$$

ISP 2. When virtual agent l retrieves a chunk from ISP 1 at hop $h^{(l)}$, indeed a peer in ISP 2 obtains the chunk from ISP 1 at time $1+t_h\cdot(h^{(l)}-2)+\tau$ after the server pumps out the chunk (1 time unit for the server pumping out the chunk, $t_h\cdot(h^{(l)}-2)$ time units for the chunk dissemination in ISP 1, τ time units for the chunk traversing the inter-ISP link), where τ is the delay of inter-ISP links. We next model the upload bandwidth that peers in ISP 1 use to serve peers in ISP 2 as effective server capacity deployed in ISP 2, in order to analyze the end-to-end chunk dissemination delay in ISP 2 using a similar methodology as applied previously:

Let $\tilde{u}^l$ denote the effective server capacity introduced by agent l. We can assume there is an imaginary server deployed in ISP 2, with the capacity of $\sum_{l=1}^{K}\tilde{u}^l$. The chunk disseminated from the server in ISP 1 to a peer in ISP 2 using a time $1+t_h\cdot(h^{(l)}-2)+\tau$, can be equivalently considered as distributed from an imaginary server in ISP 2 to the peer after traveling $1+(h^{(l)}-2)+\frac{\tau}{t_h}$ hops. This number of hops is calculated as follows: The first hop from the imaginary server takes 1 time unit, each other hop from one peer to another peer in ISP 2 takes t_h, $t_h=\frac{C_2-1}{u_{p2}}=\frac{C_1-1}{u_{p1}}$, time units, and the time $1+t_h\cdot(h^{(l)}-2)+\tau$ is thus equivalently to $1+(h^{(l)}-2)+\frac{\tau}{t_h}$ hops in ISP 2. With imaginary server capacity $\tilde{u}^l$, after $1+(h^{(l)}-2)+\frac{\tau}{t_h}$ hops, $\tilde{u}^l\cdot C_2^{1+(h^{(l)}-2)+\frac{\tau}{t_h}-1}=1$ peer receives the chunk. We can thus derive the effective server capacity introduced by agent l as $\tilde{u}^l=C_2^{(2-h^{(l)}-\frac{\tau}{t_h})}$. Therefore, the total effective server capacity in ISP 2, introduced by K agents, is

$$\mathscr{U}_{s2}=\sum_{l=1}^{K}\tilde{u}^l=\sum_{h\in\mathscr{H}} C_2^{(2-h-\frac{\tau}{t_h})}.$$

Suppose the imaginary server in ISP 2 pumps up the chunk at the same time as the server in ISP 1 does. The end-to-end dissemination delay in ISP 2, the duration from the time the server in ISP 1 pumps out the chunk to the time all peers in ISP 2 have received the chunk, can be calculated using a similar methodology as used before: let H_2 be the maximum number of dissemination hops in ISP 2 from its imaginary server to all peers. $u_{s2}\cdot C_2^{H_2-1}$ peers in ISP 2 can have the chunk after H_2 hops. Letting $u_{s2}\cdot C_2^{H_2-1}\geq N_2$, we derive

$$H_2 = 1+\lceil log_{C_2}\frac{N_2}{\sum_{h\in\mathscr{H}} C_2^{(2-h-\frac{\tau}{t_h})}}\rceil.$$

Recall that the first hop of the chunk dissemination from the server takes 1 unit time and each other hop in ISP 2 takes $t_h=\frac{C-1}{u_{p2}}$ time units. The minimum end-to-end delay for all peers in ISP 2 to receive the chunk is

$$D_2 = 1+t_h\cdot\lceil log_{C_2}\frac{N_2}{\sum_{h\in\mathscr{H}} C_2^{(2-h-\frac{\tau}{t_h})}}\rceil. \qquad (3)$$

Therefore, the minimum chunk dissemination delay in the entire system can be derived as follows, while there are K agents that incur inter-ISP traffic KR:

$$D=\max\{D_1,D_2\}.$$

B. Multiple-ISP Case: Characterization and Analysis

We next model inter-ISP traffic and streaming performance with multiple ISPs in the system. Specifically, given the numbers of agents across ISPs (which correspond to volumes of inter-ISP traffic), we analyze the minimum end-to-end dissemination delay in each ISP, and then calculate the maximum among all as the minimum chunk dissemination delay in the entire system.

ISP i**.** Let $\mathscr{U}_{si}$ denote the effective server capacity in ISP i ($i=1,\ldots,M$), which is the sum of actual deployed capacity u_{si} and server capacity brought by agents pulling chunks into the ISP, minus the capacity branching off by agents pulling chunks out of the ISP. Let H_i be the number of hops needed for all peers in ISP i to receive any chunk in the live stream, counting from the time the server pumps out the chunk. Based on the snow ball approach in Lemma 1, we know the maximum number of peers that

can receive the chunk in H_i hops is $\mathscr{U}_{si} \cdot C_i^{H_i-1}$. Letting $\mathscr{U}_{si} \cdot C_i^{H_i-1} \geq N_i$ where N_i is the number of peers in ISP i, we derive

$$H_i = 1 + \lceil log_{C_i} N_i / \mathscr{U}_{si} \rceil.$$

Since the delay of each hop in ISP i is t_h except the first hop from servers which takes 1 time unit, the minimum end-to-end dissemination delay in ISP i (counting from the time the server pumps out the chunk) is

$$D_i = 1 + t_h \cdot \lceil log_{C_i} \frac{N_i}{\mathscr{U}_{si}} \rceil. \quad (4)$$

Let $h_{j,i}^l$ denote the hop at which agent l pulls a chunk from ISP j to ISP i. Equivalently, it means that a peer in ISP i obtains the chunk at time $1+t_h \cdot (h_{j,i}^l - 2) + \tau_{j,i}$ after the server pumps out the chunk (recall that we assume all servers in all ISPs pump out the same chunk at the same time). Similar to our analysis on ISP 2 in the two-ISP case, the chunk disseminated from the server in ISP j to a peer in ISP i using $1+t_h \cdot (h_{j,i}^l - 2) + \tau_{j,i}$ time slots, can be equivalently considered as distributed from an imaginary server in ISP i to the peer after traveling $1+(h_{j,i}^l - 2) + \frac{\tau_{j,i}}{t_h}$ hops. Let $\tilde{u}_{j,i}^l$ be the effective server capacity brought by agent l into ISP i, which can be calculated as follows: With server capacity $\tilde{u}_{j,i}^l$, after $1 + (h_{j,i}^l - 2) + \frac{\tau_{j,i}}{t_h}$ hops, $\tilde{u}_{j,i}^l \cdot C_i^{1+(h_{j,i}^l-2)+\frac{\tau_{j,i}}{t_h}-1} = 1$ peer in ISP i receives the chunk. Therefore,

$$\tilde{u}_{j,i}^l = C_i^{(2-h_{j,i}^l - \frac{\tau_{j,i}}{t_h})}.$$

On the other hand, suppose agent l' pulls a chunk from ISP i to ISP j at hop $h_{i,j}^{l'}$. This results in one fewer peer to receive the chunk in ISP i at hop $h_{i,j}^{l'}$. Let $\bar{u}_{i,j}^{l'}$ be the reduced server capacity from ISP i due to agent l' pulling out the chunk. Using $\bar{u}_{i,j}^{l'} \cdot C_i^{h_{i,j}^{l'}-1} = 1$, we derive

$$\bar{u}_{i,j}^{l'} = C_i^{1-h_{i,j}^{l'}},$$

which is a generalization of the analytical result for ISP 1 in the two-ISP case.

Let $\mathscr{H}_{j,i}$ denote the set of hops at which $K_{j,i}$ agents pull chunks into ISP i from ISP j ($j \neq i$). Let $\mathscr{H}_{i,j}$ denote the set of hops at which $K_{i,j}$ agents pull chunks out of ISP i to ISP j ($j \neq i$). The overall effective server capacity in ISP i is

$$\begin{aligned} \mathscr{U}_{si} &= u_{si} + \sum_{j=1, j\neq i}^{M} [\sum_{h \in \mathscr{H}_{j,i}} C_i^{(2-h-\frac{\tau_{j,i}}{t_h})} \quad (5) \\ &- \sum_{h \in \mathscr{H}_{i,j}} C_i^{(1-h)}], \quad 1 \leq i \leq M. \end{aligned}$$

With the above analysis, the minimum end-to-end dissemination delay in the entire system is given in the following theorem.

Theorem 1. *Consider a P2P live streaming system with N peers over M ISPs, where N_i peers are distributed in ISP i with average upload capacity u_{pi} and C_i neighbors. The delay on intra-ISP links is $t_h = \frac{C_i - 1}{u_{pi}}$ ($1 \leq i \leq M$), and the latency on inter-ISP links between ISP j and ISP i is $\tau_{j,i}$. The amount of server capacity deployed in ISP i is u_{si}. $K_{j,i}$ copies of the stream are distributed into ISP i from ISP j ($i \neq j$). The minimum end-to-end delay for all peers in the system to receive a chunk is*

$$\begin{aligned} D &= \max\{D_1, D_2, ..., D_M\}, \quad (6) \\ D_i &= 1 + t_h \cdot \lceil log_{C_i} \frac{N_i}{\mathscr{U}_{si}} \rceil, \quad 1 \leq i \leq M, \quad (7) \\ \mathscr{U}_{si} &= u_{si} + \sum_{j=1, j\neq i}^{M} [\sum_{h \in \mathscr{H}_{j,i}} C_i^{(2-h-\frac{\tau_{j,i}}{t_h})} \quad (8) \\ &- \sum_{h \in \mathscr{H}_{i,j}} C_i^{(1-h)}], \quad 1 \leq i \leq M, \end{aligned}$$

where $\mathscr{U}_{si}$ is the effective server capacity to serve peers in ISP i.The amount of inter-ISP traffic incurred is $\sum_{i=1}^{M} \sum_{j=1, j\neq i}^{M} K_{j,i} R$, and the percentage over the total amount of P2P streaming traffic is

$$F = \frac{\sum_{i=1}^{M} \sum_{j=1, j\neq i}^{M} K_{j,i} R}{N \cdot R}.$$

Theorem 1 gives the generic relationship between the volume of inter-ISP traffic and the end-to-end chunk dissemination delay in multi-ISP systems. We discuss the implications of Theorem 1 with two corollaries.

Corollary 1. *Given server capacity deployment $u_{s1}, u_{s2}, \ldots, u_{sM}$, the amount of inter-ISP traffic needed when minimum end-to-end delay is achieved in the system, can be derived by solving the following optimization problem, where $K_{j,i}$'s, $h_{j,i}^l$'s, are optimization variables:*

$$\min D$$
$$\text{Subject to: Constraints } (6) - (8).$$

We can use sequential quadratic programming (SQP) to solve this non-linearly constrained optimization problem. SQP is the most successful method for solving non-linear optimization problems [18] (with convergence properties extensively studied), which has been implemented in many packages including MATLAB.

Corollary 2. *If we require that no inter-ISP traffic should be incurred, the minimum end-to-end dissemination delay in the entire system is achieved when the deployment of overall server capacity u_s among the ISPs satisfies $log_{C_1} \frac{N_1}{u_{s1}} = \ldots = log_{C_M} \frac{N_M}{u_{sM}}$.*

Corollary 2 can be illustrated as follows: When there is no inter-ISP traffic, we have $\mathscr{U}_{si} = u_{si}$ and $\sum_{i=1}^{M} \mathscr{U}_{si} = u_s$. If the end-to-end delay in ISP i, D_i, decreases, we know the effective server capacity $\mathscr{U}_{si}$ in ISP i must have been increased, according to Eqn. (7). Meanwhile, there must exist another ISP j, whose effective server capacity $\mathscr{U}_{sj}$ decreases, and thus its end-to-end delay D_j increases. Therefore, the minimum end-to-end delay in the entire system occurs when $D = D_1 = D_2 = \ldots = D_M$, which is equivalent to $log_{C_1} \frac{N_1}{u_{s1}} = \ldots = log_{C_M} \frac{N_M}{u_{sM}}$, *i.e.*, when the inter-ISP traffic is completely blocked, the minimum delay in the system occurs when the server

capacity deployment in the ISPs satisfies the equations in Corollary 2.

We further discuss implications of our model based on the theorem and corollaries. From Eqn. (8), we derive the overall effective server capacity in the system as

$$\mathscr{U}_s = \sum_{i=1}^{M} \mathscr{U}_{si} = u_s + \sum_{i=1}^{M} \sum_{j=1, j\neq i}^{M} [\sum_{h \in \mathscr{H}_{j,i}} C_i^{(2-h-\frac{\tau_{j,i}}{t_h})} - \sum_{h \in \mathscr{H}_{i,j}} C_i^{(1-h)}].$$

Such a total effective server capacity $\mathscr{U}_s$ may not be equal to the overall deployed server capacity u_s. We illustrate this point by comparing the effective server capacity brought into ISP i, $\tilde{u}^l_{j,i} = C_i^{2-h^l_{j,i}-\frac{\tau_{j,i}}{t_h}}$, and the effective server capacity branching off from ISP j, $\bar{u}^l_{j,i} = C_j^{1-h^l_{j,i}}$, when an agent l pulls a chunk from ISP j to ISP i. The difference $\tilde{u}^l_{j,i} - \bar{u}^l_{j,i} = C_j^{1-h^l_{j,i}} \cdot [(\frac{C_i}{C_j})^{1-h^l_{j,i}} \cdot C_i^{1-\frac{\tau_{j,i}}{t_h}} - 1]$ may not be equal to 0, and its sign is decided by upload bandwidths of peers in ISPs i and j (as reflected by C_i and C_j), as well as latencies on inter-ISP links and intra-ISP links (*i.e.*, $\tau_{j,i}$ and t_h). We divide our discussions into two cases.

Case 1: Equal peer upload capacity in all ISPs, *i.e.*, $u_{p1} = u_{p2} = \ldots = u_{pM}$. We then know $C_1 = C_2 = \ldots = C_M$, $\tilde{u}^l_{j,i} - \bar{u}^l_{j,i} = C_j^{1-h^l_{j,i}}(C_i^{1-\frac{\tau_{j,i}}{t_h}} - 1) \leq 0$ (since $\tau_{j,i} \geq t_h$), and therefore $\mathscr{U}_s \leq u_s$. This gives us the following intriguing insights:

(i) When all peers have the same upload capacity, any cross-ISP chunk download will lead to decrease of the total effective server capacity, and thus increase of end-to-end chunk dissemination delay in the entire system. Therefore, the best streaming performance occurs when peers stream within their ISP boundaries (*i.e.*, minimum end-to-end dissemination delay and minimum inter-ISP traffic of zero occur concurrently), when server capacity deployment satisfies the condition in Corollary 2.

With $C_1 = \ldots = C_M$, the optimal server capacity distribution in Corollary 2 becomes $u_{si} = \frac{N_i}{N} u_s, \forall i$. It shows that no matter how peer population is distributed in different ISPs, as long as the server capacity is deployed proportionally to the number of peers in each ISP, the minimum end-to-end chunk dissemination delay, $D = 1 + t_h \cdot \lceil log_{C_i} \frac{N_i}{N} u_s \rceil$, and the minimum inter-ISP traffic, 0, occur concurrently.

Case 2: Different peer upload capacities in different ISPs. If $u_{pj} > u_{pi}$ (*i.e.*, $C_j > C_i$), we know $\frac{C_j}{C_i}^{(h^l_{j,i}-1)} \cdot C_i^{1-\frac{\tau_{j,i}}{t_h}}$ could be no smaller than 1 when $\tau_{j,i}$ is close to t_h, and then we can derive $\tilde{u}^l_{j,i} - \bar{u}^l_{j,i} = C_j^{1-h^l_{j,i}} \cdot [(\frac{C_i}{C_j})^{1-h^l_{j,i}} \cdot C_i^{1-\frac{\tau_{j,i}}{t_h}} - 1] \geq 0$. This shows us the following:

(ii) To achieve minimum end-to-end dissemination delay in the entire system, peers in ISPs with smaller upload bandwidth should try to download from ISPs with larger bandwidth if the inter-ISP link delay is small, such that the total effective server capacity can be increased. In this case, the larger the inter-ISP traffic is, the smaller the end-to-end dissemination delay in the system becomes.

(iii) When the inter-ISP link latency is large, even if peers in an ISP with small peer upload bandwidth stream from another ISP with larger peer bandwidth, $\frac{C_j}{C_i}^{(h^l_{j,i}-1)} \cdot C_i^{1-\frac{\tau_{j,i}}{t_h}}$ could still be smaller than 1, and the total effective server capacity decreases. In this situation, cross-ISP download should be discouraged, in order to achieve smaller chunk dissemination delay as well as lower inter-ISP traffic.

(iv) Server capacities are better deployed more into ISPs with large peer upload capacities and small inter-ISP link delays to other ISPs, rather than in ISPs otherwise, in order to achieve the smallest end-to-end dissemination delay in the entire system. If an ISP has a high inter-ISP latency with other ISPs, it is beneficial to deploy some server capacity in the ISP.

(v) Deploying more server capacity in ISPs with more peers decreases the volume of inter-ISP traffic, while it may simultaneously increase the end-to-end dissemination delay, when peer upload bandwidths in those ISPs are not high.

IV. Empirical Studies

We next investigate the relationship among volumes of inter-ISP traffic, streaming performance, and server capacity deployment, as captured by our models, using large-scale empirical studies under realistic settings. A discrete-event simulator is implemented, which can simulate tens of thousands of peers and multiple ISPs, as well as streaming servers deployed in different ISPs and a tracker server in the system which maintains information of chunk availability at the peers. In our default settings, we simulate $N = 10,000$ peers in the entire system, which are evenly distributed in 4 ISPs. The average peer upload capacities are 1.4 in ISP 1, 1.2 in ISP 2, 1.2 in ISP 3, and 1 in ISP 4, respectively. The 4 ISPs are all connected by peering links, so cross-ISP traffic may exist between any two ISPs. A total amount of $u_s = 100$ server capacity is deployed in the system, which is the sum of upload capacities at streaming servers in all ISPs. The tracker server provides a list of 50 candidate peers to each peer as neighbors. The active neighbors to whom a peer uploads chunks to are selected from the candidate peers, with a number of 7, 6, 6, and 5 for peers in the four ISPs, respectively. The delay on each intra-ISP link is 5 time units. All inter-ISP link delays are the same.

In order to control the numbers of copies of each chunk retrieved across ISP boundaries (*i.e.*, the amount of inter-ISP traffic), agents are simulated in our simulator (though

they do not indeed exist in a practical system), each of which downloads chunks from one ISP and forwards them to another ISP. For example, agent $A_{j,i}$ forwards copies of streaming chunks from ISP j to ISP i: when we wish to simulate the retrieval of $K_{j,i}$ copies of a chunk from ISP j to ISP i, agent $A_{j,i}$ pulls $K_{j,i}$ copies of the chunk from peers in ISP j at time units $1+t_h(h^l_{j,i}-2), 1 \leq l \leq K_{j,i}$, respectively, and for each copy, it forwards it to a peer in ISP i that needs the chunk after a delay of $\tau_{j,i} - t_h$ time units (in order to simulate the effect of inter-ISP link delay). The tracker server logs the chunk availability in each peer. Thus, the tracker server can record the time units needed for all peers to obtain a chunk after the server pumps out it, which is the chunk dissemination delay.

A. End-to-End Chunk Dissemination Delay vs. Inter-ISP Traffic

We first fix the deployment of server capacity in the four ISPs, and study the relation between percentage of inter-ISP traffic (over the total amount of streaming traffic) and end-to-end chunk dissemination delay in Fig. 2. Different amounts of inter-ISP traffic are simulated by controlling the number of copies of chunks retrieved across ISP boundaries. Three cases of inter-ISP latency are investigated, where L represents the ratio of inter-ISP link delay over intra-ISP link delay: (1) low inter-ISP link delay with $L = 1$, (2) medium inter-ISP link latency with $L = 2$, and (3) high inter-ISP link latency with $L = 5$.

We observe that when all server capacity is deployed in ISP 1 with the largest peer upload bandwidth (the case in Fig. 2(a)), the larger the inter-ISP traffic is, the lower the end-to-end dissemination delay is, which validates insight (ii) in the previous section. The largest percentage of inter-ISP traffic is 75%, since each of the four ISPs has the same number of peers.

When server capacity is deployed in all ISPs (the case in Fig. 2(b)), the development of chunk dissemination delay with the increase of inter-ISP traffic is in general not monotonic; there exists an optimal volume of inter-ISP traffic with which the minimum dissemination delay is achieved. In addition, we observe that the larger the inter-ISP link delay is, the more inter-ISP traffic is required to achieve the best streaming performance.

B. Impact of Server Capacity Deployment

We next study how server capacity deployment affects the minimum end-to-end dissemination delay in the system and the required inter-ISP traffic to achieve this delay (which can be derived by solving the optimization problem in Corollary 1). In this set of experiments, different amounts of server capacity are deployed in four ISPs as follows: we vary u_{s1} (server capacity in ISP 1) from 0 to 100, and meanwhile vary $u_{s2} + u_{s3}$ (the total amount deployed in ISPs 2 and 3) between 0 and 100 as well; giving u_{s1} and $u_{s2} + u_{s3}$, u_{s4} will be fixed, as the total server capacity in the entire system is 100. Low inter-ISP link delays with $L = 1$ are used.

Fig. 3 (a) shows that when more server capacity is deployed in ISPs with lager peer upload bandwidths, the end-to-end delay in the system is smaller: the end-to-end delay is the lowest when all server capacity is deployed in ISP 1; when all capacity is deployed in ISP 2 and ISP 3, the delay is lower than that in the case when all is deployed in ISP 4. This is also what insight (iv) implies.

On the other hand, Fig. 3 (b) shows that when all server capacity is deployed in ISP 1, the volume of inter-ISP traffic is the largest. When all server capacity is deployed in ISP 2 and ISP 3, the volume of inter-ISP traffic is smaller but is larger than that in the case when all is deployed in ISP 4. This can also be concluded from insight (ii).

C. Impact of Peer Population Distribution

We further examine the minimum end-to-end chunk dissemination delay and the inter-ISP traffic required to achieve it, in four cases with different server capacity and peer population distributions:
(1) $(N_1, N_2, N_3, N_4) = (7000, 1000, 1000, 1000)$, $(u_{s1}, u_{s2}, u_{s3}, u_{s4}) = (100, 0, 0, 0)$;
(2) $(N_1, N_2, N_3, N_4) = (1000, 4000, 4000, 1000)$, $(u_{s1}, u_{s2}, u_{s3}, u_{s4}) = (0, 50, 50, 0)$;
(3) $(N_1, N_2, N_3, N_4) = (1000, 1000, 1000, 7000)$, $(u_{s1}, u_{s2}, u_{s3}, u_{s4}) = (0, 0, 0, 100)$;
(4) $(N_1, N_2, N_3, N_4) = (2500, 2500, 2500, 2500)$, $(u_{s1}, u_{s2}, u_{s3}, u_{s4}) = (25, 25, 25, 25)$.

From Fig. 4 (a), we observe that when the inter-ISP link latency is low ($L = 1$ or $L = 2$), the streaming performance in Case 1 (where server capacity is all deployed in ISP 1 with the largest peer capacity and most peers) is the best, which verifies insight (iv) in the previous section. When the inter-ISP link latency is large ($L = 5$), uniform server capacity deployment among all ISPs with even peer population distribution (Case 4) achieves the best streaming performance. This provides us another insight: to achieve minimum dissemination delay when inter-ISP link latencies are large, servers should be deployed in more ISPs with an amount relative to the peer populations.

In addition, Fig. 4 (b) further reveals that when most server capacity is deployed in ISPs with the majority of peer population and whose peer average upload bandwidth is not the highest in the system (Case 2 and Case 3), the volume of inter-ISP traffic is small. This verifies insight (v) in the previous section.

D. Impact of Different Inter-ISP Link Latencies

In our previous studies, the latencies on all inter-ISP links are the same. We next investigate how different inter-ISP link delays influence the server capacity deployment, inter-ISP traffic, and chunk dissemination delay in the system. In these set of experiments, the inter-ISP link delay is $L = 5$ between ISP 2 and each of the other ISPs, and is $L = 2$ between any two of ISP 1, ISP 3, and ISP 4. Server capacities are deployed in ISPs 1 and 2, but not

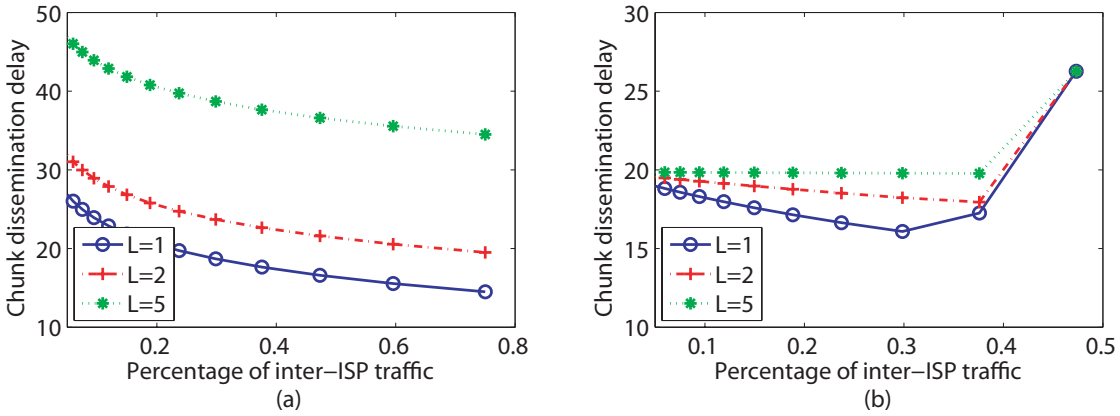


Figure 2. End-to-end chunk dissemination delay vs. inter-ISP traffic: (a) $(u_{s1}, u_{s2}, u_{s3}, u_{s4}) = (100, 0, 0, 0)$; (b) $(u_{s1}, u_{s2}, u_{s3}, u_{s4}) = (45, 10, 10, 35)$.

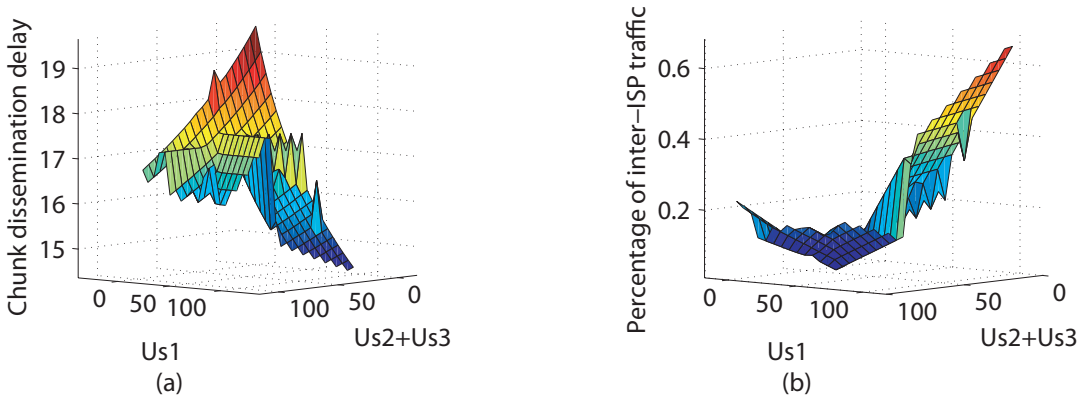


Figure 3. Minimum end-to-end chunk dissemination delay and the required inter-ISP traffic at different server capacity deployments.

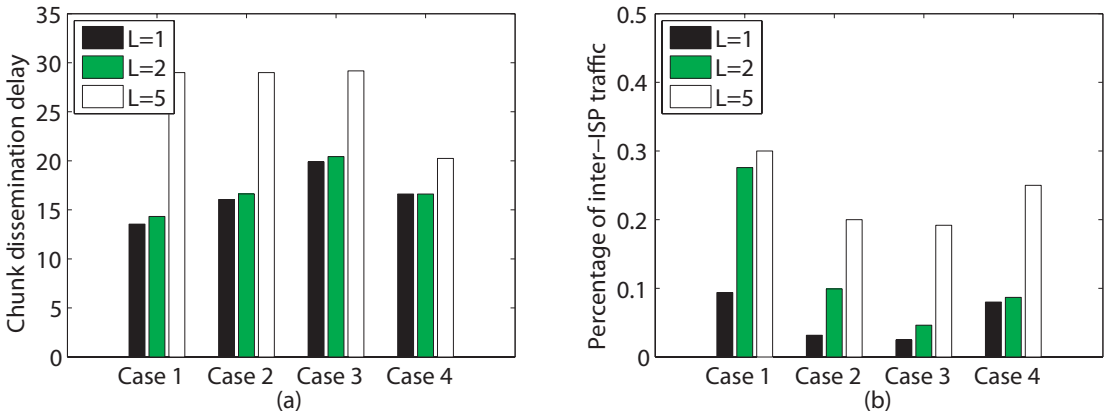


Figure 4. Minimum end-to-end chunk dissemination delay and the required inter-ISP traffic at different server capacity and peer population distributions.

in the other two. All other settings are the same as in previous experiments.

Fig. 5 (a) shows that the minimum dissemination delay is achieved when a small amount of server capacity is deployed in ISP 2. As the server capacity in ISP 2 increases, the system performance becomes worse. These observations are consistent with insight (iv).

Fig. 5 (b) shows that the minimum inter-ISP traffic is achieved when server capacity is evenly distributed in ISP 2 and ISP 3, considering that both ISPs have the same peer upload bandwidths and peer population. When the percentages of server capacity in ISP 2 and ISP 3 become

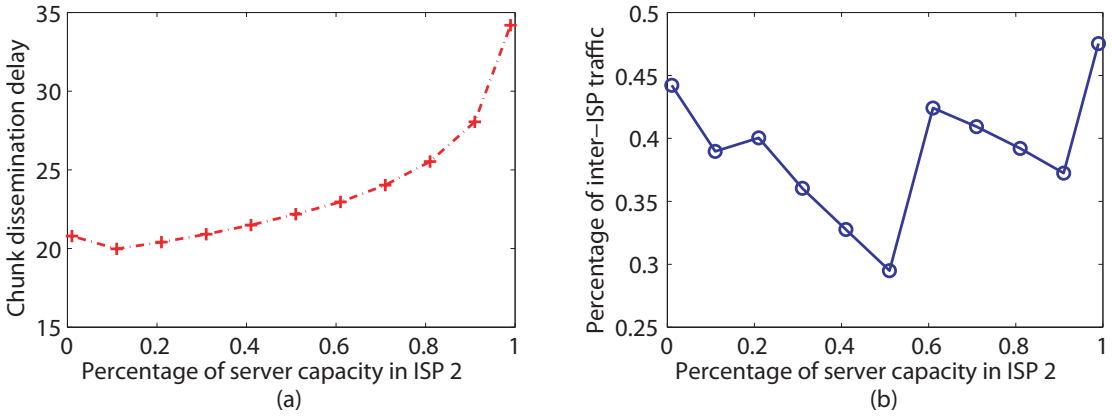


Figure 5. Minimum end-to-end chunk dissemination delay and the required inter-ISP traffic at different server capacity deployments under different inter-ISP link latencies.

unbalanced, the inter-ISP traffic is larger. The decrease of inter-ISP traffic when most server capacity is delayed in ISP 2 can be explained by the large link delay between ISP 2 and each of the other ISPs, which discourages inter-ISP chunk retrieval.

V. Related Work

A number of work have been conducted on theoretical analysis of P2P live streaming systems, in order to derive useful insights for performance improvement. Liu *et al.* [11] derive performance bounds of the streaming system, including minimum server load, maximum streaming rate, and minimum tree depth under different peer selection constraints, and also construct tree-based streaming systems to validate these bounds. Chen *et al.* [13] explore the streaming capacity of a streaming system with node degree bound. They propose a Bubble Algorithm for a system with bound on node out-degrees and a Cluster-tree Algorithm for systems with logarithmic bound on node's total degree. Liu [10] analyzes the chunk dissemination delay bound in P2P live streaming systems, based on a snow-ball algorithm. In Kumar *et al.*'s work [9], peers' upload bandwidth constraints are considered, and a stochastic fluid model is applied to calculate the maximum streaming rate that a churnless system can sustain and the probability of universal streaming at the streaming rate in a system with churns. Bonald *et al.* [16] treat chunk dissemination in a P2P live streaming system as a diffusion process, and analyze peer/chunk selection algorithms as diffusion schemes. They derive the diffusion rate and delay of chunks based on various diffusion schemes, and prove the qualitative result that the random peer/latest useful chunk selection algorithm can achieve the best diffusion at an optimal rate within an optimal delay. Zhou *et al.* [14] use recursive equations to model the buffer occupancy at peers under two chunk selection strategies, greedy and rarest first, and derive streaming continuity and start-up latency in the system. Massoulie *et al.* [12] propose a chunk dissemination heuristic for node-capacitated P2P networks, based on an optimal packet forwarding algorithm in edge-capacitated networks, and prove that the heuristic works well in complete graphs. All the above work does not take inter-ISP traffic into consideration in their analytical models.

There have been a number of algorithm/protocol designs to induce traffic locality in P2P systems. P4P [6] is a general solution for cooperative traffic control in P2P applications. In P4P, an iTracker portal is introduced as interfaces for network providers to provide information of the underlying network to P2P solution providers, in order to achieve traffic optimization. Picconi *et al.* [7] propose an algorithm to construct two levels of overlays: a primary overlay is created preferentially among nearby peers and a secondary overlay connects peers randomly across the network. Magharei *et al.* [8] construct a localized overlay by explicitly controlling the number of external connections that peers in one ISP could establish with peers in other ISPs. They propose an inter-ISP scheduling algorithm for delivering streams to individual ISPs and an intra-ISP scheduling algorithm to ensure the delivery of streams to all internal peers in an ISP. They also analytically prove the feasibility of streaming over such a localized overlay with limited external connections.

VI. Concluding Remarks

This paper targets at in-depth investigation of the impact of locality-aware protocol design on the achievable streaming performance in a P2P live streaming system. Towards this objective, we carefully model the relationship between volumes of inter-ISP traffic and the streaming performance, in P2P streaming systems over multiple ISPs at different bandwidth levels. In particular, we derive analytical formulas to derive the minimum end-to-end chunk dissemination delay in the system, as well as the amount of inter-ISP traffic needed to achieve this minimum delay. We also explore the best server capacity deployment strategies to achieve the best streaming performance, when minimum inter-ISP traffic is incurred. Our models and analyses provide useful insights on the design of efficient locality-aware P2P streaming protocols

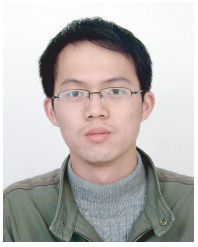
and effective server deployment strategies across multiple ISPs, which can achieve desired goals on inter-ISP traffic minimization or streaming performance optimization.

ACKNOWLEDGEMENT

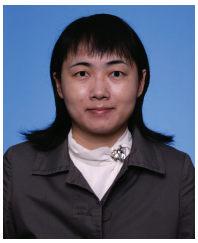
This work has been supported in part by the University of Hong Kong under Small Project Funding, and the Research Grants Council of Hong Kong under RGC General Research Fund (Ref: HKU718710E).

REFERENCES

[1] *PPLive*, http://www.pplive.tv.
[2] *SopCast*, http://www.sopcast.org/.
[3] *Zattoo*, http://www.zattoo.com/.
[4] C. S. Inc., "Cisco Visual Networking Index: Forecast and Methodology, 2009-2014, White Paper," 2010.
[5] S. Sen, O. Spatscheck, and D. Wang, "Accurate, Scalable In-Network Identification of P2P Traffic Using Application Signatures," in *Proc. of WWW*, May 2004.
[6] H. Y. Xie, Y. R. Yang, A. Krishnamurthy, Y. B. G. Liu, and A. Silberschatz, "P4P: Provider Portal for Applications," in *Proc. of ACM SIGCOMM*, August 2008.
[7] F. Picconi and L. Massoulie, "ISP Friend or Foe? Making P2P Live Streaming ISP-Aware," in *Proc. of IEEE ICDCS*, June 2009.
[8] N. Magharei, R. Rejaie, V. Hilt, I. Rimac, and M. Hofmann, "ISP-Friendly Live P2P Streaming," in *Poster. of ACM SIGCOMM*, August 2009.
[9] R. Kumar, Y. Liu, and K. Ross, "Stochastic Fluid Theory for P2P Streaming Systems," in *Proc. of IEEE INFOCOM*, May 2007.
[10] Y. Liu, "On the Minimum Delay Peer-to-peer Video Streaming: How Realtime Can It Be?" in *Proc. of MULTIMEDIA*, September 2007.
[11] S. Liu, R. Zhang-Shen, W. J. Jiang, J. Rexford, and M. Chiang, "Performance Bounds for Peer-Assisted Live Streaming," in *Proc. of SIGMETRICS*, June 2008.
[12] L. Massoulie, A. Twigg, C. Gkantsidis, and P. Rodriguez, "Randomized Decentralized Broadcasting Algorithms," in *Proc. of IEEE INFOCOM*, May 2007.
[13] S. Liu, M. H. Chen, S. Sengupta, M. Chiang, J. Li, and P. A. Chou, "P2P Streaming Capacity under Node Degree Bound," in *Proc. of IEEE ICDCS*, June 2010.
[14] Y. P. Zhou, D. M. Chiu, and J. C. S. Lui, "A Simple Model for Analyzing P2P Streaming Protocols," in *Proc. of IEEE ICNP*, October 2007.
[15] C. Feng, B. C. Li, and B. Li, "Understanding the Performance Gap Between Pull-Based Mesh Streaming Protocols and Fundamental Limits," in *Proc. of IEEE INFOCOM*, April 2009.
[16] T. Bonald, L. Massoulié, F. Mathieu, D. Perino, and A. Twigg, "Epidemic Live Streaming: Optimal Performance Trade-Offs," in *Proc. of ACM SIGMETRICS*, June 2008.
[17] *CoralReef Suite*, http://www.caida.org/tools/measurement/coralreef/.
[18] P. T. Boggs and J. W. Tolle, "Sequential Quadratic Programming," *Acta Numerica*, vol. 4, pp. 1–51, 1995.



Jian Zhao. Jian Zhao received his B.Engr. degree in 2009 from Department of Electronic Engineering and Information Science, University of Science and Technology of China, China. He is currently a Ph.D. candidate in the Department of Computer Science, the University of Hong Kong, China. His research interests include peer-to-peer streaming and optimization of network traffic.



Chuan Wu. Chuan Wu received her B.Engr. and M.Engr. degrees in 2000 and 2002 from Department of Computer Science and Technology, Tsinghua University, China, and her Ph.D. degree in 2008 from the Department of Electrical and Computer Engineering, University of Toronto, Canada. She is currently an assistant professor in the Department of Computer Science, the University of Hong Kong, China. Her research interests include measurement, modeling, and optimization of large-scale peer-to-peer systems and online/mobile social networks. She is a member of IEEE and ACM.

An Analysis and Comparison of CDN-P2P-hybrid Content Delivery System and Model

Zhihui Lu
School of Computer Science, Fudan Universiy, Shanghai, 200433, China
Email: lzh@fudan.edu.cn

Ye Wang and Yang Richard Yang
Department of Computer Science, Yale University, New Haven, CT 06511, USA
Email: {ye.wang, yang.r.yang}@yale.edu

***Abstract*- In order to fully utilize the stable edge transmission capability of CDN and the scalable last-mile transmission capability of P2P, while at the same time avoiding ISP-unfriendly policies and unlimited usage of P2P delivery, some researches have begun focusing on CDN-P2P-hybrid architecture and ISP-friendly P2P content delivery technology in recent years. In this paper, we first survey CDN-P2P-hybrid architecture technology, including current industry efforts and academic efforts in this field. Second, we make comparisons between CDN and P2P. And then we explore and analyze main issues, including overlay route hybrid issues, and playing buffer hybrid issues. After that we focus on CDN-P2P-hybrid model analysis and design, we compare the tightly-coupled hybrid model with the loosely-coupled hybrid model, and we propose that there are some main common models which need further study. At last, we analyze the prospective research direction and propose our future work.**

***Keywords*- CDN, P2P, P2P Streaming, CDN-P2P-hybrid Architecture, Live Streaming, VoD Streaming**
Note: This work is supported by 2009 National Science Foundation of China (60903164): Research on Model and Algorithm of New-Generation Controllable, Trustworthy, Network-friendly CDN-P2P hybrid Content Delivery.

I. INTRODUCTION

The ongoing growth of broadband technology in the worldwide market has been driven by the hunger of customers for new multimedia services as well as Web content. In particular, audio and video streaming has become popular for delivery of rich information to the public - both residential and business. In recent years, PC-based Internet streaming, IP network television (IPTV), large-size file downloading, and high-definition video have become mainstream broadband streaming applications. The high-bandwidth, high-traffic and high QoS demands, inherent in these applications, have brought huge challenges to the current best-effort Internet. How to implement large-scale, low cost, QoS guaranteed content delivery has become one core problem. In the next-generation Internet, various types of wired and wireless networks will coexist, such as broadband Internet, wireless wide area network (WWAN, 3G and post-3G), wireless metropolitan area network (WMAN), wireless LAN, wireless personal area network, and so on. At the same time, a variety of wired / wireless terminals are emerging, including mobile PC, TV set-top box, 3G mobile phone, netbook, iPad and so on. All these different terminals will obtain streaming content and service through a variety of heterogeneous access networks, such as ADSL, the cable network, Wi-Fi, 3G, and Wimax, et al.

Therefore, the question of how to build a number of low-cost expandable, controllable and manageable, fast and efficient, safe and reliable content service networks on top of the Internet IP layer, is a key issue.

In current content service platforms, Content Delivery Networks (CDN) is a representative technique. CDN is based on large-scale distributed cache servers located closer to the edges of the Internet for efficient delivery of digital content including various forms of multimedia content. There are many commercial CDN companies [2], including Akamai, AT&T, NTT Communication, Limelight, Mirror Image, Level 3, Verisign and Internap. Akamai is the largest among these CDN companies [26]. Akamai's EdgePlatform comprises 56,000 servers deployed in 70 countries and 1000 networks that continually monitor the Internet, including traffic, trouble spots and overall conditions.

Although CDN is an effective means of streaming content access and delivery, there are two barriers to making CDN a more common service: expensive construction cost and administration complexity. Deploying a CDN for publicly available content is very expensive. It requires administrative control over cache nodes with large storage capacity at geographically dispersed locations with adequate connectivity [12]. Each cache node must be configured to participate in the replication scheme and to automatically update its directory structure. The administrative system for a CDN must be concerned with both node hardware and content availability. Failed hardware components must be reported as well as failed or corrupted data replication. CDN can be scalable, but due to this administrative overhead and expensive cost, not rapidly so.

Manuscript received April 15, 2011; revised July 15, 2011; accepted October 15, 2011.

doi:10.4304/jcm.7.3.232-245

Popularity of Peer to Peer (P2P) streaming has been shown to greatly reduce the dependence on CDN, as well as accelerate the distribution process between content providers and consumers. Many research works have been undertaken in P2P live streaming and P2P VoD streaming [3-10]. There are several large-scale P2P streaming applications that serve millions of users, such as PPLive, UUSee, PPStream, SopCast, et al. Indeed, streaming media content delivery using various peer-to-peer or peer-assisted frameworks allows the sharing of client resources, such as CPU, memory, storage, bandwidth, etc, and has greatly reduced the dependence on central content servers or CDN servers [34]. The P2P approach is more scalable and needs less investment, and each client not only gets service but also provides service as well. The con is that it is not easy to manage and the QoS is also a problem due to dynamic churn. Peers usually perform selfishly, and will ignore the global benefit, for example the back-bone consumption etc [34]. P2P has also fundamentally altered the relationship among content owners, network providers (ISPs) and consumers. Popularity of P2P streaming applications has resulted in increased traffic on ISP networks. Being largely network-oblivious, many P2P applications may lead to substantial network inefficiency. For the benefits of backbone network carriers and ISPs, it is desirable to combine the features of P2P and CDN systems, that will best protect the previous investment and improve the service capability at the same time. P2P technology is not a replacement for CDN, while it's a complement to CDN. This is because there are instances where both sets of technologies have their own strengths and weaknesses [30]. A natural question is whether they can be combined to obtain the scalability advantage of P2P, as well as the reliability and manageability advantages of CDN. Indeed, with recent rapid growth of P2P applications and CDNs, many industrial and academic initiatives have been taken to combine the two technologies to get the "best of both worlds". The most challenging problem of current content delivery network is to realize controllable, manageable, credible, network-friendly content distribution architecture through the integration of CDN and P2P.

In recent years, some researches have begun focusing on ISP-friendly P2P content delivery technology (e.g., [4, 7, 9]) and CDN-P2P-hybrid architectures (e.g., [11,12,14-22,25,32-35,39]). In this paper, we first make a survey of CDN-P2P-hybrid architecture technology, including current industry efforts and academic efforts. Second, we make comparisons between CDN and P2P. Then we explore and analyze main issues, including overlay route hybrid issues, and playing buffer hybrid issues. After that we focus on CDN–P2P-hybrid model analysis and design, and we compare the tightly-coupled hybrid model with the loosely-coupled hybrid model, finally we point out that there are some main common CDN-P2P-hybrid models which need further study.

To the best of our knowledge, our work represents the first extensive and in-depth analysis and comparison of CDN-P2P-hybrid systems and models. To summarize, the key contributions and key findings of this paper are the following:

- We present a comprehensive survey of CDN-P2P-hybrid technology, including current industrial efforts and academic efforts in this field. Specifically, we conduct a thorough analysis of the academic achievements of this new field in recent years.
- Furthermore, we explore and discuss major issues and problems arising in the CDN and P2P integration process, including overlay route hybrid issues, and playing buffer hybrid issues. Then we make a detailed comparison of how existing hybrid schemes dealing with these issues.
- We propose that there are two kinds of CDN-P2P hybrid models: the tightly-coupled hybrid 1+1 model and the loosely-coupled 1+N hybrid model. And we further analyze their applicability based on their strengths and weaknesses.
- Finally we assess that there are some main common CDN-P2P-hybrid models that need in-depth study, including (1)the hybrid-scheduling model between CDN global scheduling and P2P distributed routing,(2)the total hybrid-delivery services capacity dynamic allocation model,(3) the P2P-ISP friendly delivery model under the guidance of CDN,(4) the controllable management model, and (5) the secure and trustable delivery model, et al.

The remainder of this paper is organized as follows: in Sec. II, we make a detailed survey on current CDN-P2P-hybrid delivery technologies. In Sec. III, we compare CDN and P2P streaming architecture's strengths and weaknesses. We explore and analyze the main issues of CDN-P2P-hybrid delivery and some existing solutions for these issues in Sec IV. In Sec V, we propose two kinds of CDN-P2P hybrid models: the tightly-coupled hybrid model and the loosely-coupled hybrid model and we also analyze the common CDN-P2P-hybrid delivery models. We conclude the paper and propose our future work in Sec.VI.

II. CDN-P2P-HYBRID ARCHITECTURE TECHNOLOGY SURVEY

A. Industrial Efforts on CDN-P2P Hybrid Architecture

The hybrid CDN-P2P delivery model is becoming increasingly popular largely due to the rapid growth in the demand for online video and due to industrial efforts.

Akamai is making a big push into the P2P-CDN business. Akamai, with its acquisition of Red Swoosh P2P technology, is expected to combine P2P file distribution software with its back-end control system and global network of edge servers [18]. Akamai applied for an open patent in 2008: "Hybrid content

delivery network (CDN) and peer-to-peer (P2P) network" [22]. In this patent, they described the following mechanisms: a content delivery network (CDN) typically includes a mapping system for directing requests to CDN servers. One or more peer machines become associated with the CDN, and the CDN mapping system is then used to enable a given peer to locate another peer in the P2P network, and/or a CDN server. Using this hybrid approach, CDN customer content may be delivered from the CDN edge network, from the P2P network, or from both networks. In one embodiment, customer content is uploaded to the CDN and stored in the edge network, or in a storage network associated therewith. The CDN edge network is then used to prime the P2P network, which may be used to take over some of the content delivery requirements for the customer content. The decision of whether to use the edge network or peer network resources for delivery may be based on load and traffic conditions.

VeriSign, CacheLogic, Grid Networks, Internap, and Joost have all announced their own CDN-P2P services as well [18]. VeriSign (VRSN) is launching CDN-P2P combination project. And Cachelogic, has been working with carriers and are already hawking their P2P-CDN offerings. For instance, they announced that they are working with Babelgum, to deliver video over its Velocix Network, a P2P-CDN service that Cachelogic claims will radically change the economics of content delivery. Even pure-play P2P companies like Pando have started dabbling with P2P CDNs [28]. Pando Networks announced they are putting the world's largest Hybrid P2P content delivery service in the hands of media companies so they can accelerate and optimize their HD media streams, downloads and RSS channels [30]. The inherent properties of hybrid P2P networking represents a fundamental evolution of the physics of content delivery — supply is directly proportional to demand and inversely proportional to cost.

GridNetworks wants to provide the underlying technology that streams high quality television content over the internet and into homes. Its architecture is part traditional CDN, part peer-to-peer network. In simple terms, the content delivered by GridCasting is initially buffered by a handful of data centers. Then after about 10-30 seconds of video playback, the data centers hand over delivery responsibilities to around 16 peers (other consumers of online video content who have cached it already and can serve as mini data centers themselves) [27]. Cisco will include Grid Networks' peer-to-peer software in its home networking products, "They're going to embed the GridCasting client into edge devices in millions of households in the coming year-plus," Grid Networks CEO Tony Naughtin told a reporter in 2008 [29].

ChinaCache, the largest CDN company in China, has developed and deployed a hybrid CDN-P2P live streaming system-LiveSky [17][35]. LiveSky works well even when the client upload bandwidth is restricted and ineffective. LiveSky provides good performance even with churn –the fraction of peers that join or leave in a specific interval. In later chapters, we continue to analyze LiveSky.

B. Academic Efforts on CDN-P2P Hybrid Architecture

Therefore, in order to fully make use of the stable backbone-to-edge transmission capability of a CDN and the scalable last-mile transmission capability of P2P, while at the same time avoiding ISP-unfriendly policies and unlimited usage of P2P delivery, some researches have been focusing on CDN-P2P-hybrid architecture and ISP-friendly P2P content delivery since 2006.

Dongyan Xu et al. published one earliest papers focusing on this issue [11]. This paper proposed and analyzed a novel hybrid architecture that integrates both CDN- and P2P-based streaming media distribution. The architecture is highly cost-effective: it significantly lowers the cost of CDN capacity reservation, without compromising the media quality delivered. In particular, they proposed and compared different limited contribution policies for peers that request media data, so that the streaming capacity of each peer can be exploited on a fair and limited basis [11]. Their analytical and simulated results formed a rigorous basis for the planning and dimensioning of the hybrid architecture. Through analysis of this paper, we think their hybrid architecture design is relatively simple; just one CDN server link to P2P network. In practice, the hybrid architecture of CDN and P2P is a multiple-to-multiple relationship. The author stated that the stages of a media data distribution process are from CDN-only at beginning, transferring to CDN-P2P coexistence, to P2P only at last. This is an idealistic situation. In practice, CDN and P2P coexist most of the time, with different shares of the workload over time.

Our previous work proposed a novel hybrid architecture [12]-PeerCDN, which combines the two approaches seamlessly with their inherent strengths. PeerCDN is a two-layer streaming architecture. The upper layer is a server layer which is composed of original CDN servers including origin servers and replica servers. The lower layer consists of groups of clients who request the streaming services, with each client acting as a peer node in the group. Each group of client peers is led by the nearby replica server-strong node. Client peers contribute their resources through the coordination of the strong node. The scheme uses a revised Kademlia-like protocol to construct a topology-aware overlay network.

Specifically, there are some academic papers focusing on this issue in some famous conferences, such as SIGCOMM2007/2008 and INFOCOM 2007/2008. In SIGCOMM2007, Cheng Huang et al.[9] showed that peer-assistance can dramatically reduce server bandwidth costs, particularly if peers prefetch content when there is spare upload capacity in the

system. They consider the impact of peer-assisted VoD on the cross-traffic among ISPs. Although this traffic is significant, if care is taken to localize the P2P traffic within the ISPs, they can eliminate the ISP cross traffic while still achieving important reductions in server bandwidth. We have proposed the P4P project in SIGCOMM2008, [13]. The P4P framework is a flexible and light-weight framework that allows network providers (ISP) to explicitly provide more information, guidelines and capabilities to emerging applications, such as P2P content distribution. P4P stands for provider portal for applications. P4P provides multiple interfaces for networks to communicate with P2P applications. The interfaces preserve network provider privacy and allow network providers and applications to jointly optimize their respective performance. The evaluations demonstrate that this can be a promising approach to improve both application performance and provider efficiency. But P4P requires ISPs to expose the necessary interface, which may result in ISPs operation burden and some security problems and need relatively close cooperation between ISPs and P2P application.

In SIGCOMM2008, David R. Choffnes et al. [14] presented Ono: the design, deployment and evaluation of an approach to reducing this costly cross-ISP traffic without sacrificing system performance. Their approach recycles network views gathered at low cost from content distribution networks (CDN) to drive biased P2P neighbor selection without any path monitoring or probing. The results show that their lightweight approach significantly reduces cross-ISP traffic and, over 33% of the time, it selects peers along paths that are within a single autonomous system (AS). This method is one kind of loosely-coupled hybrid structure between CDN and P2P. P2P cleverly uses some external public information about the CDN to make the decision of choosing neighbors with ISP-friendly consideration. Currently there are some closely-coupled hybrid CDN-P2P systems, and in this paper, we comparatively study their respective characteristics.

Combining the advantages of CDN and P2P networks has been considered a feasible orientation for large-scale video stream deliver. To guarantee QoS and facilitate management in large scale high-performance media streaming, Zhijia Chen et al. [15] extended the current P2P model towards a novel Peer-Server-Peer (PSP) architecture for media streaming. In their Peer-Server-Peer (PSP) structure, media content is first distributed among trusted servers and eventually reaches different clusters of peers for their P2P distribution. All the carefully-deployed servers (some may belong to CDN), which are formed by dedicated proxy servers for content distribution) form a trustworthy and controllable overlay network, through which servers provide initial content, guide streaming traffic to achieve overall traffic optimization, and conduct key distribution. Their research is ongoing. Xuening Liu et al. [16] proposed a peer-assisted content distribution network, i.e. PACDN, which blends the mesh-based P2P ideas into the traditional CDN to enhance performance and scalability. The basic features of PACDN include: 1) The placement of edge servers and source streaming servers, which build a hierarchical multi-tree based on an in-hierarchy peer-assisted overlay, which is optimized according to the knowledge of underlying physical topology. This scheme in the design is called “server side peer-assisted”. 2) To enhance the system scalability and reduce the deployment costs, clients and edge servers construct a Client/Server based and P2P network assisted overlay with the increase of viewers, which is called “client side peer-assisted” in this design.

Based on the above two paper’s research, in the latest ACM Multimedia2009 and ACM Transaction2010, Hao Yin et al. presented design and deployment experiences of LiveSky[17] [35], a commercially deployed hybrid CDN-P2P live streaming system. CDNs and P2P systems are the common techniques used for live streaming, each having its own set of advantages and disadvantages. LiveSky inherits the best of both worlds: the quality control and reliability of a CDN and the inherent scalability of a P2P system. They addressed several key challenges in the system design and implementation including (a) dynamic resource scaling while guaranteeing stream quality, (b) providing low startup latency, (c) ease of integration with existing CDN infrastructure, and (d) ensuring network-friendliness and upload fairness in the P2P operation. To summarize the three papers, we find the integration of CDN and P2P forms a relatively tightly-coupled relationship. In their CDN-P2P hybrid topology, they described that every CDN node lead a group of peer nodes, therefore, all the nodes have been split into different groups. The existing P2P systems have to change their own overlay structure to accept the leadership of the dominant CDN. This hybrid method is efficient, but it is not suitable for one CDN to integrate with multiple heterogeneous P2P, which needs one kind of loosely-coupled relationship. Later we will discuss these two kinds of relationship in detail.

Cheng Huang et al.[18] quantified the potential gains of hybrid CDN-P2P for two of the leading CDN companies, Akamai and Limelight. They first developed a novel measurement methodology for mapping the topologies of CDN networks. They then considered ISP-friendly P2P distribution schemes which work in conjunction with the CDNs to localize traffic within regions of ISPs. To evaluate these schemes, they used two recent real-world traces: a video-on-demand trace and a large-scale software update trace. They found that hybrid CDN-P2P can significantly reduce the cost of content distribution, even when peer sharing is localized within ISPs and further localized within regions of ISPs. This paper focuses on measurement techniques of CDN-P2P

hybrid architecture; they do not include a detailed description for the design of a CDN-P2P hybrid mechanism.

Hai Jiang et al. [19] presented a hybrid content distribution network (HCDN) integrating complementary advantages of CDN and P2P technology, which is used to improve efficiency of large scale content distribution. Their hybrid model is simplistic, but they carried out a detailed performance evaluation based on a deterministic fluid model. And they provided numeric results of HCDN, a conventional CDN and pure P2P. As shown in their analysis and numerical results, they draw the conclusion that the HCDN has many advantages on average downloading time, service capacity and system scalability.

Duyen Hoa HA et al. [20] introduced a new hybrid CDN-P2P solution for real time streaming. Different from others, their solution is based on the effective management of the playing buffer at the peer-side to best equilibrate the bandwidth used between the CDN side and the P2P side. The main idea is to divide the playing buffer into two parts: the CDN priority part and the P2P priority part. During the playback time, lacking packets in the CDN priority part will be received from CDN servers, and lacking packets in the P2P priority part will be received from other peers. By divide the playing buffer into 2 parts, they can profit from all the advantages of CDN servers and P2P network: the performance of CDN servers and the cheap cost of using peers to distribute media content. Their paper is not concerned with how CDN is integrated with P2P, and their focus is, after integration, on how to utilize the buffer division mechanism to get the respective contents of CDN and P2P resources to work in parallel.

A new concept of integrating both CDN and P2P technologies into a replication-aware CDN-P2P architecture has been proposed by Hung-Chang Yang et al.[21]. They proposed a two-step selection approach on landmark-based selection algorithm to find the nearest replica-cache server and peer-caches to download content in this architecture. Furthermore, recent studies have supported the claim that network coding technology is beneficial for large-scale P2P content distribution. Herein, they showed how to apply network coding technology to distribute content so that the content provider's rights can be protected.

Jimmy Jernberg et al. [32] presented a new approach to building a scalable Web Hosting environment as a CDN on top of a structured peer-to-peer system of collaborative web-servers integrated to share the load and to improve the overall system performance, scalability, availability and robustness. Unlike cluster-based solutions, it can run on heterogeneous hardware, over geographically dispersed areas. Manal El Dick et al. [33] have developed Flower-CDN, a locality-aware P2P based content-distribution network (CDN) in which the users that are interested in a website support the distribution of its content. The idea is that peers keep the content they retrieve and later serve it to other peers that are close to them in locality. Their architecture is a hybrid between structured and unstructured networks. When a new client requests some content from a website, a locality-aware DHT quickly finds a peer in its neighborhood that has the content available. Additionally, all peers in a given locality that maintain content of a particular website build an unstructured content overlay.

III. CDN AND P2P ARCHITECTURE TECHNOLOGY COMPARISON

Through the analysis of these above related papers, we make a detailed comparison between CDN and P2P in the following table. We can observe both of them have their own advantages and fundamental shortcomings. The two techniques have great complementary characteristics, so their integration will provide a big opportunity to build a manageable, reliable, QoS-guaranteed, and scalable content service platform.

TABLE I. COMPARISON BETWEEN CDN AND P2P

Comparison Item between CDN and P2P	**CDN**	**P2P**
Service Capability and Scalability	Service capability is limited and expansion cost is higher	Service capability can grow up with peer node increases and expansion cost is lower
Reliability and Stability	High reliability, good stability	Low reliability, dynamic, and poor stability
Network-friendly and orderly flow	Network(ISP)-friendly, flow is controlled in different regions	Network(ISP)-unfriendly , traffic disorder, cross-ISP expansion in the whole network
Content Source Monitor	Can be monitored	It is difficult to monitor
User Management	Centralized user management	Loose or less user management
QoS Guarantee	Can be guaranteed within the maximum service capacity	Best-effort, can't be controlled
Content Copyright and Security	Controllable and manageable	Uncontrollable, non-management, content pollution may arise
Service Node Authentication	Center certification	Distributed certification or non-certification
Service Node	CDN node is service node, and client nodes just	Client nodes can provide P2P content delivery service to other

	access to CDN services, service node is homogeneous	client nodes, service node is heterogeneous

IV. CDN AND P2P HYBRID ISSUES DISCUSSION

A. Overlay Route Hybrid Issues

Using a CDN-P2P hybrid approach, client users can fetch contents from the CDN edge network, from the P2P network, or from both networks. The decision of whether to use the edge network or the peer network resources for delivery is based on overlay route technology [39]. There are two kinds of CDN and P2P hybrid overlay: PAC (Peer-aided CDN) [12,16,17,20,21,22] and CAP (CDN-aided P2P) [3,36,37]. Most CDN providers, such as Akamai[21], ChinaCache[17], integrate their CDN with P2P in PAC manner. Most overlay route technologies in CDN are using DNS redirection technology, such as Akamai. In PAC method, users are mainly redirected and served by CDN. P2P overlay is applied to improve user performance and to alleviate the stress of CDN. However, in CAP method, most streaming contents are distributed to users through P2P network. CDN serves as a rescuer for some starving P2P peers. Most P2P application providers, such as PPLive, UUSee, integrate their P2P network with CDN in CAP manner. The overlay route technology of most P2P streaming application is using tracker-based redirection technology, such as PPLive, PPStream, and many P2P file sharing applications are using DHT technology, such as BT, E-mule, et al. We introduce related work of PAC and CAP, respectively.

(1) PAC Method

In [22], Akamai PAC hybrid delivery scheme's overlay route mechanism mainly relies on the CDN DNS redirection mechanism. Figure1 describes the main mechanisms as the following. Every step number corresponds to the number of Figure1. Note Step (3) and Step (3') is an alternative step; if Step (3) is selected, and then the next step is Step (4), otherwise Step (3') and (4').

(1) One peer node's request is directed to the CDN, which in one embodiment then returns to the peer node a file, sometimes referred to as a metafile. In one embodiment, this metafile includes one or more CDN or hybrid CDN-P2P domains or sub-domains that can then be used by the peer node to obtain the desired content.

(2) Thus, for example, assume the metafile includes a set of domains such as peer.ake.net, peer.cdn.net, and the like, each of which is resolved by the CDN DNS query mechanisms, which is authoritative for all domains returned in the metafiles.

(3) In this example, the first domain is designed to be resolved to another peer in the P2P network, and the second domain is designed to be resolved to an edge server in the CDN network (thus acting as failover in this example). This ordering is merely representative, as the order may be switched so that the peer is the backup. In either case, the peer node client then makes a DNS query to the first domain or sub-domain in the list, and that DNS query is resolved through the CDN DNS query mechanism to identify a nearby peer in the P2P network from which the content can be fetched.

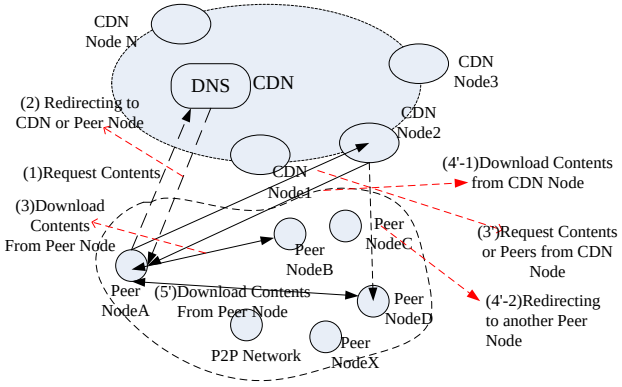


Figure1. Akamai hybrid delivery scheme

(3') If this operation fails, if the peer cannot contact the indentified peer, or if the identified peer does not have the content, the second domain is tried, this time returning an edge server in the CDN. This will be an edge server that is nearby, that is likely to have the content and that is not overloaded. But if the client's request has been redirected to an edge server, the edge server can then choose how to handle this request, ie., (4'-1) by delivering the content objects itself, or (4'-2) by redirecting the request to a peer network resource. (5') Downloading the contents from a peer node, the next step of (4'-2).

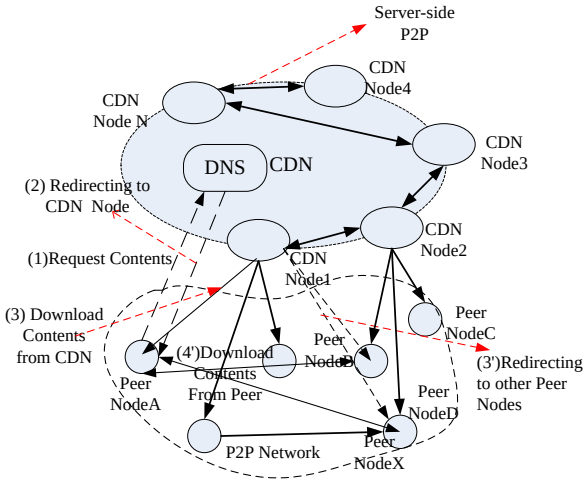


Figure2. LiveSky hybrid delivery scheme

Livesky[17] also used the PAC hybrid delivery scheme's overlay route mechanism. It first relies on the CDN DNS redirection mechanism to reach an edge server, and the edge server can act as the tracker of P2P

overlay to help the requested peer find neighbors. They described the main mechanisms in Figure2.

(1) A client first obtains the URL for the live stream from the content source (e.g., livesky://domainname/live1).

(2) The GSLB component of the CDN takes into account the client location, the edge SN (Service Node-CDN Node in Figure2) location, and the edge SN loads to find a suitable edge SN for this client. The client is then redirected to these edge SNs using traditional DNS-based redirection techniques.

(3)The edge SN serves a new LiveSky-enabled client in CDN-mode.

(3') The edge SN serves as a tracker for the P2P operation to bootstrap new clients with candidate peers.

(4') And then the peer node will request and download content from other peer nodes.

In [21], Hung-Chang Yang et al. proposed a two step selection PAC approach on landmark-based selection algorithm to find the nearest replica-cache server (CDN Node) and peer-caches to download content in this architecture. Figure3 describes the main mechanisms.

(1) The first step also depends on domain name server (DNS) to find the nearest replica-cache server. When the end user wants to get the content distribution service site via querying the local domain name server, the content provider server issues the measure messages to the CDN-DNS which act as landmarks. The local domain name server starts up the querying process to the CDN-DNS, and the CDN-DNS will measure the round-trip time between the CDN-DNS and the local domain name server. Note Figure3 merges the interaction between local DNS and CDN-DNS.

(2) When the local DNS server receives responses from all CDN-DNS, it replies to the end user which is the nearest replica-cache server of the local domain name server of that end user. If there are many replica-cache servers belonging to the same region, they can further use the Euclidean-Distance equation to compute which is the nearest replica-cache server.

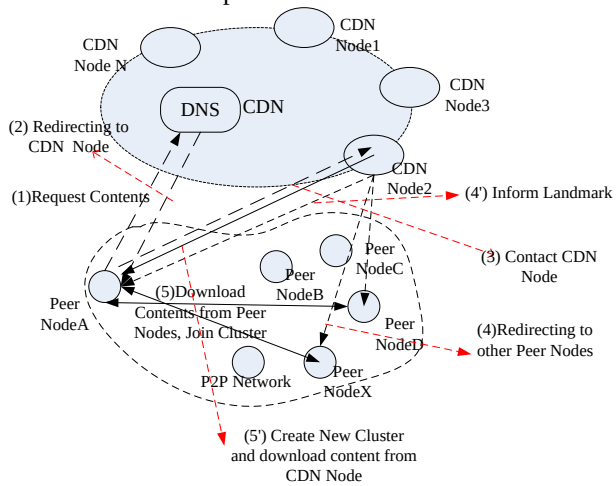


Figure3. Landmark-based two step selection approach

(3) At the second step, when the end users contact the nearest replica-cache server, the replica-cache server issues the measure messages to other peer-caches which act as landmarks. The landmark will measure the round-trip time between the landmark and the end user.

(4) When the replica-cache server receives responses from all landmarks, it replies to the end user which is the nearest peer-cache. (5) The end user will send join message to the nearest peer-cache and the nearest peer-cache will add the end user to the cluster member list. The end user downloads contents from the peer-cache.

(4') Otherwise, if the end user is out scope of the nearest peer-cache, the end user will create a cluster for itself and inform the nearest replica-cache server.

(5')Further, the end user will act as a cluster leader to inspire new peers, and download contents from the replica-cache.

Our previous work [12] proposed a PAC based PeerCDN architecture. Just as Figure4 described, PeerCDN is a two-layer streaming architecture. The upper layer is a CDN framework layer which is composed of original CDN servers, including origin servers and replica servers. The lower layer is called Peer node layer, which consists of multiple groups of clients who request the streaming services, each client is considered a peer node in the group. The connecting point between the CDN framework layer and the Peer node layer is located in the nearest replica server, we call it the Strong Node. It is the Strong Node's duty to coordinate the content direction for each request. In order to prevent the fail-over of that server, other nearby strong nodes can also be candidate leaders, if the nearest strong node fails, one of the candidate leaders will be selected to become the new leader of that group.

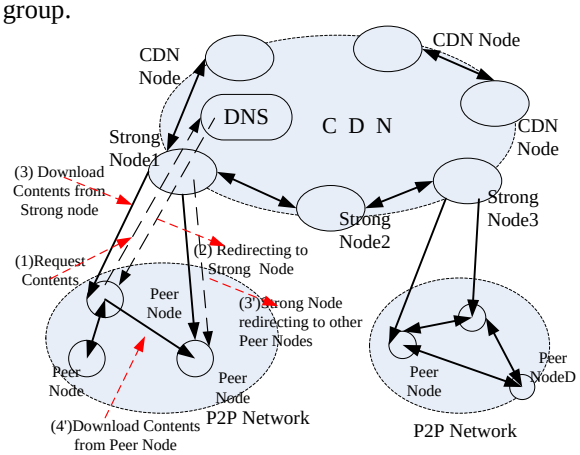


Figure4. PeerCDN hybrid overlay architecture

Figure 4 illustrates content request flow; strong node 2 is the candidate of strong node 1 and strong node 3.

(1) One peer node requests content from DNS (here merging the interaction between local DNS and CDN-DNS).

(2) DNS Redirects to Strong Node.

(3) Download content from Strong node.

(3')Optionally, Strong Node redirects to other Peer Nodes.

(4')Download Contents from selected Peer Nodes.

(2) CAP Method

Currently, in order to improve service quality, many P2P application providers (e.g. PPLive[3], UUSee[36], Zatoo[37], CoolStreaming[38]), either use their own servers or use third-party CDN to reinforce their P2P system. We summarize this method as CDN-aided P2P system (CAP). In this method, CDN is served as a backup system for rescuing. When a peer joins the overlay network, it will be redirected to the common P2P network first by a P2P tracker. If the peer cannot obtain enough content from the P2P network, it will ask the tracker for help and be redirected to a CDN to fetch required data.

(3) Other Methods

In [19], the architecture of HCDN can be further abstracted to a two-level hierarchical hybrid model. The process of content delivery includes two stages: the CDN-level and the P2P-level. In a backbone network, the CDN system is deployed and the content is strategically disseminated on surrogate servers. In an access network, the centralized P2P system is introduced, and the user nodes can exchange content between each other. So, user nodes can concurrently get content from both surrogate servers and other user nodes. In a CDN-level backbone network, the surrogate server is a logical entity and may consist of multiple physical servers or clusters. The P2P-level access network is an overlay network. The peers may be geographically separate and the access networks may be overlapped.

Figure 5 illustrates the architecture of HCDN. They didn't point out the order of CDN and P2P request routing. They pointed out that the request routing in traditional CDN, such as the iterative or recursive scheme, can be also adopted in HCDN. In P2P system implementations, the range of indexing service can be decided by request routing (e.g. an intelligent DNS scheme) or selected manually by user nodes.

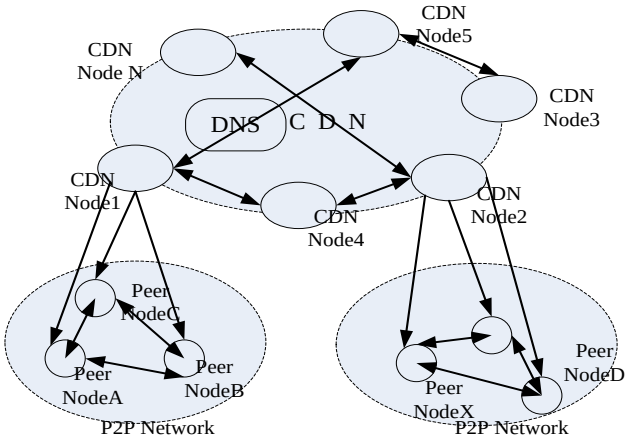


Figure5. The architecture of HCDN

In [18], they do not describe the overlay route hybrid mechanisms in detail, but they assume that the peers only share with others in the same CDN service region. In this way, the peers assist the CDN, but do not create any additional cross region traffic compared with a pure-CDN solution. Therefore, their architecture is similar to the architecture of HCDN.

When we study the hybrid relationship between CDN and P2P technologies, we need to consider the mainstream delivery mechanisms of current P2P streaming systems. Existing approaches for live P2P streaming can be generally divided into two classes: tree-based approaches and mesh-based approaches [24]. Tree-based approaches use push-based content delivery over multiple tree-shaped overlays, and mesh-based approaches use pull-based content fetching method from a randomly connected mesh. A tree is probably the most natural structure for a multicast overlay, but is vulnerable in the presence of dynamic end-hosts [23]. Data-driven approaches form a mesh out of overlay nodes to exchange data, which greatly enhances their resilience. It however suffers from an efficiency-latency tradeoff, given that the data have to be pulled from mesh neighbors with periodical notifications. In general, CDN is more suitable to support the tree and push based content delivery method.

Both methods have advantages and disadvantages, and neither can completely solve the P2P delivery task. This motivates recent research for a tree-mesh hybrid delivery method. Feng Wang et al. [23] suggested a novel hybrid tree/mesh design - mTreebone that leverages both overlays.

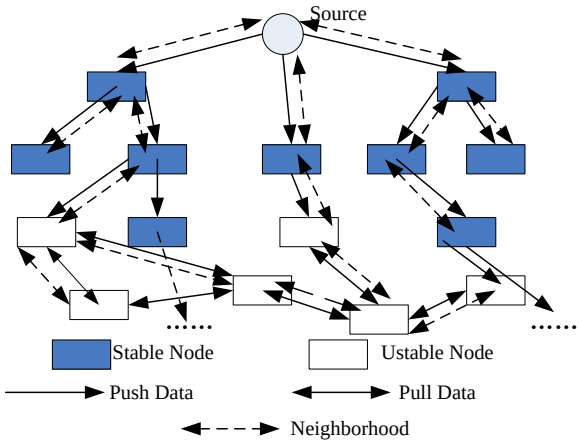


Figure6. mTreebone framework [23]

Just as shown in Figure6, the key idea of mTreebone is to identify a set of stable nodes to construct a tree-based backbone, called treebone, with most of the data being pushed over this backbone. The source of the treebone can be a CDN node. These stable nodes, together with other non-stable nodes, are further organized through an auxiliary mesh overlay, which facilitates the treebone to accommodate node dynamics and fully exploit the available bandwidth between overlay nodes. To realize such a hybrid overlay for live streaming, the author described that the

following unique and important issues have to be addressed [23]. First, they have to identify the stable nodes in the overlay; second, they have to position the stable nodes to form the treebone, which should also evolve to optimize its data delivery; third, they have to reconcile the treebone and the mesh overlays, so as to fully explore their potentials.

In some cases, CDN and P2P integration need to reconcile tree-based and mesh-based overlays. The most typical example is that LiveSky adopts a similar hybrid approach combining the multitree and mesh schemes to achieve both efficient delivery and robustness to churn [17]. Peers are organized in a tree-based overlay on a per-substream basis. This ensures that all nodes contribute some upload bandwidth. Additionally, in order to be robust to network or node failures, peers also use a mesh-style pull mechanism to retrieve missing frames for continuous playback [17].

B. Playing Buffer Hybrid Issues

(1)Introduction of Buffer Mechanism

The buffer at each peer represents a sliding window of the media channel, containing blocks to be played in the immediate future.

In the tree-based approach, an overlay construction mechanism organizes participating peers into a single tree or multiple trees. Each peer determines a proper number of trees to join based on its access link bandwidth [24]. The content delivery is a simple push mechanism from the tree's upper layer to the tree's lower layer node's playing buffer.

In the mesh-based approach, participating peers form a randomly connected overlay, or a mesh. Each peer tries to maintain a certain number of parents (i.e., incoming degree) and also serves a specific number of child peers (i.e., outgoing degree) [24]. Upon arrival, a peer contacts a bootstrapping node to receive a set of peers that can potentially serve as parents, and then the peer will pull contents from some selected parents into its playing buffer.

(2) Buffer Design Mechanism Comparison between Live Streaming and Vod Streaming

Another important issue to consider is, applying CDN-P2P hybrid techniques into live streaming and VoD streaming is different due to the following fundamental differences between the two types of streaming [6]. First, end-to-end delay is more important for live streaming than VoD streaming. In live streaming, the shorter the end-to-end delay is, the more lively the stream is perceived by the users (defined as liveness in). In VoD streaming, liveness is simply irrelevant because the video stream is already pre-recorded. Second, a user joining an on-going live streaming session is only interested in the stream starting from his/her joining time, while in the VoD streaming case the whole video must be delivered to the new user, and VOD streaming allows users to execute VCR-like commands such as "dragging the progress bar back and forth", "forward", "rewind". If a user drags the progress bar to a new position, he is interested in the stream starting from this new position. Therefore, in live streaming, we can determine the period from the user joining time by adding 20 or 30 seconds as an emergency zone, and the emergency zone data can depend on the CDN. However, in VoD streaming, when one user drags the progress bar to a new position, we should consider how to adjust the beginning position of emergency zone to this new position as soon as possible, this will increase the risk of re-buffering delay.

(3) Buffer Design Comparison

In the CDN-P2P hybrid scenario, the client's playing buffer may be filled via the push-pull hybrid mode. The following compares such kinds of hybrid modes.

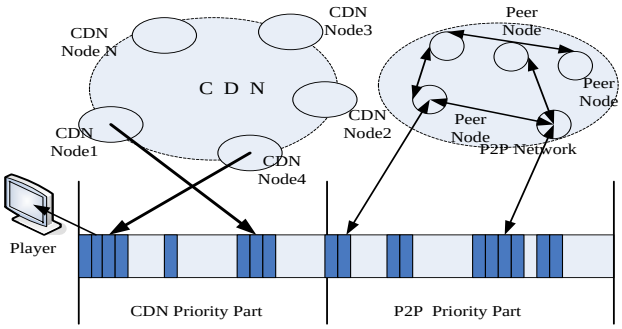


Figure7. Application layer hybrid buffer design

In [20], different from other hybrid CDN-P2P systems, their mechanism works not on hybrid overlay layer but on application level so it is easier to deploy and does not need any modification at the ISP's side. Just as Figure 7 illustrates, they proposed a new hybrid CDN P2P mechanism which acts on the effective management of the playing buffer at the client-side. This solution is motivated by: data slots in the buffer which are near the play side must be ready before the scheduled playing times and data slots in the rest of the buffer can be filled later. Therefore, the part which is near the play side has higher priority for filling data and the others have less priority. From that, they simply treat the playing buffer as two parts: the CDN server's priority part and the priority part for peers in the P2P network. The CDN server priority part is the part of the buffer whose data slot must be received as soon as possible to assure the media playback begins in time. On the other hand, the P2P network priority is a part of the buffer whose data slot can be received later. The client will pull contents from the CDN node into its CDN priority part buffer, and then it will pull contents from peer nodes into its P2P priority part buffer. By contrast, mTreebone adopted a different hybrid buffer management method.

In mTreebone, the data blocks are delivered by two means. In general, they are pushed over the treebone. And if a gap appears in the stream received by a node, due to either temporal capacity fluctuation in the treebone or node dynamics as discussed later, the node

may pull the missed blocks through the mesh overlay. They introduce a seamless push/pull buffer that coordinates the treebone and the mesh to make data delivery efficient yet resilient against failure [23].

Figure8 illustrates push/pull switching, where a tree-push pointer is used to indicate the latest data block delivered by the push method, and a mesh-pull window facilitates the pull delivery [23]. When a node is temporarily disconnected from the treebone, its tree-push pointer will be disabled and only the mesh-pull window works to fetch data from its mesh neighbors. When it connects to the treebone again, the tree-push pointer will be re-activated. The mesh-pull window is always kept behind the tree-push pointer so as not to request data currently being delivered by the treebone. Therefore, no duplicated data blocks are received from both the treebone and the mesh. By comparing these two methods above, we find mTreebone is more flexible than the above Application Layer Hybrid method, which adopts a more static method to divide the buffer. But mTreebone's buffer is more dependent on the tree, while Application Layer Hybrid method only depends on the CDN node in the playing start-up phase, which can reduce pressure on the CDN.

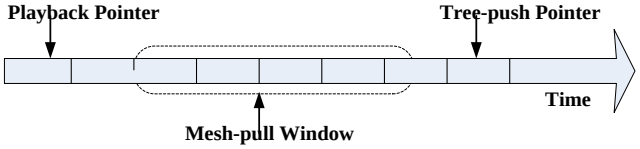


Figure8. mTreebone push/pull switch buffer design [23]

(4) Buffer Data Block Size Comparison between CDN and P2P

In the CDN-P2P hybrid scenario, an important issue to consider is the size of data blocks. In general, the data block size of CDN and P2P delivery is different. For example, UUSee Inc. is one of the leading P2P multimedia solution providers in mainland China, it simultaneously broadcasts over 800 live streaming channels to millions of peers, mostly encoded to high quality streams around 500 Kbps. The peer buffer size in UUSee is 500 media blocks, and each block represents 1/3 second of media playback [5][36]. So we can calculate that the size of each block is approximately 20.8KB. But in each transmission, CDN is capable of providing larger size blocks, such as 1MB- 16 seconds data for each request. If so, we need to consider CDN and P2P distributing different sizes of blocks to fill the buffer. Now most of the hybrid schemes do not yet consider this issue.

V. CDN-P2P-HYBRID MODEL ANALYSIS AND DESIGN

A. Tightly-coupled Hybrid Model vs Loosely-coupled Hybrid Model

In the current multimedia technology field, it is a key problem to establish a new-generation content delivery solution, which supports large-scale accessing, and can be cost-effectively extended, effectively monitored, QoS guaranteed, network-friendly, regional controllable, and safe and reliable. CDN and P2P are two mainstream content delivery technologies in the current Internet, but constrained by the computing model, both of them have their own advantages, while there are some fundamental shortcomings. The two models have great complementation, so their integration is an important trend. We need to research CDN and P2P hybrid delivery model for realizing controllable, manageable, credible, network-friendly content distribution technology, exploring difficult problems of CDN-P2P hybrid distribution technology. We find there are two kinds of CDN-P2P hybrid models: the tightly-coupled hybrid model vs. the loosely-coupled hybrid model.

B. Tightly-coupled Hybrid 1+1 Model

We think the tightly-coupled hybrid model is 1+1 model. It means one CDN system can only integrate with one P2P system. Just as Figure 9 shows, CDN and P2P have an overlap area, where CDN nodes and P2P nodes closely collaborate to execute content delivery function, including the CDN nodes acting as a tracker to involve in the construction process of the P2P overlay network; CDN nodes manage and guide P2P nodes to realize ISP-friendly P2P delivery, CDN nodes and P2P nodes collaborate on content delivery [34].

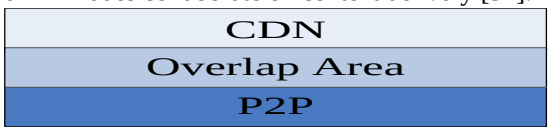


Figure 9. Tightly-coupled hybrid 1+1 model [34]

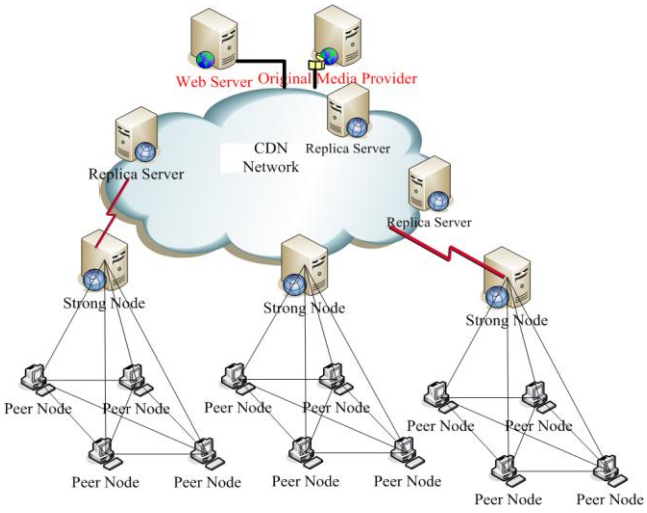


Figure 10. PeerCDN CDN-P2P hybrid architecture [12]

As Figure 10 shown, our PeerCDN [12] is one kind of typical 1+1 tightly-coupled hybrid model. PeerCDN realizes the management of regional autonomy when constructing the overlay network. The overlay network is constructed geographically to become a topology-aware overlay network through redirecting of strong

nodes. In our PeerCDN architecture, each group of client peers is led by the nearest Strong Node.

In this model, the P2P system is attached to the CDN system. In other words, CDN nodes lead to build P2P systems. This model is efficient for one CDN integrating with one P2P, but it is not suitable for one CDN integrating with multiple P2P. For example, PPLive and PPStream are two of the largest P2P streaming media operators, if PPLive and PPStream want to integrate with a CDN through such method, PPLive and PPStream have to break their current overlay construction and data transmission mechanisms to adapt to the CDN, which will increase the integration difficulty.

C. Loosely-coupled Hybrid 1+N Model

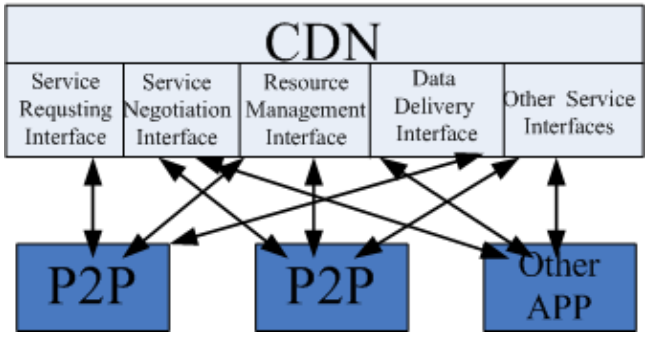


Figure 11. Loosely-coupled hybrid 1+N model

The popularity of Cloud Computing technology calls for Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). Therefore, cloud computing era requires content services available on-demand and be utilized in an open and loosely-coupled fashion. If a CDN operator wants to integrate with a number of heterogeneous P2P operators, it needs to expose a series of public service interfaces to serve multiple P2P operators [34], including the service requesting, service negotiation interface, data delivery service interface, resource management interface and other interfaces. This kind of integration is a loosely-coupled 1+N hybrid model. Loosely-coupled hybrid model means that one CDN can provide services for multiple P2P systems with public service interfaces, rather than being tightly integrated with only one P2P system, just as Figure 11 shows. CDN is not directly involved in the construction of P2P overlay network, and CDN does not directly lead P2P systems, but when one P2P system request service from CDN, it can send request to CDN interface, and negotiate SLA with CDN. And finally P2P can use some CDN nodes' data delivery service when an emergency or flash crowd happens.

For example, China Telecom, as China's largest network operators, is building a large-scale CDN system, they want to make their CDN able to provide services for more P2P or other streaming operators (PPLive, PPStream, UUSee et al.) in parallel, such as content accelerating services. Therefore, they need such a loosely-coupled 1+N hybrid model. At the same time, 1+N hybrid model can be easily extended into N+N hybrid model.

Web Services interfaces are neutral for platform and technology, so application business objects can integrate easily through Web Services. Web Services technology has been verified to well support enterprise application integration (EAI) and B2B integration (B2Bi) solution. In order to improve our PeerCDN architecture, we proposed WS-CDSP [25]: a novel Web Services-based Content Delivery Service Peering Scheme, which can support loosely-coupled multimedia content peering delivery service architecture. WS-CDSP belongs to such loosely-coupled 1+N hybrid models. The main function is to enable explicit cooperation and integration among P2P, CDN and VoD. The main components in the novel architecture are described in the Figure 12. Note that the curve circle of every P2P system in the figure represents different organization's P2P, not different autonomous regions of the same P2P system, which is different with PeerCDN.

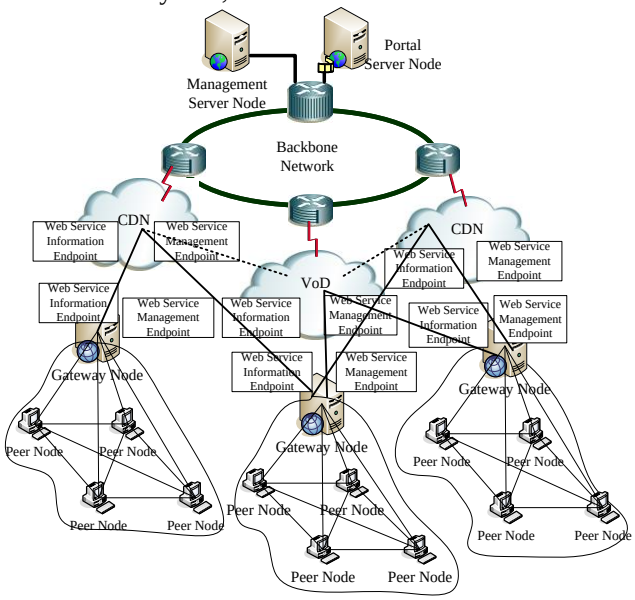


Figure12. WS-CDSP: web services-based content delivery service peering architecture [25]

In order to support explicit cooperation between P2P, CDN and VoD, every P2P, CDN and VoD system should provide two endpoints: one is the Web Service Information Endpoint; the other is the Web Service Management Endpoint [25]. The Web Service Information Endpoint can act as black box interface to expose the system information of CDN ,VoD, or P2P. CDN and VoD system can provide cache or VoD server node information, topology structure information, and media content item information through Web Service Information Endpoint and all these information can be described with XML and exposed by WSDL [25]. The Web Service Management Endpoint provides a manageable interface, and the platform manager can request the running load status of CDN ,VoD, or P2P, or subscribe to the events [25]. CDN and P2P

technologies can be loosely integrated through these interfaces.

In order to construct open service relationship, we have proposed SNEGO: CDN-P2P loosely-coupled SLA negotiation model in [34]. As Figure13 described, CDN firstly provides an open and standard-based agreement interface for the P2P and other applications to negotiate with it. We use WS-Agreement for establishing service agreement between two CDN and P2P. This scheme also allows CDN easily integrated with other application.

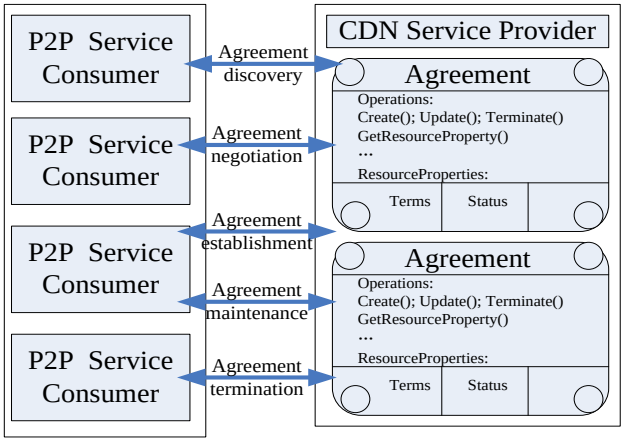


Figure 13. CDN-P2P loosely-coupled SLA negotiation model [34]

D. Comparison of Current CDN-P2P-hybrid Schemes

Furthermore, we compare the abovementioned current CDN-P2P-hybrid schemes in term of tightly coupled model and loosely coupled model in Table II.

TABLE II. COMPARISON OF CURRENT CDN-P2P-HYBRID SCHEMES

Current CDN-P2P-hybrid Scheme	Hybrid Mode	Whether CDN nodes are involved in P2P content delivery	Whether CDN nodes are involved in P2P Overlay Construction
Dongyan Xu's Scheme	Tightly Coupled	Yes	No
PeerCDN	Tightly Coupled	Yes	Yes
Ono	Loosely Coupled	No	Yes, but passively
PSP	Tightly Coupled	Yes	No
PACDN	Tightly Coupled	Yes	Yes
LiveSky	Tightly Coupled	Yes	Yes
Akamai CDN-P2P	Tightly Coupled	Yes	Yes
HCDN	Tightly Coupled	Yes	No
WS-CDSP	Loosely Coupled	Yes	No
SNEGO	Loosely Coupled	Yes	No

E. Common CDN-P2P-Hybrid Delivery Models

In both the tightly-coupled and loosely-coupled models, we think that there are some common models that need to be further studied in the future work:

(1)The hybrid-scheduling model between CDN global scheduling and P2P distributed scheduling: CDN utilizes GSLB to realize global scheduling, and P2P streaming utilizes Tracker and local decision to realize distributed scheduling. CDN and P2P both have content service capabilities, but their capabilities have different characteristics, therefore what we need to solve is how to integrate the CDN GSLB scheduling mechanism with the P2P distributed scheduling mechanism to achieve mixed scheduling and united service. And this kind of mixed service can achieve scalability through P2P services, while at the same time, ensuring service reliability through CDN. In the CDN-P2P hybrid architecture, we need to study the controllable scheduling model, specifically study how to integrate the CDN's strengths of global scheduling and centralized management with P2P's characteristics of distributed scheduling and flexible extension.

(2) Total hybrid-delivery services capacity dynamic allocation model: CDN node service capacity is different in the traditional CDN scenario and the CDN-P2P hybrid scenario. In traditional CDN scenario, CDN node service capacity is fixed, for example, if one CDN node is bound to concurrently serving 500 users with 300kbps for every user, the maximum capacity is 500 as client nodes consecutively request the CDN Node more than tens of minutes. But in the CDN-P2P hybrid scenario, peer nodes only occasionally utilize the CDN Node for tens of seconds, so the CDN node can serve more than 500 users. So we can dynamically extend the CDN node service capacity through using some wise scheduling algorithms. At the same time, the P2P streaming system service capacity can be extended easily, but it lacks the stability of service, and it can not assure the quality of service. Therefore, in the CDN-P2P hybrid scenario, we can optimize the allocation mechanism to do a seamless hand over process to balance the load of two content service systems (CDN node and P2P node). By dynamically and flexibly allocating the amount of CDN and P2P services, we can achieve "1 +1 > 2" results, which means the intelligent CDN-P2P hybrid service capacity is larger than the simple sum of CDN and P2P's respective capacities.

(3) P2P-ISP friendly delivery model under the guidance of CDN: CDN has realized the network-friendly feature by optimizing cache node deployment and redirecting users in accordance with the nearby principle. For the ISP and the backbone network operators, a significant drawback of P2P is cross-ISP traffic and unlimited, greedy consumption of network bandwidth. This produces a significant impact on the ISP network. How to design a Network-friendly P2P distribution model with the participation of CDN node is a key problem, which the CDN-P2P hybrid model

needs to solve. CDN and P2P can cooperate to make peer node locality-aware to fast find nearby stored copies of the requested content.

(4) Controllable management model: the decentralized feature of a P2P network makes it lack management capacity, for the management of end-users, churn management, and content availability management. The CDN system typically has complete management capabilities, including cache node management, content management, AAA (Authentication, Authorization, and Accounting), business and network management, etc. How to make the entire CDN-P2P hybrid system manageable through integration is a key problem. Specifically, controllable management model should focus on making CDN help mange peer node's churn behavior (i.e., failure, leave, join) and increase total robustness in the highly dynamic P2P environment.

(5) Secure and trustable delivery model: some issues such as identity authentication and content protection are also key problem that need to be considered. How to integrate CDN centralized authentication and P2P decentralized authentication, how to combine CDN secure and controllable content distribution and P2P flexible and free content distribution and overcome their shortcomings. This needs to study a secure and reliable CDN-P2P hybrid distribution model.

VI. CONCLUSION AND FUTURE WORK

CDN and P2P are two mainstream content delivery technologies in the current Internet, but constrained by the computing and service model, both of them have advantages and fundamental disadvantages. The two technologies have much complementary features; therefore their integration is very necessary and helpful to build a new-generation efficient, QoS-guaranteed, and large-scale content service system. However, their integration is not simple. A common streaming content distribution platform is not a simple combination of CDN and P2P, rather the integration of CDN and P2P needs to solve many problems. At present, research and development work has been started; in particular, we need to study CDN and P2P integration mechanisms from the key issues, basic models, and key algorithms.

CDN and P2P integration research is still in developing stage. In this paper, we reviewed the related research work of this field in recent years, and then we analyze the key issues that need to be solved, such as overlay route hybrid issues, playing buffer hybrid issues, and so on. In order to realize the mixed mode of P2P extending CDN and CDN guaranteeing P2P, we point out there are some reliable, secure and friendly hybrid delivery models that need to be researched in depth.

In our future work, we will focus on designing these key hybrid models and related algorithms, and developing some prototype systems to experimentally verify the advanced features of these models and algorithms. Finally, we plan to carry out larger scale experiments with our prototype and make a real deployment in our ongoing CNGI (China Next Generation Internet) project- National Higher Education Conference Video Resources Sharing Project.

REFERENCES

[1] Ao-Jan Su, David R. Choffnes, Aleksandar Kuzmanovic, and Fabián E. Bustamante, "Drafting behind Akamai (Travelocity-based detouring)", SIGCOMM'06, pp.435-446.

[2] C. Huang, A. Wang, J. Li, K.W. Ross, "Measuring and evaluating large-scale CDNs", Internet Measurement Conference (IMC) 2008.

[3] Yan Huang, Tom Z. J. Fu, Dah-Ming Chiu, "Challenges, Design and Analysis of a Large-scale P2P-VoD System", SIGCOMM'08, pp.375-388.

[4] Jiajun Wang, Cheng Huang, Jin Li, "On ISP-friendly rate allocation for peer-assisted VoD", ACM Multimedia 2008, pp. 279-288.

[5] Zimu Liu, Chuan Wu,Baochun Li, Shuqiao Zhao, "Distilling superior peers in large-scale P2P streaming systems", IEEE INFOCOM 2009, pp. 82-90.

[6] Tai T.Do, Kien A. Hua, Mounir A. Tantaoui, "P2VoD: Providing Fault Tolerant Video-on-Demand Streaming in Peer-to-Peer Environment", IEEE International Conference on Communications, 2004, pp. 1467-1472.

[7] T. Karagiannis, P. Rodriguez, and D. Papagiannaki, "Should Internet service providers fear peer-assisted content distribution?", ACM/USENIX Internet Measurement Conference, IMC'2005, pp. 63-76.

[8] Thomas Bonald, Laurent Massouli, Fabien Mathieu, Diego Perino, Andrew Twigg , "Epidemic live streaming: optimal performance trade-offs", SIGMETRICS 2008, pp. 325-336.

[9] Huang, C., Li, J., & Ross, K. W., "Can internet video-on-demand be profitable?", SIGCOMM'07, pp.133-144.

[10] Li, Jin .et al., "On peer-to-peer (P2P) content delivery", Springer Journal on Peer-to-Peer Networking and Applications, 2008 , pp. 45-63.

[11] Dongyan Xu, Sunil Suresh Kulkarni, Catherine Rosenberg, Heung-Keung Chai, "Analysis of a CDN–P2P hybrid architecture for cost-effective streaming media distribution", Multimedia Systems (2006) 11(4), pp.383–399.

[12] Jie Wu, ZhiHui lu, BiSheng Liu, ShiYong Zhang, "PeerCDN: a novel P2P network assisted streaming content delivery network scheme", Proceeding of IEEE CIT2008, IEEE Computer Society, pp. 601-606.

[13] Haiyong Xie,Y. Richard Yang,Arvind Krishnamurthy et al., "P4P: provider portal for applications" ,SIGCOMM'08, pp. 351-362.

[14] David R. Choffnes and Fabian E. Bustamante, "Taming the torrent a practical approach to reducing cross-ISP traffic in peer-to-peer systems", SIGCOMM'08, pp.363-374.

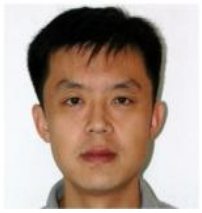
[15] Zhijia Chen, Hao Yin, Chuang Lin, Xuening Liu, Yang Chen, "Towards a trustworthy and controllable peer-server-peer media streaming: an analytical study and an industrial perspective", GLOBECOM 2007, pp.2086-2090.

[16] Xuening Liu, Hao Yin, Chuang Lin, Yu Liu, Zhijia Chen, Xin Xiao, "Performance analysis and industrial practice of peer-assisted content distribution network for

large-scale live video streaming", AINA 2008, pp. 568-574.
[17] Hao Yin, Xuening Liu, Tongyu Zhan, Vyas Sekar, Hui Zhang, "Design and deployment of a hybrid CDN-P2P system for live video streaming: experiences with LiveSky", ACM Multimedia 2009, pp.25-34.
[18] Cheng Huang,Angela Wang, "Understanding hybrid CDN-P2P:why limelight needs its own red swoosh", NOSSDAV '08, pp.75-80.
[19] Hai Jiang, Jun Li, Zhongcheng Li, et al, "Efficient large-scale content distribution with combination of CDN and P2P networks", International Journal of Hybrid Information Technology Vol.2,No.2, 4, 2009, pp. 13-24.
[20] Duyen Hoa HA,Thomas Silverton,Olivier Fourmaux, "A novel hybrid CDN-P2P mechanism for effective real-time media streaming", www-rp.lip6.fr/~fourmaux/Stages/HA.ACM_Rapport.pdf.
[21] Hung-Chang Yang, Min-Yi Hsieh, Hsiang-Fu Yu, Li-Ming Tseng, "A replication-aware CDN-P2P architecture based on two-step server selection and network coding", PCM 2008, pp. 738-747.
[22] Michael M.Afergan, et al., "Hybrid content delivery network (CDN) and peer-to-peer (P2P) network", United States Patent Application 20080155061.26, Assignee: Akamai, www.freepatentsonline.com/y2008/0155061.html,Jun 2008.
[23] Feng Wang, Yongqiang Xiong, Jiangchuan Liu, "mTreebone: a hybrid tree/mesh overlay for application-layer live video multicast", IEEE ICDCS 2007, pp. 49-56.
[24] Nazanin Magharei, Reza Rejaie,Yang Guo, "Mesh or multiple-tree: a comparative study of live P2P streaming approaches", IEEE INFOCOM 2007, pp.1424-1432.
[25] ZhiHui Lu Jie Wu Chuan Xiao WeiMing Fu YiPing Zhong, "WS-CDSP: a novel web services-based content delivery service peering scheme", Proceeding of 2009 IEEE SCC 2009, IEEE Computer Society, pp.348-355.
[26] Akamai,http://www.akamai.com/html/technology/index.html.
[27] Mark Hendrickson, "Cisco Invests in Hybrid Content Delivery Network-GridNetworks", http://www.techcrunch.com/2008/03/10/cisco-invests-in-hybrid-content-delivery-network-gridnetworks, Mar, 2008.
[28] Om Malik, "Grid Joins the P2P CDN Party",http://gigaom.com/2007/10/30/grid-networks/, 2007.
[29] Liz Gannes, "Cisco to Distribute Grid Networks' P2P Client",http://newteevee.com/2008/03/16/cisco-to-distribute-grid-networks-p2p-client/,March 16, 2008.
[30] CDNs and P2P, http://www.pandonetworks.com/node/185,October 10, 2007.
[31] Dan Rayburn, "Akamai and Limelight To Deploy P2P For Higher Quality, Not Cost Savings", http://blog.streamingmedia.com/the_business_of_online_vi/,July 02, 2009.
[32] Jimmy Jernberg, Vladimir Vlassov, Ali Ghodsi, Seif Haridi, "DOH: a content delivery peer-to-peer network", Euro-Par 2006, pp.1026-1039.
[33] Manal El Dick, Esther Pacitti, Bettina Kemme, "Flower-CDN: a hybrid P2P overlay for efficient query processing in CDN", ACM EDBT 2009, pp. 427-438.
[34] ZhiHui Lu,Jie Wu, WeiMing Fu, "Towards a novel web services standard-supported CDN-P2P loosely-coupled hybrid and management model", 2010 IEEE International Conference on Services Computing (SCC 2010), IEEE Computer Society, pp.297-304.
[35] Hao Yin, Xuening Liu, Tongyu Zhan, Vyas Sekar, Feng Qiu, Chuang Lin, Hui Zhang, Bo Li, "LiveSky: enhancing CDN with P2P",ACM Transactions on Multimedia Computing, Communications, and Applications,Volume 6 Issue 3, 2010, pp. 16-34.
[36] Z. Liu, C. Wu, B. Li, and S. Zhao, "UUSee: large-scale operational on-demand streaming with random network coding", INFOCOM 2010, pp.2070-2078.
[37] Chang, H., Jamin, S., Wang, W., "Live streaming performance of the Zattoo network," Internet Measurement Conference (IMC2009), pp. 417-429.
[38] Bo Li, Susu Xie, Yang Qu, Keung, G.Y., Chuang Lin, Jiangchuan Liu, Xinyan Zhang, "Inside the new Coolstreaming: principles, measurements and performance implications", INFOCOM 2008, pp.1031-1039.
[39] ZhiHui Lu, et al., "Scalable and reliable live streaming service through coordinating CDN and P2P", Proceeding of IEEE 17th International Conference on Parallel and Distributed Systems, IEEE ICPADS2011, pp.581-588.

ZhiHui Lu received a Ph.D degree from Fudan University in 2004, and now is an Assistant Professor in School of Computer Science, Fudan University. He is a member of the IEEE and the IEEE Computer Society. His research interests are network multimedia technology, content delivery network, P2P streaming technology and service computing.

Ye Wang is a Ph.D. candidate in Computer Science, Yale University. His advisor is Prof. Y. Richard Yang. And his research group is Laboratory of Networked Systems. He received my B.E. and M.S. degree in Electronic Engineering from Tsinghua University (Beijing, China) in 2005 and 2007. His current research interests include computer networking, P2P streaming system.



Yang Richard Yang is an Associate Professor of Computer Science and Electrical Engineering at Yale University. His current research interests include computer networks, multimedia content delivery, mobile computing, wireless networking, and network security. He leads the Laboratory of Networked Systems (LANS) at Yale. He has served as a committee member of many conferences, as a panelist of several funding agencies, as an advisor of several industrial and academic organizations. Dr. Yang's research is supported by both government funding agencies such as NSF and industrial companies such as Microsoft, Huawei

Generalized Multi-Constrained Path (G_MCP) QoS Routing Algorithm for Mobile Ad hoc Networks

Kunagorn Kunavut and Teerapat Sanguankotchakorn
Telecommunications Field of Study
School of Engineering and Technology
Asian Institute of Technology, Thailand
Email: {kunagorn.kunavut, teerapat}@ait.ac.th

***Abstract*— In mobile ad hoc network, efficient routing protocol is required to perform route discovery and maintenance. These protocols can be classified into two main types which are proactive and reactive routing protocols. Most of them usually use the suboptimal path to reach destination without considering QoS parameters. This results in network congestion during high traffic load situation. Hence, many algorithms have been proposed to offer QoS routing to these protocols. However, most of them find the feasible path by using only one or two QoS metrics. This is not enough to support many applications with QoS guaranteed, especially multimedia applications since they have more stringent various QoS requirements. To provide QoS routing in ad hoc network based on such environment, we propose the effective algorithm called Generalized MCP (G_MCP) to find the feasible path based on proposed weighted Connectivity Index (combination of link connectivity and capacity) and non-linear cost (combination of multiple additive QoS metrics using non-linear function). We adopt the fall-back approach. That is, G_MCP will find the path from source to destination by considering weighted Connectivity Index first. If there is a tie, the path with least non-linear cost will be chosen. Based on this approach, G_MCP has the comparable time complexity with the Shortest-Widest Path algorithm. We construct the simulation in a number of scenarios based on proactive protocol called OLSR. According to the simulation results, it is obvious that G_MCP performances are superior than OLSR and Shortest-Widest Path algorithms in terms of throughput, packet delivery ratio, delay and success ratio. Therefore, it can be concluded that G_MCP is able to support various applications and be operated well in highly dynamic mobility environment.**

***Index Terms*— Connectivity Index, non-linear cost, multi-constrained path, QoS routing, mobile ad hoc network**

I. Introduction

Mobile ad hoc network (MANET) is an infrastructure-less wireless network. Each node in this network behaves like a router to find paths and also forward packets to destinations by using effective routing protocols. These protocols can be approximately classified into two main types which are proactive and reactive protocols. In proactive or table-driven protocols, every nodes maintain their routing tables by periodically exchanging routing information. The examples of proactive protocol are DSDV [1], TBRPF [2] and OLSR [3]. In reactive or on-demand protocols, all nodes do not maintain their topology information. The paths to destinations are necessarily obtained through route discovery process only when they are required. AODV [4] and DSR [5] are the examples of this routing category.

Among these protocols, OLSR is effectively operable in heavy load or congested networks because of its ability to periodically compute paths to destinations [6]. OLSR consists of four main modules: neighbor sensing, message flooding via multipoint relay (MPR) nodes [7], topology information and path computation. Every nodes implementing OLSR collect Hello messages from the others in order to discover the 1-hop and 2-hop neighbors. It also floods the topology control (TC) messages to the others every predefined interval in order to build a partial topology of network. According to this function, only MPR nodes are allowed to forward TC messages to reduce duplicate transmissions of control overhead. However, OLSR generates more control overhead than reactive protocols. After creating partial topological information of the network, each node computes paths to destinations using path computation module which usually finds the shortest hop count paths.

Most of traditional ad hoc routing protocols usually select the suboptimal paths to reach destinations by using either delay or hop count as the routing metric. This causes the network to be easily congested during high load situation. Thus, many QoS routing algorithms [9]–[23] were proposed to allow nodes to find paths by considering various QoS parameters such as bandwidth, delay, reliability and etc. However, most of them apply only one or two QoS metrics which are mainly bandwidth and/or delay in path computation process. Hence, these algorithms are not suitable to support a large number of ubiquitous and different QoS required applications (see Table I).

In general, QoS metrics can be roughly classified into two types which are additive and non-additive QoS metrics. In case of additive QoS metrics, an end-to-end cost of the path is determined by the sum of the individual link cost along the path from source to destination. Whereas the non-additive QoS metrics of a path are given by

Manuscript received November 17, 2011; accepted December 17, 2011

doi:10.4304/jcm.7.3.246-257

TABLE I.
QOS REQUIREMENTS OF COMMON APPLICATIONS [8]

Applications	QoS Requirements*		
	Bandwidth	Delay	Reliability
Web browsing	Medium	Medium	High
Email	Low	Low	High
FTP	Medium	Low	High
Telnet	Low	Medium	High
IP telephony	Low	High	Low
Video conference	High	High	Low

Note: *This is a relative comparison.

the minimum or maximum value of an individual link. Many research works [30]˘[32] have been proposed to solve multi-constrained path (MCP) problem in wireline network by combining multiple additive QoS metrics in a single mixed metric or cost. This combined cost is applied in path computation process as a routing metric to find the feasible paths to destinations. However, only additive QoS parameters are considered in path selection process of these algorithms. The non-additive QoS parameter such as bandwidth is left behind. In [23], the concrete method to apply MCP QoS routing in mobile ad hoc networks was demonstrated. But the considered QoS parameters are only additive type.

Hence, we propose the QoS routing algorithm called Generalized MCP (G_MCP) to find the feasible paths. We introduce weighted Connectivity Index (combination of link connectivity and capacity), and non-linear cost [30] (combination of multiple additives QoS metrics using non-linear function) as non-additive and additive QoS parameters in our proposed algorithm, respectively. Thus, it is capable to support variety of applications since multiple QoS constraints are put into account by combining them into mixed metrics. It is called G_MCP since it is flexible and open to be applied to any networks including both wireline and wireless networks. However, in this work, we consider this algorithm in only mobile ad hoc networks in order to ensure that they are capable to support QoS required applications and be able to operate in highly dynamic topology environment.

II. RELATED WORK

Most of traditional ad hoc routing protocols either proactive or reactive protocols lack capability to support QoS. Thus, many QoS routing algorithms have been researched in decades to offer QoS over ad hoc networks [9]˘[23]. These QoS routings can be classified into two main paradigms which are source and hop-by-hop QoS routings.

A. Source QoS Routings for Ad Hoc Networks

In source QoS routing, source node sends the request packets toward the destination node in order to gather global state information of the network and also monitor if the constrained path satisfies the QoS requirements or not. Consequently, the path will be selected if it has enough resources to support the required application.

In [9], Ticket-based QoS routing mechanism was proposed to find the feasible paths with enough resource to satisfy either delay or bandwidth. Flexible QoS Model for MANETs (FQMM) [10] offers QoS routing by performing additional QoS check function later after the routes to destinations are found to ensure that the generated traffic is not greater than the bandwidth specified in each traffic profile.

The Adaptive Dispersity QoS Routing (ADQR) protocol [11], an extension from the Signal Power Adaptive Fast Rerouting (SPAFAR) protocol [12], finds paths to destinations based on signal strength. However, if there is no single path satisfying bandwidth requirement, the routing algorithm will find multiple disjoint paths with longer-lived connections. The Ad hoc QoS On-demand Routing (AQOR) [13] provides QoS support by selecting the shortest end-to-end delay link satisfying bandwidth requirement.

In QoS-Aware Source-Initiated Ad hoc Routing (QuaSAR) [14], the QoS metrics incorporated in this routing algorithm are namely, battery power, signal strength, bandwidth and delay. The applications have opportunity to independently select the ranking of them since they have different QoS requirements. QoS-Aware Routing Based on Bandwidth Estimation for Mobile Ad hoc Networks (BEQR) [15] considers only bandwidth constraint during route discovery process. This algorithm offers two methods to estimate the available bandwidth which are Listen and Hello methods (see Table II for comparison of these source QoS routings).

B. Hop-by-Hop QoS Routings for Ad Hoc Networks

Algorithms applying hop-by-hop QoS routings locally calculate their own state information such as available bandwidth, signal strength, battery power, loss rate and etc., and distribute these information to the other nodes in the network. These information will be used by the others in path computation process to find the feasible paths to destinations satisfying QoS constraints.

A Core-Extraction Distributed Ad hoc Routing Algorithm (CEDAR) [16] creates core nodes for performing route computation. Once admissible route is set up via core node, shortest-widest path is selected among all available paths. In [17], Widest Path heuristic was proposed in OLSR to find the maximum bandwidth path to destination by modifying both MPR selection and route calculation processes. QoS-Enhanced OLSR Routing in Mobile Ad hoc Networks (QOLSR) [18] was also proposed to allow each node to use Shortest-Widest Path algorithm to find feasible paths to destination by selecting paths with maximum link bandwidth. If there is more than one widest path, a path with shortest delay will be chosen. Many works which adopted the idea of QOLSR [19], [20] were also proposed.

In [21], Widest-Shortest Path algorithm was proposed to offer interference-aware QoS routing by finding a path with minimum hop count. If there is a tie, the widest bandwidth link will be selected. In [22], the new QoS

TABLE II.
COMPARISON OF QOS ROUTING PROTOCOLS

Routing Protocol	Route Discover	QoS Routing Scheme	Routing Metric
Ticket-based [9]	Reactive	Source	Bandwidth, Delay and Cost
FQMM [10]	Reactive	Source	Bandwidth
ADQR [11]	Reactive	Source	Signal Strength and Bandwidth
AQOR [13]	Reactive	Source	Bandwidth and Delay
QuaSAR [14]	Reactive	Source	Battery Power, Signal Strength, Bandwidth and Delay
BEQR [15]	Reactive	Source	Bandwidth
CEDAR [16]	Proactive and Reactive	Hop-by-hop	Bandwidth and Hop Count
Widest Path [17]	Proactive	Hop-by-hop	Bandwidth
QOLSR (Shortest-Widest Path) [18]˜[20]	Proactive	Hop-by-hop	Bandwidth and Delay
Widest-Shortest Path [21]	Proactive	Hop-by-hop	Bandwidth and Hop Count
Shortest-Highest Path [22]	Proactive	Hop-by-hop	Weighted CI and Delay
MCP QoS Routing [23]	Proactive	Hop-by-hop	Multiple Additive QoS Metrics

routing metric was proposed to combine link connectivity denoted by Connectivity Index and link capacity into a single metric called weighted Connectivity Index (CI). The Shortest-Highest Path algorithm was also proposed to find the path with highest weighted CI and shortest end-to-end delay. In [23], the Multi-Constrained Path (MCP) QoS routing was proposed to select the feasible path based on only various additive QoS metrics (see Table II for comparison of these hop-by-hop QoS routings).

III. NOTATION AND PROPOSED QOS METRICS

A. Notation

A graph G, denoted by $G = (V, E)$, where V is the set of vertices and E is the set of edges or a relation that associated between two vertices. When we refer to the network, a vertex is a node and an edge is a link between two nodes. A link between two nodes i and j is represented as (i, j) and each link $(i, j) \in E$. The number of links associated with a node x in a graph of network G is called degree of a node x, denoted by $d(x)$. If each link of a graph is associated with some specific values (weights), such graph is said to be weighted.

A subgraph of G originating and covering up to n-hop from node i is denoted by $G_i^{n-hop} = (V_i^{n-hop}, E_i^{n-hop})$, where V_i^{n-hop} is the set of all nodes contained within n hops of node i, and E_i^{n-hop} is the set of links associating between two nodes in G_i^{n-hop}.

When a path (or link) from node x to node y exists, where $x, y \in V$, this path is denoted by p_{xy}.

B. Proposed Weighted Connectivity Index

A topological index is a numeric quantity which is mathematical derived from the structural graph of a molecule. In 1975, Randić [24] introduced the Connectivity Index (CI) so called Randić Index which has become the widely used topological index in many applications such as chemical and physical properties [25]˜[27].

The Randić Index is defined in the literature as follows:

$$\chi = \chi(G) = \sum_{(i,j)\in E} \frac{1}{\sqrt{d(i)d(j)}} \tag{1}$$

where the summation is carried out over all links of G.

In this work, we propose to use CI of node, defined as the CI of subgraph originating at each node, to illustrate the link characteristic of every node in the network. It is shown in [28] that the higher value of CI, the better link connectivity of mobile node in ad hoc network is. This implies that nodes with lower link connectivity have higher probability to cause link break in the connecting paths since they may move out of the coverage area of their neighbors. Of course, this leads to increasing dropped packets caused by disconnected links which also affects throughput. Thus, it can be anticipated that the network performances such as throughput and packet delivery ratio can be improved if link connectivity is put into account in path selection process.

In wireless networks, link capacity (available channel bandwidth) indicates transmission capacity of data. This implies somewhat that the higher the link capacity is, the stronger the link connectivity becomes. Therefore, by considering the connectivity index of node, the combined merit of degree of nodes and link capacity can be achieved. In [22], we proposed new QoS routing metric called weighted CI of node which is defined as the Randić Index of subgraph modified to accommodate the link capacity. We verified that weighted CI can be effectively used as the QoS routing metric to improve the network performances.

The weighted CI of any nodes i in graph G can be computed by partitioning the network graph G into subgraph G_i^{n-hop} covering only nodes and links within n-hop from node i. Let u and v represent nodes in a set of V_i^{n-hop} and each link (u, v) is in a set of E_i^{n-hop}. The n-hop weighted CI of node i or $\chi_w(G_i^{n-hop})$ can be defined as

$$\chi_w(G_i^{n-hop}) = \sum_{(u,v)\in E_i^{n-hop}} \frac{q_{(u,v)}}{\sqrt{d(u)d(v)}} \tag{2}$$

where $q_{(u,v)}$ $(0 \le q_{(u,v)} \le 1)$ is normalized link capacity and when $q_{(u,v)} = 0$ refers to unavailable link. The term $\frac{q_{(u,v)}}{\sqrt{d(u)d(v)}}$ is called the Connectivity Index Contribution Factor (CICF) between node u and v denoted by $\psi(u, v)$ which is defined as follows:

$$\psi(u, v) = \frac{q_{(u,v)}}{\sqrt{d(u)d(v)}} \tag{3}$$

In this work, we define the normalized available link bandwidth as $q_{(u,v)}$ which refers to the ratio of free time (idle period) to overall observed time. Free time can be measured by sensing the channel to monitor the traffic status and determine how much link bandwidth is available for transmitting and/or receiving. This bandwidth estimation method is approximately the same as the Listen*f* scheme which is used in [15], [29].

To calculate weighted CI based on more number of hop counts results in more accurate information about node s connectivity. However, to obtain the node link state information beyond the neighboring nodes (more than 1 hop count), more control overhead and more bandwidth consumption are needed. Thus, there is a tradeoff between information accuracy and bandwidth consumption.

In OLSR which is proactive routing protocol, a node can collect state information up to 2-hop neighbors by using Hello messages. Thus, it is natural for any node i to collect information up to 2-hop neighbors to calculate its own weighted CI. Therefore, weighted CI based on 2-hop information of any node i called 2-hop weighted CI or $\chi_w(G_i^{2-hop})$ can be computed without generating excessive control overhead to the network. For any acyclic graph, 2-hop weighted CI of any node can be calculated simply by using its neighbors 1-hop weighted CI as shown in [22]. However, for any connected graph, 2-hop weighted CI can be generally computed as shown in the Proposition 1:

Proposition 1: For any connected graph, 2-hop based weighted Connectivity Index of any node is equal to the summation of 1-hop based weighted Connectivity Index of its neighbors subtracted by the summation of CICF of all links between its neighbors.

This proposition can be proved simply by example. Assume that Fig. 1 depicts a subgraph G_X^{2-hop} which is a graph partitioned based on 2-hop criteria of node X. The nodes A, B and C are 1-hop neighbors, while nodes D, E, F, G, H, I, J, K and L are 2-hop neighbors of node X, respectively. Links drawn in dash lines are not included in a subgraph G_X^{2-hop} since they are considered as the 3-hop links from node X. Each 1-hop and 2-hop links are associated with CICF which are denoted by lower case letters (a $\sim$ v). Thus, 2-hop weighted CI based on fresh view of node X or $\chi_w(G_X^{2-hop})$ can be computed as

$$\chi_w(G_X^{2-hop}) = \sum_{(u,v)\in E_X^{2-hop}} \psi(u,v)$$

where E_X^{2-hop} is a set of all links satisfying 2-hop criteria of node X, then

$$\begin{aligned}
\chi_w(G_X^{2-hop}) &= a+b+c+d+e+f+g+h+i \\
&\quad +j+k+l+m+n+p+q+r \\
&\quad +s+t+u+v \\
\chi_w(G_X^{2-hop}) &= (a+b+c+d+m+n+p+q) \\
&\quad +(e+f+g+h+n+r+s+t) \\
&\quad +(i+j+k+l+p+s+u+v) \\
&\quad -(n+p+s) \\
&= [\chi_w(G_A^{1-hop})+\chi_w(G_B^{1-hop}) \\
&\quad +\chi_w(G_C^{1-hop})]-[\psi(A,B) \\
&\quad +\psi(A,C)+\psi(B,C)]
\end{aligned}$$

Thus, we can conclude that

$$\begin{aligned}
\chi_w(G_X^{2-hop}) &= \sum_{j\in V_X^{1-hop}; j\neq X} \chi_w(G_j^{1-hop}) \\
&\quad - \sum_{(u,v)\in E_X^{1-hop}; u,v\neq X} \psi(u,v) \quad (4)
\end{aligned}$$

where V_X^{1-hop} is a set of all nodes that are contained within 1 hop of node X and E_X^{1-hop} is a set of all links that are contained within 1 hop of node X. In any graph without cycles where there is no link between its neighbors, 2-hop weighted CI computation of node X as expressed in Eq. (4) is reduced to

$$\chi_w(G_X^{2-hop}) = \sum_{j\in V_X^{1-hop}, j\neq X} \chi_w(G_j^{1-hop}) \qquad (5)$$

which concides with the result shown in [22].

In any connected graph, this proposition provides a simple implementation (in terms of control overhead) for any node $i \in V$ to compute 2-hop weighted CI ($\chi_w(G_i^{2-hop})$), since there is no additional control overhead generated to the networks. In OLSR, every nodes can sense up to 2-hop neighbors, hence, they know the degree of their 1-hop neighbors. Consequently, they are able to compute their own 1-hop weighted CI ($\chi_w(G_i^{1-hop}), i \in V$) which will be piggybacked onto Hello messages and œooded to their neighbors. By collecting these information, each node i knows degree of its neighbors and is able to compute $\sum_{(u,v)\in E_i^{1-hop}} \psi(u,v); u,v \neq i$. Each node i also knows

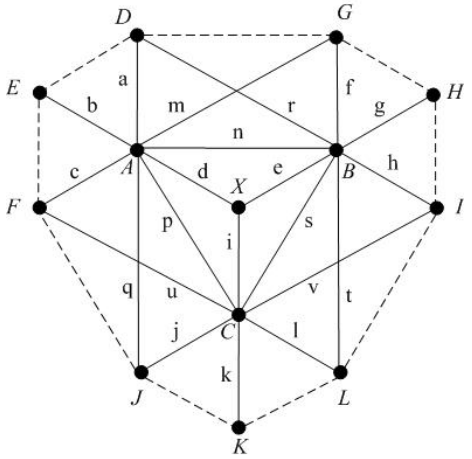


Figure 1. A weighted subgraph G_X^{2-hop} partitioned based on 2-hop information of node X (not include the links drawn in dash lines)

$\chi_w(G_j^{1-hop}); j \in V_i^{1-hop}, j \neq i$ and is able to compute $\chi_w(G_i^{2-hop})$ by using $\sum_{j \in V_i^{1-hop}} \chi_w(G_j^{1-hop}); j \neq i$ subtracted by $\sum_{(u,v) \in E_i^{1-hop}} \psi(u,v); u, v \neq i$ as shown in Proposition 1.

The weighted CI can be classified as one of non-additive QoS parameters since it is the combination of link capacity and link connectivity. Hence, cost of the path can be determined by the value of weighted CI at the bottleneck link or node. Thus, the state information of 2-hop weighted CI of a path p from source node s to destination node d illustrated as $R_w(p_{sd})$ can be defined as

$$R_w(p_{sd}) = min\left\{\chi_w(G_s^{2-hop}), \ldots, \chi_w(G_d^{2-hop})\right\} \quad (6)$$

C. Non-linear Cost Function

In this work, the non-linear cost function proposed in [30] is used to non-linearly combine multiple additive QoS metrics e.g., latency, loss rate and etc. For the sake of understanding, we review briefly here.

Each link $(i,j) \in E$ in graph G is associated with a primary cost parameter $c(i,j)$; K additive QoS parameters $w_k(i,j)$, $k = 1, 2, \ldots, K$; K constraints c_k, $k = 1, 2, \ldots, K$; all parameters are non-negative. The Multi-Constrained Path (MCP) problem is to find path p from source node s to destination node d that satisfies the following requirement:

- $w_k(p_{sd}) = \sum_{(i,j) \in p_{sd}} w_k(i,j) \leq c_k$ for all k

In MCP problem, primary cost of path ($\sum_{(i,j) \in p_{sd}} c(i,j)$) is not necessary to be minimized as in Multi-Constrained Optimal Path (MCOP) [31]. Hence, the non-linear cost function [30], [31] proposed to solve both MCP and MCOP problems is defined as

$$g_\lambda(p_{sd}) = \sum_{k=1}^{K} \left(\frac{w_k(p_{sd})}{c_k}\right)^\lambda, \lambda \geq 1 \quad (7)$$

Suppose that an algorithm finds a path p whose cost function expressed in Eq. (7) is minimized for a given $\lambda \geq 1$, then, the important theorem regarding to the bound on the performance of this heuristic is established as follows [31]:

Theorem 1: Consider the MCP problem. Assume that there is at least one feasible path $p*$ in the network. Let p be a path that minimizes the cost function g_λ for a given $\lambda \geq 1$. Then,

- $w_k(p_{sd}) \leq c_k$ for at least one k, and
- $w_k(p_{sd}) \leq \sqrt[\lambda]{K} c_k$ for all other k

Corollary 1: As λ increases, the likelihood of finding a feasible path also increases.

Proof for Theorem 1 and Corollary 1 can be found in [31]. In Eq. (7), when $\lambda \to \infty$, the largest term of $(\frac{w_k(p_{sd})}{c_k})$ will dominate the other terms. In this case, Eq. (7) can be replaced by

$$g_\infty(p_{sd}) = max\left\{\frac{w_1(p_{sd})}{c_1}, \frac{w_2(p_{sd})}{c_2}, \ldots, \frac{w_K(p_{sd})}{c_K}\right\} \quad (8)$$

IV. Proposed Generalized MCP QoS Routing

As mentioned previously, considering only one or two QoS metrics in path selection process is not enough to support all types of applications, since each application has different QoS requirements (see Table I). However, by including various QoS metrics in path selection algorithm, the feasible path may not exist with all parameters at their optimal values. The problem of finding a feasible path is said to be NP-complete when two or more parameters are used in path computation process [32], [33]. Thus, the routing problem is solvable in polynomial time if we consider only one parameter or define the precedence among multiple QoS metrics (only one metric is accounted at a time) in path computation process.

The proposed Generalized MCP (G_MCP) QoS routing finds the path from source to destination firstly based on 2-hop weighted CI. If there is a tie (finds two or more paths with the same value of weighted CI), the path with least non-linear cost will be chosen. This G_MCP is also called *Least Cost-Highest CI Pathf* algorithm. In addition, this algorithm is solvable in polynomial time by adopting the fall-back approach of generally combined multiple metrics which are weighted CI and non-linear cost.

Even though G_MCP considers various QoS routing parameters in path selection process, the computational complexity of this algorithm is comparable to Shortest-Widest Path algorithms [18]˜[20]. This is because the ranking of multiple QoS metrics is defined. In this work, the high precedence is given to weighted CI. Since if the requirement on channel bandwidth or link capacity cannot be met, or link connectivity is lost due to absent nodes in the path, it will affect the other additive QoS parameters which are combined into non-linear cost.

Fig. 2 illustrates the pseudo-code of routing procedure in G_MCP. For a graph $G = (V, E)$, where V is the

G_MCP($G = (V, E)$; $N \subset V$)
Step 1) Initially, $N := \{s\}$
  For all $i \neq s$
    $\zeta_i := R_w(p_{si})$ and $\gamma_i := g_\lambda(p_{si})$
Step 2) $M := \{\}$
  Find $m \notin N$ so that $\zeta_m = max_{i \notin N}\ \zeta_i$
    $M := M \cup \{m\}$
Step 3) **If** there is more than one element in M **then**
    Find $m \in M$ so that $\gamma_m = min_{i \notin N}\ \gamma_i$
  $N := N \cup \{m\}$
  If N contains all nodes in V **then**
    Algorithm is completed.
Step 4) **For** all $i \notin N$,
    Tmp $:= \zeta_i$
    $\zeta_i := max\{\zeta_i, min(\zeta_m, R_w(p_{mi}))\}$
    If $\zeta_i \neq$ Tmp **then**
      $\gamma_i := \sum_{k=1}^{K} \left(\frac{w_k(p_{sm}+p_{mi})}{c_k}\right)^\lambda, \lambda \geq 1$
Step 5) Go to step 2)

Figure 2. The Generalized MCP (G_MCP) algorithm

set of nodes and E is the set of links, M is temporary set and N is the subset of set V. Assume node s is the source or computing node. Let $\zeta_i = R_w(p_{si})$ and $\gamma_i = g_\lambda(p_{si})$ denote the weighted Connectivity Index and non-linear cost of the paths from node s to any node i, respectively ($\zeta_i = 0$ and $\gamma_i = \infty$, if no link exists from node s to node i). By convention, $\zeta_s = \infty$ and $\gamma_s = 0$. In step 1) each node calculates the state information: 2-hop weighted CI and non-linear cost of the known paths from node s to node i. In Step 2), algorithm finds node(s) which provides the highest value of 2-hop weighted CI to tentatively selected node i. If there are more than one node with identical maximum value of 2-hop weighted CI, step 3) will select the node m which provides the least non-linear cost link to node i. Step 4) updates the new state information of the tentatively selected nodes around the newly selected node m. The algorithm iterates to step 2) until all nodes in set V are added as the destination nodes.

In this work, we implement G_MCP QoS routing on OLSR. However, only implementing QoS routing on OLSR may not be effective enough since some better feasible paths in terms of least cost-highest CI paths may be hidden by using the native MPR [7] due to the fact that the native MPR aims at optimizing control overheads by minimizing number of MPR nodes. Since only MPR nodes are allowed to forward topology information, only partial graph of network are known to each node. Thus, we also implement our proposed MPR computation process [34] to optimize the feasible paths found in each node by allowing it to select its MPR nodes based on weighted CI and non-linear cost as illustrated by the pseudo-code in Fig. 3.

Heuristic MPR($G = (V, E)$; $N1, N2, MPR \subset V$)
Step 1) Initially, $MPR := \{\}$
Step 2) **For** all nodes $x \in N1$, **Compute** $d(x)$
Step 3) **Find** $n \in N1$ which provide the only path to reach some nodes in $N2$
$MPR := MPR \cup \{n\}$
Step 4) **While** there exist nodes in $N2$ which are not covered by at least one node in the MPR
Step 4a) **For** all $x \in N1, x \notin MPR$
Compute numbers of nodes in $N2$ which are connected to node x and not yet covered by at least one node in MPR
Step 4b) $T := \{\}$
Find $n \in N1$ that provides the hightest weighted CI and least non-linear cost link
$T := T \cup \{n\}$
Step 4c) **If** there is more than one element in T **then**
Find $n \in T$ that has the maximum number of nodes in $N2$ which are not yet covered
$MPR := MPR \cup \{n\}$

Figure 3. Heuristic for MPR selection process [34]

In Fig. 3, let $N1$, $N2$, MPR, T, $d(x)$ denote the sets of 1-hop neighbors, 2-hop neighbors, multipoint relay (MPR) nodes, temporary nodes and degree of node x in a graph of network G, respectively. Nodes in $N1$ and $N2$ are already assigned and updated whenever a MPR selector (MPR computing node) receives Hello messages. In steps 1) and 2), each MPR selector resets its own MPR set to an empty set and computes the degree of its 1-hop neighbors. It will select the node(s) from 1-hop neighbors set to be MPR nodes if it is the node(s) which provides the only path to reach some 2-hop neighbors as shown in step 3). However, in step 4), if there are some 2-hop neighbors which are not yet discovered, it will find node(s) which provides the highest weighted CI link(s) from itself to its 1-hop neighbors. And if there is a tie (two or more nodes with the same value of weighted CI), node(s) providing the least non-linear cost link(s) to its 1-hop neighbors will be assigned to a set T. However, if there is another tie, a node in temporary set T providing the maximum number of 2-hop neighbors which are not yet discovered will be selected as MPR node as shown in step 4c).

V. Performance Evaluation and Simulation Results

In this section, we investigate the proposed G_MCP algorithm by constructing simulation using NS-2 simulator [35]. We measure the efficiency and effectiveness of the proposed G_MCP algorithm in wireless mobile ad hoc networks and compare it with traditional *OLSR* [3] and *Shortest-Widest Path* algorithm with the optimal path selection [20]. To indicate the reliability of the results obtained in the proposed G_MCP algorithm, the performances measurement with 95 % confidence interval are computed.

In this simulation, λ in non-linear cost function in our proposed G_MCP, expressed in Eq. (7) is set to its largest possible value ($\lambda = \infty$) to achieve the highest probability to find the feasible paths [23], [31]. G_MCP can support any number of additive QoS constraints by combining them into a non-linear cost. However, only delay and packet loss rate are considered as additive QoS constraints in all simulations here. Since both parameters can indicate timeliness and precision which used to measure the output performances of a routing process.

According to the subjective tests, the International Telecommunication Union (ITU) G.114 specifications recommend that more than 400 ms one-way end-to-end delay for real-time traffic is unacceptable [36]. Thus, the delay constraint used in these simulation scenarios is set to 400 ms. For the packet loss rate constraint, we set to 10 % as used in [23], since ad hoc networks are highly mobility networks with limited resources.

The performance evaluation metrics are described below:

- Throughput: the amount of data that are delivered in the network over the time
- Packet delivery ratio (PDR): the ratio of the total number of packets received by destinations to the total number of packets sent by sources
- Average end-to-end delay: the average amount of time it takes all packets to reach the destinations
- Success Ratio (SR): the ratio of the total number of connection requests whose feasible paths are found by routing algorithm to the total number of connections requested by sources

To demonstrate that the proposed G_MCP algorithm works well and is open enough to support various applications, the simulations are constructed in two main scenarios: CBR and MPEG-4 (which is encoded in VBR) services.

A. Constant Bit Rate (CBR) Services

In this scenario, we measure the performances of G_MCP using CBR traffic. The parameters used in this simulation scenario are listed in Table III. Two conditions are setup: load-varying and speed-varying, in order to demonstrate how well the G_MCP algorithm can operate and handle the CBR services when the offered load and movement speed are changed.

TABLE III.
PARAMETERS USED IN CBR AND MPEG-4 SERVICES SCENARIO

Parameters	CBR		MPEG-4	
	Load-Varying	Speed-Varying	Load-Varying	Speed-Varying
Area	1000 m by 1000 m			
MAC Protocol	IEEE 802.11		IEEE 802.11b	
Channel Capacity	2 Mbps		11 Mbps	
No. of Nodes	50 nodes			
Speed	2 m/s	1 to 30 m/s	2 m/s	1 to 30 m/s
Pause Time	0 s			
No. of S-D Pairs	10 connections		5 connections	
Offered Load	25 to 150 kbps per flow	100 kbps per flow	Rate Factor* 1 to 6	Rate Factor* 4
Delay Constraint	400 ms			
Loss Rate Constraint	10 %			
Simulation Time	300 seconds			
No. of Simulations per Scenario	10 times			

Note: *Rate Factor is a parameter used in MPEG-4 video traffic generator to scale up or scale down video input.

1) CBR Services in Load-Varying Condition: G_MCP is designed to let each node in network to be able to balance the load, since it considers the link capacity in path computation process. However, if it finds two or more path with the same weighted CI, non-linear cost will be considered. Thus, it is expected to perform well in either light or heavy load situation. Fig. 4 shows the simulation results of all performance metrics versus offered load.

Fig. 4(a) illustrates that throughputs of all routing algorithms are approximately the same in light load situation (around 25 kbps). However, when the offered load increases, OLSR has the worst performance comparing to the other QoS routing algorithms (G_MCP and Shortest-Widest Path), since it always selects the paths with shortest hop count to reach destinations without considering any QoS parameters. Whereas the Shortest-Widest Path and G_MCP consider bandwidth in their path finding processes. By comparing G_MCP with Shortest-Widest Path, even though both algorithms consider bandwidth in their path finding processes, however, G_MCP put into account Connectivity Index (in addition to bandwidth) which intuitively illustrates the probability of link break (the higher the CI is, the lower probability of link break [28]) in path finding process as well.

In Fig. 4(b), PDRs of all protocols continuously decrease when more CBR packets are generated into the network, since the congestion occurs and some packets are discarded when there is no more space for buffering the incoming packets. By comparing all protocols, it is obvious that both G_MCP and Shortest-Widest Path algorithms outperform OLSR. The reason is similar as that of Fig. 4(a), that is, both algorithms consider bandwidth in path finding process. Therefore, the bandwidth is guaranteed which results in higher packet deliver ratio comparing to OLSR. G_MCP has slightly better PDR than Shortest-Widest Path (3.11 % when offered load is at 75 kbps per connection) since, similarly, packet loss rate is also considered in G_MCP.

Fig. 4(c) highlights the improvement of G_MCP in terms of end-to-end delay over OLSR and Shortest-Widest Path. The delays of all protocols are not much different when offered load is low. However, the delay increases when offered load is high which causes the congestion and long end-to-end delay. Both G_MCP and Shortest-Widest Path algorithms can improve the end-to-end delay by considering delay as one of QoS routing metric. However, G_MCP has slightly lower delay than Shortest-Widest Path (up to 8.56 % at heavy load situation) since packet

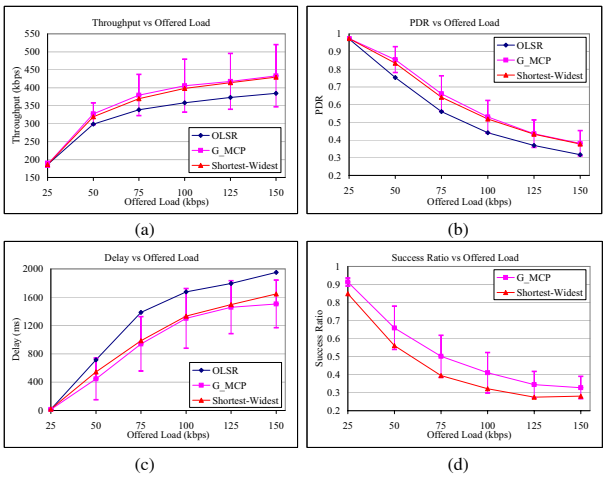


Figure 4. Effect of offered load on CBR traffic measured by (a) throughput, (b) packet delivery ratio, (c) end-to-end delay and (d) success ratio

loss rate, another additive QoS metric, is also considered in path finding process (via non-linear cost function). Thus, G_MCP has the ability to avoid the congesting path and experiences the lowest end-to-end delay.

One of the performance metrics which is especially used for measuring performance of QoS routings is the success ratio of finding the feasible path. Therefore, Fig. 4(d) depicts the success ratios of only QoS routing algorithms. It is obvious that success ratio decreases when offered load increases. Since some feasible paths may not be found by sources due to the fact that less channel bandwidth is available, more packets are discarded and time to take packets to reach destinations is longer.

Between both QoS routing algorithms, G_MCP has better success ratio (more feasible paths are found) than Shortest-Widest Path algorithm (up to 27.33 %), due to the fact that both algorithm consider the same non-additive QoS parameter (bandwidth) as primary QoS metric. However, G_MCP considers multi-constrained QoS parameters simultaneously as the secondary additive QoS metric whereas Shortest-Widest Path algorithm considers only delay in their path finding process. Therefore, the paths satisfying only delay constraint found by Shortest-Widest Path algorithm is not necessary to satisfy multi-constraints QoS in G_MCP which results in lower success ratio in Shortest-Widest Path algorithm.

2) CBR Services in Speed-Varying Condition: G_MCP is expected to improve the performances of ad hoc networks to handle services in highly dynamic topology network and is robust to link failures. Therefore, the simulation results of all performance metrics versus speed are carried out and illustrated in Fig. 5.

In Fig. 5(a) and (b), it is obvious that the throughputs and PDRs of all algorithms decrease when the speed of nodes increases. Considering the fact that OLSR is the proactive protocol which has to send the Hello and TC messages every 2 and 5 seconds (default values) in order to update the network link states. Therefore, when the speed increases, nodes may either traverse a long distance which causes the topology change or not be able to update the link state due to the link break, thus, nodes may use the out-of-date link states to forward the packets. This, of course, results in higher packet loss and decreasing throughput. In case of G_MCP and Shortest-Widest Path algorithms, they are also proactive protocols which have to send Hello and TC messages to update the link states (to calculate weighted CI in case of G_MCP), therefore, they follow the same characteristics as typical OLSR.

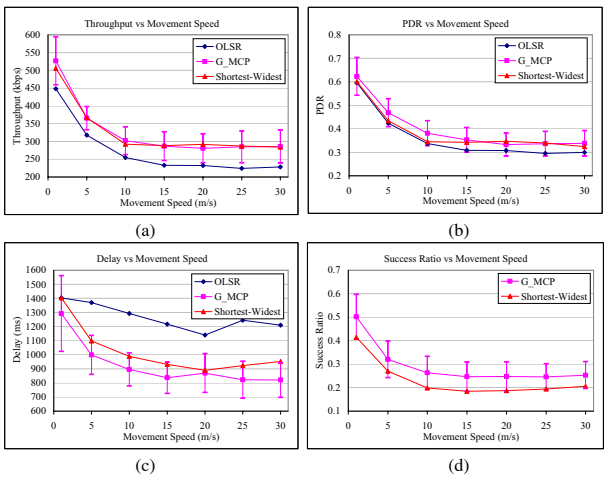


Figure 5. Effect of mobility on CBR traffic measured by (a) throughput, (b) packet delivery ratio, (c) end-to-end delay and (d) success ratio

When we compare G_MCP and Shortest-Widest Path algorithm with OLSR, it is obvious that both algorithms outperform OLSR (up to 27.94 % in terms of throughput), since both of them consider the bandwidth in path calculation. Therefore, even though the out-of-date link states are used to forward the packets, as long as the links do not break, the forwarded packets still receive a certain level of bandwidth guarantee. When the speed is not more than 15 m/s, G_MCP achieves the highest PDR (up to 10.43 % comparing to Shortest-Widest Path and 14.61 % comparing to OLSR). At high speed, throughput and PDR of G_MCP is approximately the same as Shortest-Widest Path algorithm, since in G_MCP, the calculation of weighted CI is needed. However, when speed increases, the node almost cannot exchange the Hello and TC messages, therefore the weighted CI calculation could not be done properly, so there is almost no effectiveness of weighted CI.

Fig. 5(c) illustrates the great improvement of G_MCP over OLSR (up to 33.92 %) and Shortest-Widest Path (up to 13.86 %) in terms of end-to-end delay. Since the path finding in OLSR is done without putting into account any QoS parameters whereas both G_MCP and Shortest-Widest Path always select the path satisfying delay requirement (delay requirement is considered in both algorithms). Comparing G_MCP to Shortest-Widest Path algorithm, the end-to-end delay of G_MCP is lower than that of Shortest-Widest Path in all speeds. This is because of the effect of weighted CI (the links are more stable) and non-linear cost function (where delay requirement is included). The weighted CI, by its definition as shown in Eq. (2), implies how strong the link connections of the node are. Therefore, the path with the highest weighted CI obtains the highest stability, thus, results in lower end-to-end delay.

Success ratios of both QoS routing algorithms are measured and illustrated in Fig. 5(d). It is obvious that G_MCP improves success ratio over Shortest-Widest Path (up to 33.51 %). Because G_MCP considers both non-additive and multiple additive QoS metrics using weighted CI and non-linear cost, respectively. Thus, it has higher probability to select the path satisfying multiple requirements defined by applications than Shortest-Widest Path algorithm which considers only bandwidth and delay constraints. Without considering another QoS constraint i.e. loss rate in Shortest-Widest Path algorithm, the lower success ratio will be obtained since the selected path may not satisfy this omitted QoS constraint.

B. MPEG-4 Services

In this scenario, MPEG-4 traffics are generated based on the Transform Expand Sample (TES) methodology [37], [38] which is an approach for modeling any set of given observations in a time series. We use MPEG-4 traffic generator [39] to generate the sequences of I (Intra-coded), P (Predictive coded) and B (Bidirectional coded) frames every 1/30 second and import them to NS-2 [35]. We also set two scenarios which are load-varying and speed-varying conditions to verify the performances of each routing protocol. The parameters used in this scenario are shown in Table III.

In MPEG-4 video traffic generator, there is a parameter called Rate Factor which is used to control the MPEG-4 bitstreams. Rate Factor is a parameter defined to scale up or scale down video input while preserving the same sample path and autocorrelation function for the frame size distribution. In load-varying condition, Rate Factor is varied to control MPEG-4 traffic, whereas in speed-varying condition, Rate Factor is kept constant for fair comparison.

1) MPEG-4 Services in Load-Varying Condition: Fig. 6 illustrates throughput, PDR, end-to-end delay and success ratio of all considered routing algorithms (OLSR, Shortest-Widest Path, G_MCP) when they deliver MPEG-4 bitstreams in load-varying condition.

As demonstrated in Fig. 6(a), throughputs of all protocols increase when Rate Factor increases (more MPEG-4 traffics are offered to the network). It shows that throughput of OLSR is the lowest comparing to the other algorithms, since OLSR itself doesn t provide any QoS mechanisms. Whereas G_MCP shows significant improvement over OLSR in terms of throughput (up to 24.48 %) and it has also slightly better throughput than Shortest-Widest path (up to 3.6 %). Both G_MCP and Shortest-Widest Path algorithms consider bandwidth in their path finding processes. However, G_MCP also considers link connectivity in form of weighted CI before selecting a path. Moreover, this weighted CI also indicates the reliability of a selected path since it indicates connectivity level of each intermediate node along this path. This results in lower loss rate which directly affects the throughput.

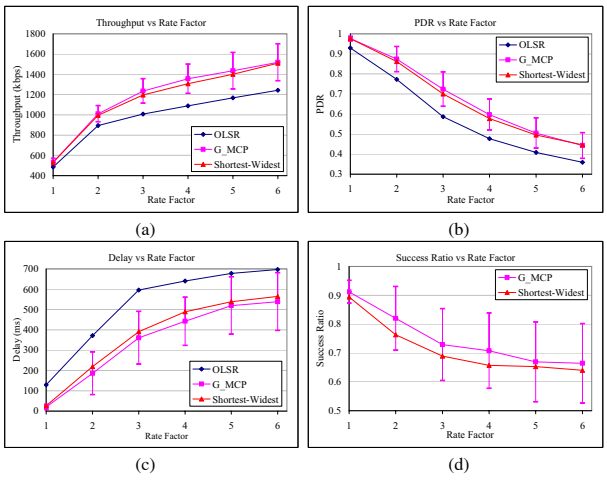


Figure 6. Effect of offered load on MPEG-4 traffic measured by (a) throughput, (b) packet delivery ratio, (c) end-to-end delay and (d) success ratio

Fig. 6(b) shows that all protocols can reliably transmit MPEG-4 frames in light load situation (Rate Factor = 1). However, when offered loads increase, G_MCP can transmit MPEG-4 frames with the highest PDR (up to 25.16 % and 3.47 % comparing to OLSR and Shortest-Widest Path algorithm, respectively) due to the same reason explained in Fig. 6(a). That is, path returned by G_MCP is a path with the highest reliability. In addition, loss rate, one of additive QoS metrics included in non-linear cost, is also considered as the second QoS metric in G_MCP while this metric is omitted in Shortest-Widest Path algorithm.

As mentioned previously in ITU G.114 specification, it is recommended that one-way end-to-end delay for real-time traffic should be less than 400 ms [36]. In Fig. 6(c) which depicts delays of all algorithms, end-to-end delay of OLSR is acceptable when Rate Factor is not more than 2, while delays of both G_MCP and Shortest-Widest Path algorithms are acceptable when they transmit MPEG-4 bitstreams with Rate Factor that is not more than 3, since they consider delay constraint when selecting a path. Between G_MCP and Shortest-Widest Path, G_MCP has the lower end-to-end delay because it has ability to avoid the congested path by considering loss rate due to congestion.

It is obvious that G_MCP outperforms Shortest-Widest Path algorithm in either light or heavy load situation when measuring success ratio (up to 7.76 %) as illustrated in Fig. 6(d). Because multiple constraints i.e. weighted CI (bandwidth and link connectivity), non-linear cost (delay and loss rate) are considered as mentioned previously in Section IV, whereas only bandwidth and delay are considered in Shortest-Widest Path algorithm. Therefore, the ratio that all predefined QoS requirements are satisfied when using G_MCP algorithm is higher.

2) MPEG-4 Services in Speed-Varying Condition: Performances of all protocols when they deliver MPEG-4 bitstreams in speed-varying condition are shown in Fig. 7.

In Fig. 7(a), throughputs decrease when node speed increases from 1 m/s to 10 m/s and remains roughly constant when speed is beyond 10 m/s. In this figure, both G_MCP and Shortest-Widest Path algorithms achieve lots of improvement over OLSR (up to 26.16 %) since both of them consider bandwidth in their path finding process. PDRs of all protocols follow the same trend as throughput, as illustrated in Fig. 7(b). Both G_MCP and Shortest-Widest Path algorithms have better performances in terms of PDR than OLSR (up to 27.10 %). Between them, G_MCP can transmit MPEG-4 bitstreams with slightly better PDR than Shortest-Widest Path (up to 3.01 %) when speed is less than 30 m/s, since link connectivity and packet loss rate are also considered in G_MCP when selecting a path. Thus, a path returned by G_MCP is

a little bit more stable than another one which results in better throughput and PDR. However, at high speed, performances of G_MCP are not much improved because CI calculation in each node is not accurate due to highly dynamic topology change, as explained in Section V-A.2.

End-to-end delay of G_MCP is the lowest (highest performance) among all routing protocols, as shown in Fig. 7(c). Its delay is always less than 400 ms, which is an acceptable level, as recommended in the ITU G.114 specification regardless of the node speed. Because a path returned by G_MCP is the best feasible path in terms of highest weighted CI and least non-linear cost. Thus, this path is the highest stability path as mentioned in Section V-A.2 which results in lowest end-to-end delay. Whereas only bandwidth and delay are considered in Shortest-Widest Path, loss rate which is another important QoS metric is left behind. Moreover, paths returned by this algorithm may not be stable. Thus, delay of Shortest-Widest Path oscillates a little bit around 400 ms and is longer than that of G_MCP. OLSR itself doesn t provide QoS routing which results in the lowest performance in terms of end-to-end delay.

From Fig. 7(d), it is obvious that G_MCP has better success ratio than Shortest-Widest Path (up to 15.45 % at speed = 10 m/s) regardless of node speed. It means that G_MCP has higher probability to find the feasible paths when it delivers MPEG-4 bitstreams. It can achieve higher success ratio because multiple constraints are put into account in G_MCP when selecting the feasible paths. Thus, the returned path in G_MCP has higher probability to satisfy bandwidth, delay and loss rate constraints defined by application in our simulation setting. Whereas Shortest-Widest Path does not consider an important QoS metric which is loss rate, therefore, the returned path found by Shortest-Widest Path algorithm is not necessary to satisfy loss rate constraint. This results in lower number of feasible returned paths.

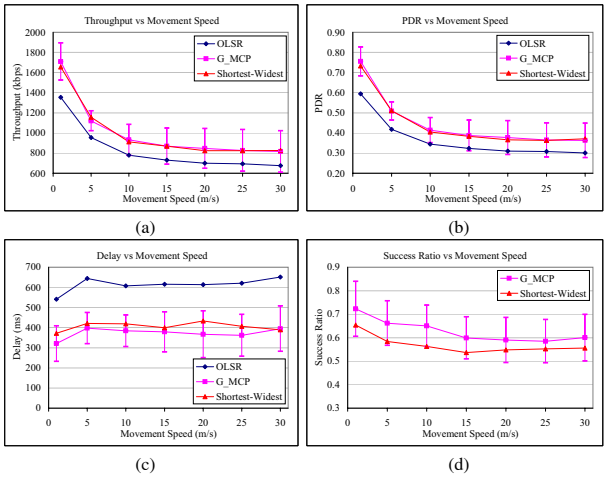


Figure 7. Effect of mobility on MPEG-4 traffic measured by (a) throughput, (b) packet delivery ratio, (c) end-to-end delay and (d) success ratio

VI. EFFICIENCY OF G_MCP

Sections V demonstrates the performance and effectiveness of G_MCP in terms of throughput, packet delivery ratio, end-to-end delay and success ratio. However, to study the efficiency of G_MCP QoS routing, the comparison of time complexity and communication complexity (namely, control overhead) of all algorithms is needed. Therefore, this simulation scenario is set up to illustrate the time and communication complexities of each routing algorithm. We use the same simulation parameters as listed in Table III in CBR traffic for load-varying condition except that the number of nodes are varied and simulation time is reduced to 60 seconds. Table IV details the platform of the machine used to perform this simulation.

The time complexities of all algorithms can be compared by simply measuring execution time (of ns-2 for each scenario) of each protocol in the same simulation environment and parameters to select paths to any reachable nodes in the network. Fig. 8(a) depicts that the execution time for running the simulation of each routing protocol increases when there are more number of nodes in the network, since the complexities of these routing algorithms are directly proportional to number of nodes (up to about 25 minutes for 100 nodes). The execution time here is the time taken by ns-2 simulator for each scenario in exactly the same environment and parameters. Therefore, the amount of time difference occurs due to only routing algorithm itself. By comparing the execution time of these routing algorithms, their complexities can be measured indirectly and compared relatively.

It is obvious that the execution times of all routing algorithms are not much different when number of nodes in the networks is not more than 50 nodes. However, when number of nodes increases. The execution times of both G_MCP and Shortest-Widest Path algorithms are approximately the same but a bit larger than OLSR, since they require more computation of multiple QoS con-

TABLE IV.
SPECIFICATIONS OF THE PLATFORM USED IN THIS SIMULATION

Processor	Intel Quad Core Q9650 processor 3.0 GHz
Memory	2.0 GB
Operating System	Fedora Core 7
Simulator	NS version 2.29

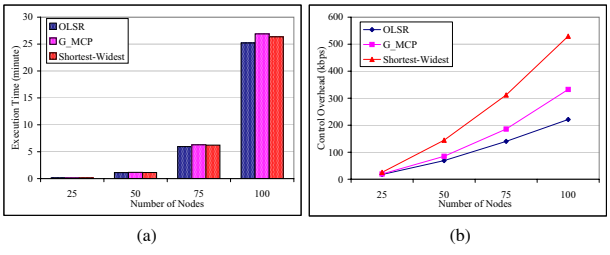


Figure 8. Efficiency of each algorithm on (a) execution time to find paths to any reachable nodes and (b) control overhead flooded to the networks

straints (weighted CI and non-linear cost in G_MCP, and bandwidth and delay in Shortest-Widest Path algorithm) by defining the precedence among them.

Fig. 8(b) shows the communication complexity which is illustrated by the total control overheads of each routing algorithm. In this figure, it is obvious that control overheads of all algorithms tend to increase when there are more nodes in the networks.

In Shortest-Widest Path algorithm, it modifies MPR computation process to select MPR nodes based on bandwidth and delay to improve the performances in terms of throughput, error rate and packet loss rate [20]. However, this algorithm does not consider topological information which results in the largest number of control overhead œooded to the network due to the increasing number of MPR nodes.

Control overhead of G_MCP is moderate among all algorithms. It is more than OLSR about 23.10 % but less than Shortest-Widest Path algorithm about 41.24 % at 50 nodes. In G_MCP, MPR computation process is also modified to select the optimal path based on weighted CI and non-linear cost. Therefore, each node selects its MPR nodes based on topological information or link connectivity to the others. This leads to more number of MPR nodes than OLSR which results in more control overheads. It should be noted that the increasing control overhead in G_MCP is due to only the increasing number of MPR nodes. There is no additional control packet sent to compute weighted CI since this is carried out by piggybacking onto Hello messages.

VII. Conclusions

In this paper, G_MCP is proposed to incorporate QoS routing into mobile ad hoc network by considering both additive and non-additive QoS parameters in path computation process. The weighted Connectivity Index (combined parameter of link connectivity and capacity) and non-linear cost (combined multiple additive QoS metrics) are proposed as non-additive and additive QoS metrics to be used in this type of network, respectively.

In simulations, we compared performances of the proposed G_MCP with some routing algorithms namely, OLSR and Shortest-Widest Path algorithms using both CBR and MPEG-4 (encoded into VBR) traffics. We can conclude that G_MCP outperforms OLSR in terms of throughput, PDR and end-to-end delay regardless of the offered load and node speed. It also gains lots of advantages over Shortest-Widest Path QoS routing algorithm in terms of success ratio of finding the feasible paths.

However, G_MCP is a little bit more complex than OLSR but it has approximately the same level of complexity as Shortest-Widest Path algorithm since they requires the additional computation in selecting the paths based on multiple QoS metrics. Control overhead of G_MCP is larger than OLSR but less than Shortest-Widest Path algorithm. Thus, Shortest-Widest Path algorithm consumes more bandwidth than the others in order to exchange the control overhead when there are more number of nodes in the networks.

In G_MCP, multiple QoS parameters are effectively combined and considered in path finding process. It can find the feasible paths satisfying multiple QoS constraints which are differently required by various applications. Since link connectivity is one of QoS parameters that is taken into account in selecting paths, therefore, ad hoc networks implementing G_MCP are more robust to link failures and are capable to operate in highly mobility network.

References

[1] C.E. Perkins and P. Bhagwat, Highly Dynamic Destination Sequenced Distance Vector Routing (DSDV) for Mobile Computers,*f* Proceedings of ACM SIGCOMM, pp. 234˘244, August 1994.

[2] R. Ogier, F. Templin and M. Lewis, Topology Dissemination Based on Reverse-Path Forwarding (TBRPF),*f* RFC 3684, February 2004.

[3] T. Clausen and P. Jacquet, Optimized Link State Routing Protocol (OLSR),*f* RFC 3626, October 2003.

[4] C. E. Perkins, E. Belding-Royer and S. Das, Ad hoc On-Demand Distance Vector (AODV) Routing,*f* RFC 3561, July 2003.

[5] D. Johnson, Y. Hu and D. Maltz, The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4,*f* RFC 4728, February 2007.

[6] C. Mbarushimana and A. Shahrabi, Comparative Study of Reactive and Proactive Routing Protocols Performance in Mobile Ad Hoc Networks,*f* Proceedings of 21st International Conference on Advanced Information Networking and Applications Workshop (AINAW), pp. 679˘684, May 2007.

[7] A. Qayyum, L. Viennot and A. Laouiti, Multipoint Relaying: An Efficient Technique for Flooding in Mobile Wireless Networks,*f* INRIA Research Report RR-3898, 2000.

[8] A. S. Tanenbaum, Computer Networks, Fourth edition, New Jersey, Prentice Hall, 2003.

[9] S. Chen and K. Nahrstedt, Distributed Quality-of-Service Routing in Ad hoc Networks,*f* IEEE Journal on Selected Areas in Communications, vol. 17, no. 8, pp. 1488˘1505, August 1999.

[10] H. Xiao, W. K.G. Seah, A. Lo and K. C. Chua, A Flexible Quality of Service Model for Mobile Ad hoc Networks,*f* Proceedings of IEEE Vehicular Technology Conference, Vol. 1, pp. 445˘449, September 2000.

[11] Y. Hwang and P. Varshney, An Adaptive QoS Routing Protocol with Dispersity for Ad Hoc Networks,*f* Proceedings of 36th Hawaii International Conference on System Sciences (HICSS), January 2003.

[12] Y. Hwang and P. Varshney, An Adaptive Routing Protocol for Ad-hoc Networks using Multiple Disjoint Path,*f* Proceedings of IEEE Vehicular Technology Conference, vol. 3, pp. 2249˘2253, May 2001.

[13] Q. Xue, and A. Ganz, Ad Hoc On-demand Routing (AQOR) in Mobile Ad Hoc Networks,*f* Journal of Parallel and Distributed Computing, vol. 63, no. 2, pp. 154˘165, 2003.

[14] S. Medidi and K. Vik, Quality of Service-aware Source-Initiated Ad Hoc Routing,*f* Proceeding of IEEE Sensor and Ad Hoc Communications and Networks (SECON), pp. 108˘117, October 2004.

[15] L. Chen and W. Heinzelman, QoS-aware Routing Based on Bandwidth Estimation for Mobile Ad Hoc Networks,*f* IEEE Journal on Selected Areas in Communications, vol. 23, no. 3, pp. 561˘572, March 2005.
[16] P. Sinha, R. Sivakumar and V. Bharghavan, CEDAR: A Core-Extraction Distributed Ad hoc Routing Algorithm,*f* Proceedings of IEEE INFOCOM, March 1999.
[17] Y. Ge, T. Kunz and L. Lamont, Quality of Service Routing in Ad Hoc Networks Using OLSR,*f* Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS 03), January 2003.
[18] H. Badis, A. Mauaretto, K. A. Agha and G. Pujolle, QoS for Ad Hoc Networking Based on Multiple Metrics: Bandwidth and Delay,*f* Proceedings of the 5th IEEE International Conference on Mobile and Wireless Communications Networks, pp. 15˘18, October 2003.
[19] H. Badis and K. A. Agha, QOLSR Multi-path Routing for Mobile Ad Hoc Networks Based on Multiple Metrics: Bandwidth and Delay,*f* Proceedings of IEEE Vehicular Technology Conference, vol. 4, pp. 2181˘2184, May 2004.
[20] H. Badis, A. Mauaretto, K. A. Agha and G. Pujolle, Optimal Path Selection in a Link State QoS Routing Protocol,*f* Proceedings of IEEE Vehicular Technology Conference, vol. 4, pp. 2570˘2574, May 2004.
[21] D. Nguyen and P. Minet, QoS Support and OLSR Routing in a Mobile Ad hoc Network,*f* Proceedings of the International Conference on Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies, April 2006.
[22] K. Kunavut and T. Sanguankotchakorn, QoS-aware Routing for Mobile Ad hoc Networks Based on Multiple Metrics: Connectivity Index (CI) and Delay,*f* Proceedings of IEEE Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON 2010), pp. 46˘50, May 2010.
[23] K. Kunavut and T. Sanguankotchakorn, Multi-Constrained Path (MCP) QoS Routing in OLSR based on Multiple Additive QoS Metrics,*f* Proceedings of IEEE Symposium on Communications and Information Technologies 2010 (ISCIT 2010), pp. 226˘231, October 2010.
[24] M. Randić, On Characterization of Molecular Branching,*f* Journal of The American Chemical Society, vol. 97, no. 23, pp. 6609˘6615, November 1975.
[25] I. Gutman, O. Araujo and D.A. Morales, Estimating the connectivity index of a saturated hydrocarbon,*f* Indian Journal of Chemistry, vol. 39, pp. 381˘385, 2000.
[26] I. Gutman, O. Miljković, G. Caporossi and P. Hansen, Alkanes with small and large Randić connectivity indices,*f* Chemical Physics Letters, vol. 306, no. 5, pp. 366˘372, June 1999.
[27] G. Caporossi, I. Gutman, P. Hansen and L. Pavlović, Graphs with maximum connectivity index,*f* Computational Biology and Chemistry, vol. 27, pp. 85˘90, 2003.
[28] M. A. Rajan, M. G. Chandra, L. C. Reddy and P. Hiremath, A Study of Connectivity Index of Graph Relevant to Ad Hoc Networks,*f* International Journal of Computer Science and Network Security, vol. 7, no. 11, pp. 198˘203, November 2007.
[29] L. Shi, A. Fapojuwo, N. Viberg, W. Hoople and N. Chan, Methods for Calculating Bandwidth, Delay, and Packet Loss Metrics in Multi-hop IEEE 802.11 Ad Hoc Networks,*f* Proceedings of IEEE Vehicular Technology Conference, pp. 103˘107, May 2008.
[30] H. D. Neve and P. V. Mieghem, A Multiple Quality of Service Routing Algorithm for PNNI,*f* Proceedings of IEEE ATM Workshop, pp. 324˘328, May 1998.
[31] T. Korkmaz and M. Krunz, Multi-constrained Optimal Path Selection,*f* Proceedings of the IEEE INFOCOM, vol. 2, pp. 834˘843, April 2001.
[32] J. M. Jaffe, Algorithms for finding paths with multiple constraints,*f* Networks, vol. 14, pp. 95˘116, 1984.
[33] Z. Wang and J. Crowcroft, Quality of Service Routing for Supporting Multimedia Applications,*f* IEEE Journal on Selected Areas in Communications, vol. 14, no. 7, pp. 1228˘1234, September 1996.
[34] K. Kunavut and T. Sanguankotchakorn, Optimized Path Selection Process in OLSR Based on Weighted Connectivity Index and Delay,*f* Proceedings of IEEE Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON 2011), pp. 348˘351, May 2011.
[35] Network Simulator 2, http://www.isi.edu/nsnam/ns
[36] Chen, T. Farley and N. Ye, QoS Requirements of Network Applications on the Internet,*f* International Journal on Information-Knowledge-Systems Management, vol. 4, pp. 55-76, January 2004.
[37] B. Melamed, TES: A Class of Methods for Generating Autocorrelated Uniform Variates,*f* ORSA Journal on Computing, vol.3, pp. 317˘329, 1991.
[38] B. Melamed, D. Raychaudhuri, B. Sengupta and J. Zdepski, TES-Based Modeling for Performance Evaluation of Integrated Networks,*f* Proceedings of IEEE INFOCOM, 1992.
[39] Video Traffic Generator, http://www.sce.carleton.ca/∼amatrawy/mpeg4

Kunagorn Kunavut was born in 1979. He received the B.S. and M.S. in Telecommunications Science from Assumption University, Thailand, in 2000 and 2004, respectively. He is currently pursuing the doctoral degree in Telecommunications Field of Study at Asian Institute of Technology, Thailand under Royal Thai Government (RTG) Fellowship. He has been served as a faculty member at Assumption University since 2001. His current research interests include QoS, medium access control, wireless multimedia and wireless networks protocol.

Teerapat Sanguankotchakorn was born in Bangkok, Thailand on 08th December 1965. He received the B.Eng in Electrical Engineering from Chulalongkorn University, Thailand in 1987, M.Eng and D.Eng in Information Processing from Tokyo Institute of Technology, Japan in 1990 and 1993, respectively under Japanese Government Scholarship. In 1993, he joined Telecommunication and Information Systems Research Laboratory at Sony Corporation, Japan, where he holds two patents on Signal Compression. He joined Asian Institute of Technology in October 1998 as an Assistant Professor. Currently, he is an Associate Professor at Telecommunications Field of Study, School of Engineering and Technology, Asian Institute of Technology. His research interests include Digital Signal Processing, High Speed Network Protocol, IP Network and Broadband Access Network, QoS in IP and Mobile Ad Hoc Network, and Cross-layer Design in Wireless Network. He is the senior member of IEEE and member of IEICE, Japan.

Game Theoretic Modeling of NGANs: Impact of retail and wholesale services price variation

João Paulo R. Pereira
Polytechnic Institute of Bragança (IPB), Portugal
jprp@ipb.pt

Pedro Ferreira
Technical University of Lisbon (IST), Portugal
pedrof@cmu.edu

***Abstract* - The increasing demand for broadband access leads operators to upgrade the existing access infrastructures (or building new access network). Broadband access networks require higher investments (especially passive infrastructures such as trenches/ducts and base station towers/masts), and before making any decision it is important to analyze all solutions. The selection of the best solution requires understanding the technical possibilities and limitations of the different access technologies, as well as understanding the costs of building and operating the networks. This study analyzes the effect of asymmetric retail and wholesale prices on operators' NPV, profit, consumer surplus, welfare, retail market, wholesale market, and so on. For that, we propose a tehno-economic model complemented by a theoretic-game model. This tool identifies all the essential costs of building (and operating) access networks, and performs a detailed analysis and comparison of the different solutions in various scenarios. Communities, operators/service providers, and regulators can use this tool to compare different technological solutions, forecast deployment costs, compare different scenarios, and so on, and help them in making deployment (or regulatory) decisions. The game-theory analyses give a better understanding of the competition and its effect on the business case scenarios' economic results.**

***Index Terms* - Next generation networks (NGN), Cost model, Game-theory model, Segmented regulation**

I. Introduction

Service providers, network operators, and Internet access providers are faced with the challenge of providing higher capacity access to the end user and offering wider services [1]. Consequently, new Internet infrastructure and technologies that are capable of providing high-speed and high-quality services are needed to accommodate multimedia applications with diverse quality of service (QoS) requirements. Until a few years ago, Internet access for residential users was almost exclusively provided via public switched telephone networks (PSTN) over the twisted copper pair [2]. The new quadruple play services (i.e., voice, video, data, and mobility), which require high-speed broadband access, created new challenges for the modern broadband wireless/wired access networks [3]. The new services led to both the development of several different last-mile solutions to make the access network capable of supporting the requirements and a stronger integration of optical and wireless access networks.

The move toward next-generation networks (NGNs) has significant implications for the technical architecture and design of access network infrastructure, as well as the value chains and business models of electronic communications service provision [4]. This migration has begun to transform the telecommunication sector from distinct single-service markets into converging markets [5]. NGNs allow consumers to choose between different access network technologies to access their service environment. In our work, the NGN architecture will be limited to the developments of network architectures in the access network (local loop), referred to as the next-generation access network (NGAN).

Although the cost of bandwidth in the active layer has reduced significantly (and continually) in recent years, the cost of civil works (such as digging and trenching) represents a major barrier for operators to deploy NGA infrastructure. Studies and deployments [6] show that civil infrastructure is the largest proportion of the costs of fixed access deployment (up to 80%). Duct is a critical part of the next-generation access networks and its sharing would reduce or eliminate this capital cost and barrier to entry. However, duct access may need to be complemented by extra civil work to increase infrastructure capacity, the use of dark fiber (where available), or the use of conduits of alternative infrastructure providers. This also highlights that different and/or complementary regulatory tools may be required in different parts of the network [7].

II. Effects of NGNs on Market Definition

The entry of new competitors can be based on the resale of services from the incumbent, on building up their own infrastructures, on renting unbundled infrastructure from incumbents, or, on the combination of the above elements. The availability of these options to competitors and price definition are generally determined by regulatory policies [8]. So, the introduction of NGNs

Manuscript received October 25, 2011; revised November 3, 2011; accepted November 25, 2011.

doi:10.4304/jcm.7.3.258-264

by telecommunication network operators obligates the national regulators adapt their access regulation regimes to the new technological conditions. Regulation and/or promotion of competition by regulatory measures need to be analyzed and compared.

The access network is usually the most expensive component in terms of capital investment (specifically passive infrastructure) and OA&M costs. Of the several costs, civil engineering costs are greatest when it is necessary to run a new fiber or copper connection to the cabinet, building, or home. Moreover, access to existing infrastructure, such as the ducts of the incumbent or other market players or sewage pipes, is critically important to avoid digging.

For [9], a local loop network can be divided into three main layers or segments: a service layer and two infrastructure layers (see Figure 1). Layer 1 includes passive infrastructures, such ducts and cables, and requires the greatest investment. Layer 2 consists of active infrastructures, such as the technical installations at the end of the fibers that send, receive, and manage the optical signals. Layer 3 includes several services that consumers buy from telecommunication operators.

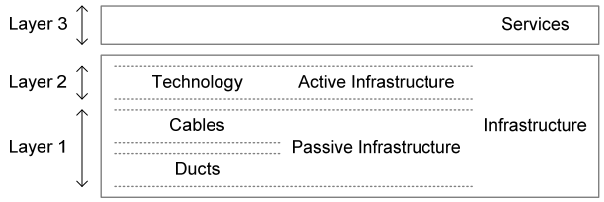


Figure 1. Network layers [9]

III. Business Case Definition

The definition of a business case implies a great number of assumptions, such as the penetration rate, components prices, and the market share rate. However, it is difficult to get an exact forecast of its performance. The utility of a business case is to offer a more approximated estimation that allows the construction of scenarios for the future. A business case should be as realistic as possible in order to be useful and reflect all the variables of interest of the market, as well as their evolution and expected behavior [10].

A. Territory and demography

The geographical areas considered are an area with high population density and an area with low population density and high coverage. For the rural area, the rollout strategy does not cover the whole area (1173 km2)—the target area is limited to 34.04 km2 with 23,000 inhabitants (see next table). In our model, we consider the last 10 years to estimate the average rate of increase: 0.62% for the urban area and value of 0.01% for rural target area. The population density in the urban area is 3,748 inhabitants per square kilometer and 675 in the rural.

Parameters presented in next table are important to calculate the cost of trenches/ducts, which are the most significant proportion of the costs of fixed access deployment.

TABLE I. Average Lengths Assumptions

Segment	Region 1 - Urban	Region 2 - Rural
Feeder	750 m	1500 m
Distribution	300 m	750 m
Drop	15 m	25 m

Several studies and models [11-13] assume that in urban areas, the duct availability rate is about 60% for feeder segments, and 40% for the distribution segment. In rural areas, the duct availability rate is 25% for feeder and 0% for the distribution network. The report from [14] assumes that a substantial proportion (80% near to the CO and 30% nearer to the premises) of existing ducts can be re-used for fiber deployment [15].

B. Service profiles assumptions

In this business case, we define two different services: slow Internet browsing service with downstream throughput of 2 Mbps, and triple play service with 20 Mbps of downstream rate. The expected tariff evolution (the factor by which the tariff is expected to increase or decrease annually) is defined for both tariffs: connection and monthly fee (see next table).

The assumptions presented are based in the data from the review of the literature. We observe that several studies and deployments [11, 16-20] use the yearly price erosion of between 5% and 15%. The service price assumptions (prices and annual variation) are presented in next table.

TABLE II. Service Profile Characteristics: Retail Prices

Service Profiles	One time Activation Fees (Connection)	Expected tariff evolution [%]	Monthly Subscription Fees	Expected tariff evolution [%]
Serv. 1	100 €	-10%	20 € /month	-5%
Serv. 2	100 €	-10%	50 € /month	-8%

C. Broadband market forecasts

Next figure shows the penetration forecast for DSL, HFC, fiber and WiMAX for urban areas. In 2020, for the urban area, the expected penetration rates for the fixed technologies are 1.5% for WiMAX, 14.25% for HFC, 22.71% for fiber, and 30.97% for DSL. In the rural area, the expected penetration rate in 2020 is 10.95% for HFC, 23.7% for DSL, 16.41% for fiber, and 7.5% for FWA. We also assume that in rural areas the FWA operator has higher market share than in urban areas.

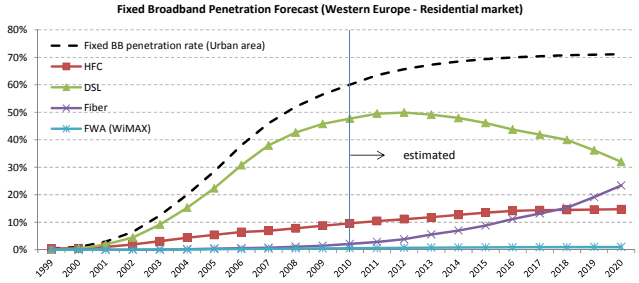


Figure 2. Fixed broadband penetration forecasts (2010-2020)

D. *Competitive situation and operators market share*

In this section, the market share (relative size) of all the firms (operators) is projected. As competition between operators is different in each area, we estimate the market share for each operator depending on the area, technology, service, and the market.

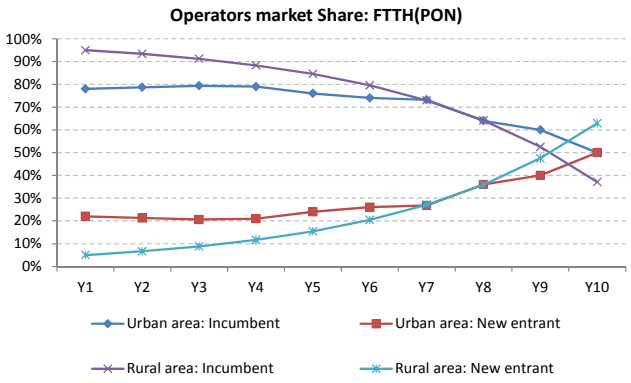


Figure 3. Market share per operator and region (FTTH market)

IV. GAME THEORY FOR COMPETING MODELING

With game theory, we want to understand the effects of the interaction between the different players defined in our business case. In the proposed games, the profit (outcome) of each operator (player) will be dependent not only on their actions, but also on the actions of the other operators in the market.

This section analyzes the impact of the price (retail and wholesale) variations on several output results: players' profit, consumer surplus, welfare, costs, service adoption, and so on. For that, two price-setting games are played (Figure 4.). Players' profits and NPV are used as the payoff for the players in the games analyzed.

From the several markets presented previously, in this section we present the results for FTTH (PON) market. We assume that two competing FTTH(PON) networks (incumbent operator and new entrant) are deployed in both areas. For the game-theoretic model, it is necessary to change the adoption model used in the techno-economic model in a way that reflects the competition between players (see next Figure 5.). We assume that the variation of the services prices of one player has an influence on the market share of all players (detailed in the next section).

In our model we also use the Nash equilibrium to find equilibrium. Proposed tools include a module to search the Nash equilibrium in the game. One strategy is a Nash equilibrium when both competitors play their best strategy related to the other strategies selected (players know each other's strategy in advance).

A. *Strategies*

To analyze the impact of retail and wholesale services price variations, we propose two games (see next figure): (1) analysis the impact of retail price variation on NPV (wholesale prices are defined by regulator); and (2) analysis the impact of retail and wholesale price variations on profit, consumer surplus, welfare, and retail/wholesale market (different wholesale prices in each region). For the game-theoretic evaluation, the model calculates the NPV and operator's profit for both operators' pricing strategies. Operators' NPVs are used as payoffs for the players in the first and second game, and operators' profits for the third game.

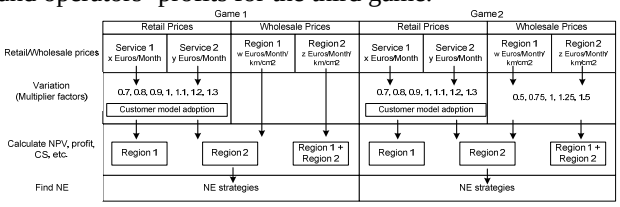


Figure 4. Games proposed

From the several assumptions, we posit: (a) the price that players charge for their services (retail and wholesale) will be varied; (b) the retail price setting will influence the market share of both players (resulting in a higher or lower market share); and (c) consumers only buy a retail service if the price is less than their willingness to pay.

As stated above, we assume that when one player increases/decreases the retail price, the market share of all players will be affected. For example, if one player offers cheaper services, it will be able to capture a higher market share. If a price decreases to nearly zero, everyone will use the service, and the market share of this operator will be close to 100% (total market). On the other hand, if an operator charges a higher price for a service, no one will subscribe to the service from this player, and its market share will decrease to 0%.

B. *Adoption model*

The impact of varying retail prices on market shares is estimated using the Boltzmann equation.

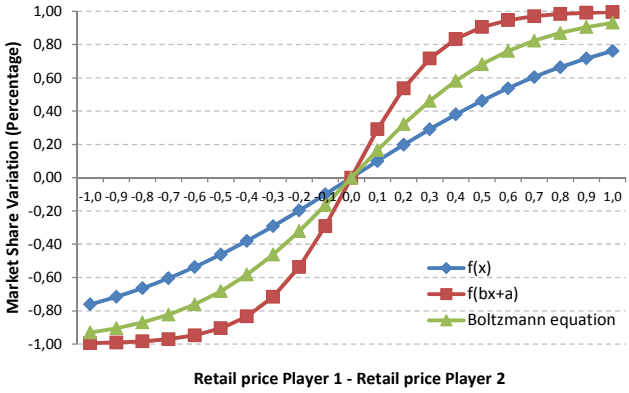


Figure 5. Models to estimate the impact of the price on the service adoption (a=0.4, b=3, dx=0.3)

C. *Main assumptions*

We assume that the willingness to pay for each retail service is different in both regions. In the urban area (region 1) the maximum amount subscribers would be willing to pay for service 1 is 26 euros and 65 euros for service 2. In the rural area we assume a willingness value of 22 euros for service 1 and 55 euros for service 2 (see TABLE III.).

TABLE III. WILLINGNESS ASSUMPTIONS

Parameters	Region 1 (Urban area)		Region 2 (Rural area)	
	Serv. 1	Serv. 2	Serv. 1	Serv. 2
Monthly Subscription Fee (Year1)	20€	50€	20€	50€
Willingness Value	26 €	65€	22€	55€
Willingness Multiplier	1.3	1.3	1.1	1.1

For the wholesale infrastructure we assume a duct availability of player 1 100% in the urban area and 90% in the rural area. We also assume that operator 2 (new entrant) leases 100% of the ducts available in the urban area and 100% of the ducts available (operator 1 has only 90% and the remaining 10% are deployed by operator 2) in the rural area from operator 1 (incumbent operator). In the other hand, player 1 leases the 10% remaining (in region 2) from operator 2. The wholesale prices assumptions are: 9.1€ (month / km / cm2) for urban area and 7.5€ (month / km / cm2) for the rural area. The wholesale infrastructure assumptions and described in next table.

TABLE IV. WHOLESALE INFRASTRUCTURE ASSUMPTIONS

Parameters	Region 1 (Urban)		Region 2 (Rural)	
	Feeder segment	Distribution Segment	Feeder segment	Distribution Segment
Provider 1				
Duct Availability	100%	100%	90%	90%
Wholesale price charged to access ducts (€/Km)	€110	€110	€90	€90
Proportion of ducts leased	0%	0%	10%	10%
From operator	-	-	2	2
Provider 2				
Duct Availability	0%	0%	10%	10%
Wholesale price charged to access ducts (€/Km)	€110	€110	€90	€90
Proportion of ducts leased	75%	75%	100%	100%
From operator	1	1	1	1

The next sections present the three games results and analyses. In the first game, retail prices vary between tariff multiplier 0.7 and 1.3 (in increments of 0.1). For the second game, retail prices vary between 0.8 and 1.2, and wholesale prices between 0.5 and 1.5.

Game 1: Impact of retail prices variation on NPV

In this game we assume that wholesale prices are fixed and that operators choose retail prices to maximize their profit. The impact of varying retail prices on market shares is estimated using the Boltzmann equation (described above). The main goal of this analysis is to determine the optimal retail price strategy for both players. The retail prices vary between –30% and 30%, with increasing steps of 10% (next table).

TABLE V. RETAIL PRICES VARIATION VALUES

Tariff multiplier factor	0.7	0.8	0.9	1	1.1	1.2	1.3
Service 1 price	14	16	18	20	22	24	26
Service 2 price	35	40	45	50	55	60	65

The combination of the two retail prices and seven multiplier factors leads to 49 possible strategies for each player (49x49 matrix) in each region (2,401 total strategies). The next table presents the structure of the combinations and calculated NPV.

TABLE VI. STRUCTURE OF COMBINATIONS AND RESULTS FOR GAME 1

Strategies	Player 1 Retail Price R1& R2		Player 2 Retail Price R1 &R 2		NPV Player 1		NPV Player 2		Tot. P1	Tot. P2
	S1	S2	S1	S2	R1	R2	R1	R2	R1 + R2	R1 + R2
1	0.7	0.7	0.7	0.7	…	…	…	…	..	…
2	0.7	0.7	0.7	0.8	…	…	…	…	…	…
n	…	…	…	…	…	…	…	…	…	…

The results (payoff matrix) of this game are presented in Table 11- shows the sum of the payoffs of each player in both regions. This table presents the NPV for both players for each possible combination of strategies (one strategy for each player); Nash equilibrium strategies are also identified.

The first two rows represents the prices multiplier factor of player 2 (for services 1 and 2) and the first two columns show the variation (multiplier factors) of player 1. Each cell contains two values: The left value corresponds to the NPV of player 1, and the value on right side corresponds to the NPV of player 2. For example, the first value calculated (15831024€) corresponds to the NPV of player 1 when the strategy of player 1 is to decrease the price of service 1 and service 2 by about 30% (multiplier factor 0.7), and the strategy of player 2 is also to decrease the price of service 1 and service 2 by about 30%.

From these results presented in Table 11 we find three pure NE strategies (black cells) that are described in the next table. The next table shows the NE strategies that maximize the profit of both players. To maximize profit, in the first equilibrium strategy, operator 1 increases retail prices by 10%. Operator 2, in face of the imposed wholesale prices, decreases the price of service 1 and service 2 by30% and 20%, respectively. A new entrant has to pay the wholesale to the incumbent, but if increase the retail prices their market share will decrease (see model above).

TABLE VII. PURE NE STRATEGIES FOR BOTH REGIONS

Strategy	Player 1 (Incumbent)		Player 2 (New entrant)		NPV K€ Player 1	NPV K€ Player 2
	Retail Serv. 1	Retail Serv. 2	Retail Serv. 1	Retail Serv. 2		
1	1.1 (22€)	1.1 (55€)	0.7 (14€)	0.8 (40€)	9.565	555
2	1.2 (24€)	1.2 (60€)	1.3 (26€)	1.1 (55€)	1.435	23.715
3	1.3 (26€)	1 (50€)	1.2 (24€)	0.7 (35€)	5.015	3.295

The next figure shows the impact of service 2 variation on NPV of both operators. From the analysis of the next figure we can conclude that the variation of retail prices of service 2 has a greater influence in the NPV than the variation of service 1 price. Service 2 price variation can

drop the NPV of operator 1 to negative. On the other hand, operator 2 can turn the NPV positive when the tariff of service 2 increases.

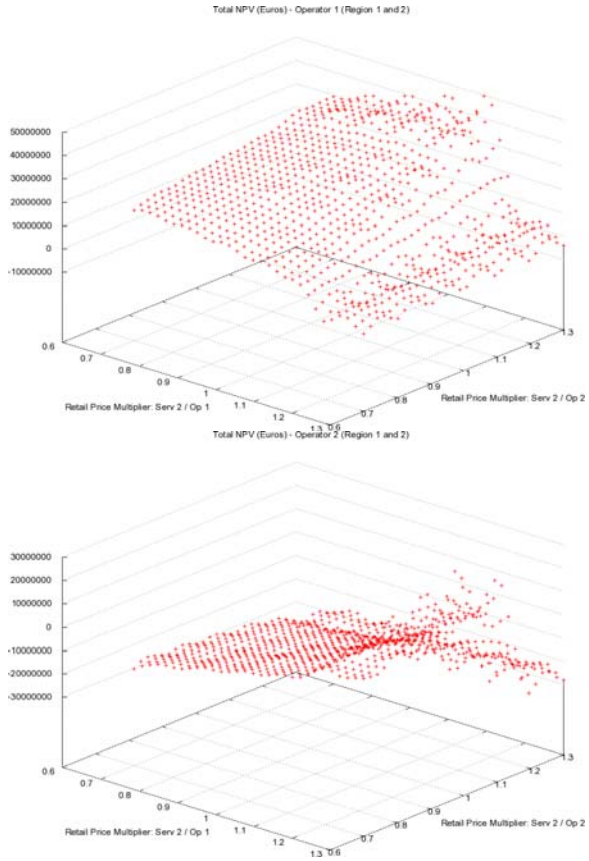


Figure 6. NPV variation: Operator1 and 2/Retail service 2

Game 2: Impact of retail and wholesale prices variation on NPV

In this game we assume that wholesale prices are not pre-imposed and we investigate what is the reaction of operators when they can also choose different wholesale prices in different regions (see next table). In game 2 we assume that has the same variation for both regions. Retail prices vary between 0.8 (-20%) and 1.2 (20%) (in increments of 0.1). For wholesale price we assume a variation between 0.5 and 1.5 (in increments of 0.25).

TABLE VIII. RETAIL AND WHOLESALE PRICES VARIATION VALUES FOR GAME 2

Service	Tariff multiplier factor				
Retail price	0.8	0.9	1	1.1	1.2
Wholesale price	0.5	0.75	1	1.25	1.5

In this context, the combination of the three prices and variation multipliers (described in the previous table) leads to 625(5^4) possible strategies for each player (625x625 matrix) in each region (390625 strategies in both regions) - TABLE IX. shows the structure used.

As the matrix is to bigger, for this game we decide to present the NE strategies (players profit is used as payoff) and the graphs that show the impact of variation in the several results (presented in TABLE XII.).

TABLE IX. STRUCTURE OF COMBINATIONS AND RESULTS FOR GAME 2

Strategies	Player 1				Pl 2	Results - NPV					
	Retail Price		Wholesale Price		…	Player 1		Player 2		Tot P1	Tot P2
	R1& R2		R1	R2	…	R1	R2	R1	R2	R1 + R2	R1 + R2
	S1	S2	Duct Access		…						
1	0.8	0.8	0.8	0.8	…						
2	0.8	0.8	0.8	0.8	…						
n	…	…	…	…	…						

The analysis of the results finds five NEs strategies. As player 2 do not operates in the wholesale market of region 1, the variation of this price is not significant (see next table). We conclude that, in the business case defined, when operators can charge different retail and wholesale prices, they choose to increase wholesale prices. To maximize profits, operators increase wholesale prices and decrease retail prices. However, the increase in wholesale prices precludes entry of new operators into the market.

TABLE X. PURE NE STRATEGIES IN BOTH REGIONS (GAME 2)

Player 1 (Incumbent operator)				Player 2 (New entrant)				Profit (K€) Player 1	Profit (K€) Player 2
Retail		Wholesale		Retail		Wholesale			
S1	S2	R1	R2	S1	S2	R1	R2		
0.8	0.8	1.25	1.25	0.8	0.8	0.50 0.75 1 1.25 1.5	1.25	22 402	101
0.8	0.9	1.25	1	0.8	0.8	0.50 0.75 1 1.25 1.5	1.25	19 543	6.198

The main results of this game are summarized in the next figures. In the graphs we can see the impact of retail prices (Figure 7.) and wholesale prices (Figure 8.) on players profit. We can verify that both prices can turn profit positive/negative.

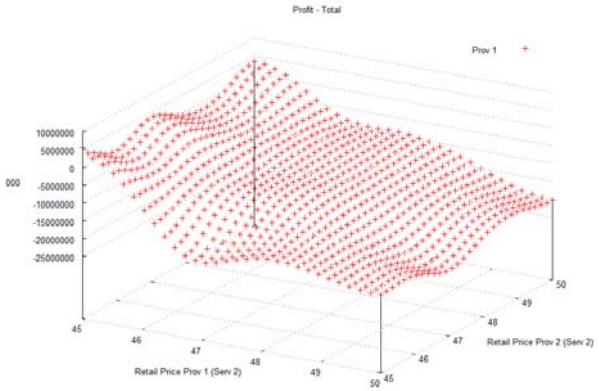


Figure 7. Profit variation: Retail service 2

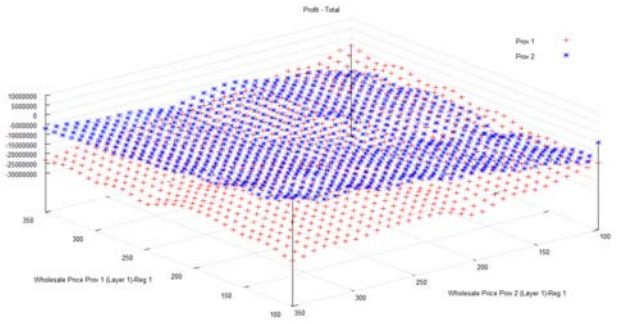


Figure 8. Profit variation: Wholesale service

As expected, consumer surplus decreases with the increase of prices (Figure 9.). As also predictable and modeled above the impact of retail prices variation has higher influence in the market share of competitors (see Figure 10.).

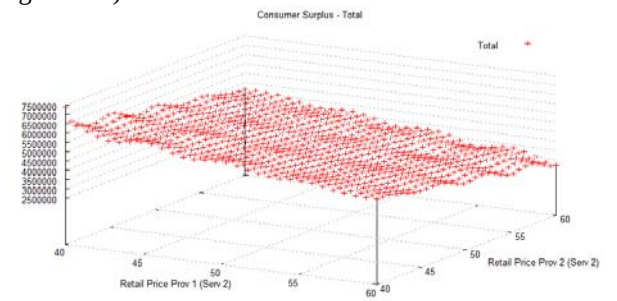


Figure 9. Consumer Surplus variation

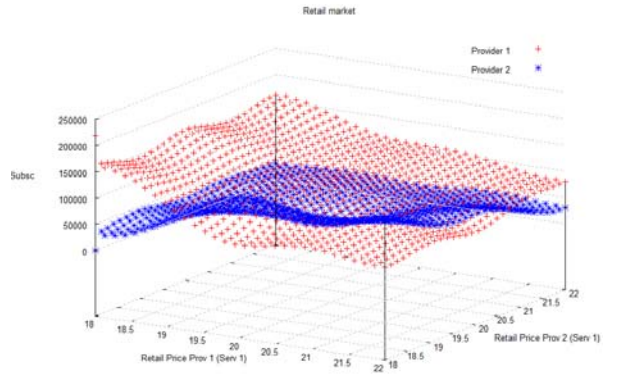


Figure 10. Retail market variation

The comparison of the two games above shows that when the regulator defines wholesale prices, operators increase retail prices to maximize profit. However, when wholesale prices are not regulated, operators maximize profit by decreasing retail prices and increasing wholesale prices. However, without regulation, the higher wholesale prices will limit the entrance of new competitors.

V. Conclusions

The European Commission argues that infrastructure-based competition is the best and fastest way for broadband development. The arguments are that infrastructure-based competition provides efficiency incentives to operators, reduces prices, increase penetration, stimulates innovation, and so on. On the other hand, service-based competition implies that the new entrants (alternative operators) are dependent on the incumbent. However, because of the high costs of deploying infrastructures (especially trenching and ducting), service competition has been used as a substitute or complement to infrastructure competition. In regions with lower numbers of existing access infrastructures, new entrants are obligated to build their own infrastructure. In this way, infrastructure sharing can stimulate the construction of new access infrastructures that can be leased to other operators.

The results of this investigation show that the sharing of passive infrastructures (e.g., ducts, trenching, base station sites, antenna masts, etc.) is a viable strategy, particularly in the context of new building (in scenarios with developed access infrastructure). When an operator deploys an access network, the access to existing civil engineering significantly reduces the investment. There are strong arguments to be made for allowing infrastructure sharing.

In this context, regulators must guarantee new entrant operators access to civil engineering; this will stimulate investment in new networks. The reduction of the barriers to new infrastructure investment by opening passive existing infrastructure would be key in the future. This study has shown that in rural areas, characterized by a small number of developed access infrastructure, the access to civil engineering does not make the scenario economically viable for the operator.

VI. References

1. S.L. Kota, “Satellite Multimedia Networks and Technical Challenges,” Microwave Review, 2006.
2. O.C. Ibe, Fixed Broadband Wireless Access Networks and Services, John Wiley & Sons, Inc., 2002.
3. J.P. Pereira and P. Ferreira, “Access networks for mobility A techno-economic model for broadband access technologies,” Proc. Testbeds and Research Infrastructures for the Development of Networks & Communities and Workshops, 2009. TridentCom 2009. 5th International Conference on, 2009, pp. 1-7.
4. J.S. Marcus, et al., Next Generation Networks (NGNs), European Parliament, 2009.
5. F. Kirsch and C.V. Hirschhausen, “Regulation of Next Generation Networks: Structural Separation, Access Regulation, or no Regulation at all?,” Proc. First International Conference on Infrastructure Systems and Services: Building Networks for a Brighter Future (INFRA), 2008, pp. 1-8.
6. J.P. Pereira, “Telecommunication Policies for Broadband Access Networks,” Proc. The 37th Research Conference on Communication, Information and Internet Policy, 2009, pp. 1-13.
7. Analysys-Mason, Telecoms infrastructure access – sample survey of duct access, Ofcom, 2009.
8. J.P. Pereira and P. Ferreira, “Next Generation Access Networks (NGANs) and the geographical segmentation of markets,” Proc. The Tenth International Conference on Networks (ICN 2011), 2011, pp. 6.
9. C. Jaag and M. Lutzenberger, “Approaches to FTTH-Regulation: An International Comparison,” Proc. Second Annual Conference on Competition and Regulation in Network Industries, 2009, pp. 23.

10. J. Rendón, et al., "A business case for the deployment of a 4G wireless heterogeneous network in Spain," Proc. 18th European Regional International Telecommunications Society, 2007.
11. T. Monath, et al., "Economics of Fixed Broadband Access Network Strategies," IEEE Communications Magazine, vol. 163, no. 8, 2003, pp. 132-139.
12. F. Loizillon, et al., Final results on seamless mobile IP service provision economics, IST- Information Society Technologies, 2002.
13. T. Monath, "Techno-economic results for fixed access network evolution scenarios," Book Techno-economic results for fixed access network evolution scenarios, Series Techno-economic results for fixed access network evolution scenarios, ed., Editor ed.^eds., 2005, pp. 28.
14. Analysys-Mason, The costs of deploying fibre-based next-generation broadband infrastructure, Analysys-Mason, Broadband Stakeholder Group, 2008.
15. CSMG, Economics of Shared Infrastructure Access, 2010.
16. K. Stordahl, "Broadband demand and the role of new technologies," Proc. Telecommunications Network Strategy and Planning Symposium, 2008. Networks 2008. The 13th International, 2008, pp. 1-23.
17. K. Stordahl, Market development up to 2015, MARCH - Multilink architecture for multiplay services, 2010.
18. K. Stordahl, et al., "Risk methodology for evaluating broadband access network architectures," Telektronikk, vol. 2, no. 3, 1999, pp. 273-285.
19. V. Riihimäki, "Managing Uncertainties in Broadband Investments- Case Studies of Real Options for Rural Area Access Networks," Department of Communications and Networking, Aalto University, Aalto, FI, 2010.
20. European-Union, Europe's Digital Competitiveness Report 2010, European Union, 2010.

TABLE XI. GAME 1 RESULTS - SUMMARY

Player 2 (New entrant) strategies

Player 1 (Incumbet operator) strategies	Price S1	0,70						,,,	1,20		,,,	1,30						,,,
	Price S2	0,70		0,80		,,,	1,30	,,,	0,70		,,,	1,30		0,70		,,,	1,10	,,,
0,70	0,7	15831024	-18582287	18183087	-19363781		28936533 -30770826		17496173	-19795217		30601681	-31983756	18113915	-20768607		26428738 -25938555	
	0,8	14472132	-14083788	19556293	-16693988		34299456 -30618352		16137281	-15296719		35964604	-31831283	16755023	-16270109		29303695 -23304216	
	0,9	12185918	-9824136	17582209	-11580297		39612688 -30427158		13851067	-11037066		41277837	-31640088	14468808	-12010456		31362952 -20234582	
	1	9338223	-5994009	14713466	-6738115		44866176 -30195243		11003372	-7206939		46531325	-31408173	11621113	-8180329		32577289 -16820707	
	1,1	6280874	-2707019	11341982	-2384199		50053039 -29925006		7946023	-3919949		51718187	-31137937	8563765	-4893339		33014991 -13215411	
	1,2	-7258790	6298197	-6933535	11368737		-4619767 -29519466		-5593641	5085267		-2954619	-30732396	-4975899	4111877		-3280895 23715662	
	1,3	-7216198	6452291	-6935595	11588571		-4402383 -29202072		-5551049	5239361		-2737234	-30415002	-4933308	4265971		-3375080 24186698	
,,,	,,,																	
1,10	0,7	14054700	-15642494	16406763	-16423988		27160208 -27831033		19064485	-17729134		32169993	-29917673	21886956	-20768607		30201779 -25938555	
	0,8	12695808	-11143995	17779969	-13754195		32523131 -27678559		17705593	-13230635		37532916	-29765199	20528064	-16270109		33076737 -23304216	
	0,9	10409594	-6884343	15805885	-8640503		37836364 -27487364		15419379	-8970983		42846148	-29574004	18241850	-12010456		35135994 -20234582	
	1	7561899	-3054215	12937141	-3798322		43089852 -27255449		12571684	-5140855		48099637	-29342089	15394155	-8180329		36350330 -16820707	
	1,1	4504550	232775	**9569657**	**555595**		48276714 -26985213		9514335	-1853865		53286499	-29071853	12336806	-4893339		36788033 -13215411	
	1,2	-9035114	9237991	-8709859	14308530		-6396091 -26579672		-4025329	7151351		-1386307	-28666312	-1202858	4111877		492146 23715662	
	1,3	-8992522	9392085	-8711919	14528364		-6178707 -26262279		-3982737	7305445		-1168922	-28348918	-1160266	4265971		397962 24186698	
1,20	0,7	13511514	-15140682	15863576	-15922176		26617022 -27329221		19138606	-17069339		32244114	-29257878	22830217	-20768607		31145040 -25938555	
	0,8	12152622	-10642183	17236783	-13252383		31979945 -27176747		17779714	-12570840		37607037	-29105404	21471325	-16270109		34019997 -23304216	
	0,9	9866407	-6382531	15262698	-8138692		37293177 -26985553		15493499	-8311188		42920269	-28914210	19185110	-12010456		36079254 -20234582	
	1	7018712	-2552404	12393955	-3296511		42546666 -26753638		12645805	-4481060		48173758	-28682294	16337416	-8180329		37293591 -16820707	
	1,1	3961364	734586	9022471	1057406		47733528 -26483401		9588456	-1194070		53360620	-28412058	13280067	-4893339		37731293 -13215411	
	1,2	-9578300	9739802	-9253046	14810342		-6939278 -26077861		-3951208	7811146		-1312186	-28006517	-259597	4111877		**1435407** **23715662**	
	1,3	-9535709	9893896	-9255106	15030176		-6721894 -25760467		-3908617	7965240		-1094801	-27689124	-217006	4265971		1341222 24186698	
1,30	0,7	11507889	-13885393	13859952	-14666887		24613397 -26073932		11507889	-9292724		24613397	-21481263	11492251	-20732168		19807074 -25902116	
	0,8	10148997	-9386894	15233158	-11997094		29976320 -25921458		10148997	-4794225		29976320	-21328789	10133359	-16233669		22682032 -23267776	
	0,9	7862782	-5127242	13259073	-6883402		35289552 -25730264		7862782	-534573		35289552	-21137595	7847145	-11974017		24741289 -20198142	
	1	5015088	-1297114	10390330	-2041221		40543041 -25498349		**5015088**	**3295555**		40543041	-20905679	4999450	-8143889		25955625 -16784268	
	1,1	1957739	1989876	7018846	2312695		45729903 -25228112		1957739	6582545		45729903	-20635443	1942101	-4856899		26393328 -13178971	
	1,2	-11581925	10995092	-11256671	16065631		-8942903 -24822571		-11581925	15587761		-8942903	-20229902	-11597563	4148317		-9902559 23752102	
	1,3	-11539334	11149186	-11258730	16285465		-8725518 -24505178		-11539334	15741855		-8725518	-19912509	-11554971	4302411		-9996743 24223138	

TABLE XII. GAME 2 RESULTS - SUMMARY

Player 2 strategies

Player 1 Strategies	R Price S1			0,80												...
	R Price S2			0,80												...
		W Price R1		0,50				1,00				1,50		0,75		...
			W Price R2	0,50		0,75				1,25				0,50	...	
0,8	0,8	0,5	0,5	20981654	1704052	20954077	1728871	20926500	1753691	20898923	1778510	20871345	1803330	20981654	1704052	
			0,75	21232678	1425137	21205100	1449956	21177523	1474776	21149946	1499595	21122369	1524415	21232678	1425137	
			1	21483701	1146222	21456124	1171041	21428547	1195861	21400969	1220680	21373392	1245500	21483701	1146222	
			1,25	21734724	867307	21707147	892127	21679570	916946	21651993	941766	21624416	966585	21734724	867307	
			1,5	21985748	588392	21958171	613212	21930593	638031	21903016	662851	21875439	687670	21985748	588392	
		0,75	0,5	21113446	1557616	21085869	1582436	21058292	1607255	21030715	1632075	21003137	1656894	21113446	1557616	
			0,75	21364470	1278701	21336892	1303521	21309315	1328340	21281738	1353160	21254161	1377979	21364470	1278701	
			1	21615493	999786	21587916	1024606	21560339	1049425	21532761	1074245	21505184	1099064	21615493	999786	
			1,25	21866516	720872	21838939	745691	21811362	770511	21783785	795330	21756207	820150	21866516	720872	
			1,5	22117540	441957	22089963	466776	22062385	491596	22034808	516415	22007231	541235	22117540	441957	
		1	0,5	21245238	1411181	21217661	1436000	21190084	1460820	21162506	1485639	21134929	1510459	21245238	1411181	
			0,75	21496261	1132266	21468684	1157085	21441107	1181905	21413530	1206724	21385953	1231544	21496261	1132266	
			1	21747285	853351	21719708	878171	21692130	902990	21664553	927809	21636976	952629	21747285	853351	
			1,25	21998308	574436	21970731	599256	21943154	624075	21915577	648895	21887999	673714	21998308	574436	
			1,5	22249332	295521	22221754	320341	22194177	345160	22166600	369980	22139023	394799	22249332	295521	
		1,25	0,5	21377030	1264745	21349453	1289565	21321876	1314384	21294298	1339204	21266721	1364023	21377030	1264745	
			0,75	21628053	985830	21600476	1010650	21572899	1035469	21545322	1060289	21517745	1085108	21628053	985830	
			1	21879077	706916	21851500	731735	21823922	756555	21796345	781374	21768768	806194	21879077	706916	
			1,25	22130100	428001	22102523	452820	22074946	477640	22047369	502459	22019791	527279	22130100	428001	
			1,5	22381124	149086	22353546	173905	22325969	198725	22298392	223544	22270815	248364	22381124	149086	
		1,5	0,5	21508822	1118310	21481245	1143129	21453668	1167949	21426090	1192768	21398513	1217588	21508822	1118310	
			0,75	21759845	839395	21732268	864214	21704691	889034	21677114	913853	21649536	938673	21759845	839395	
			1	22010869	560480	21983291	585300	21955714	610119	21928137	634939	21900560	659758	22010869	560480	
			1,25	22261892	281565	22234315	306385	22206738	331204	**22402607**	**101928**	22151583	380843	22261892	281565	
			1,5	22512915	2650	22485338	27470	22457761	52289	22430184	77109	22402607	101928	22512915	2650	
	...	...														
...	...															