Peer-to-Peer Netw Appl
DOI 10.1007/s12083-009-0048-4

# Adaptive topology formation for peer-to-peer video streaming

**Hao Luan · Kin-Wah Kwong · Xiaojun Hei · Danny H. K. Tsang**

Received: 26 November 2008 / Accepted: 6 May 2009


**Abstract** In this paper we propose an adaptive P2P video streaming framework to address the challenges due to bandwidth heterogeneity and peer churn on the Internet. This adaptive streaming framework consists of two major components, source rate adaptation and adaptive overlay topology formation, to maximize the video quality and fully utilize the overall peer upload capacity. In the source rate adaptation, the video server adapts the video source rate automatically based on the local measurement of peers' download rates, so that the P2P network is not overloaded beyond its bandwidth capacity and peers are able to achieve smooth video playback. To combat bandwidth heterogeneity, we propose to construct a desirable link-level homogeneous overlay topology using a Markov chain Monte Carlo method, so that peers achieve an equal per-connection upload/download bandwidth. In this link-level homogeneous network, video flows do not encounter any bottlenecks along the delivery paths, and peers achieve high download rates to ensure smooth video playback. We also design a fully distributed algorithm to implement the dual mechanisms of the adaptive topology formation and the source rate maximization. To evaluate the performance of our streaming framework, we conduct both mathematical analysis and extensive simulations. The simulation results confirm our analysis and show that the proposed distributed algorithm is able to maximize the video playback quality with fast convergence.

**Keywords** P2P video streaming · Topology formation · Random walk

Parts of the results in this paper have been presented in the IEEE Consumer Communications and Networking Conference (CCNC), Las Vegas, Nevada, USA, Jan. 2008.
This work is supported by RGC Earmarked Research Grant 620306.

H. Luan
Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario, Canada
e-mail: hluan@bbcr.uwaterloo.ca

K.-W. Kwong
Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA, USA
e-mail: kkw@seas.upenn.edu

X. Hei (✉)
Department of Electronics and Information Engineering, Huazhong University of Science and Technology, Wuhan, Hubei, China, 430074
e-mail: heixj@hust.edu.cn

D. H. K. Tsang
Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong, China
e-mail: eetsang@ece.ust.hk

## 1 Introduction

The emerging peer-to-peer (P2P) networks have appeared to be the most promising driving wheel for the large scale video streaming on the Internet [1–3]. Unlike IP multicast, P2P networks do not rely on any dedicated network infrastructure and hence offer the possibility of a rapid IPTV service deployment at low cost. To date, P2P video streaming systems have already achieved a number of large-scale deployments, accommodating tens of thousands of simultaneous users [4].

Published online: 28 May 2009

In P2P streaming networks, peers contribute their uplink bandwidth to assist video streaming to other peers, thus alleviating the bandwidth load on dedicated streaming servers so that the system is scalable to support millions of users simultaneously. Yet, the road to providing high-quality video streaming over P2P networks has been facing various technical challenges due to heterogeneous and dynamic nature of P2P networks.

- *Bandwidth heterogeneity*: Peers normally use different access networks and reside in diverse Internet service providers (ISPs); therefore, their upload bandwidth may vary significantly. When video traffic flows along the overlay paths between peers, the network throughput is often throttled by the bottleneck peers in the middle, and the streaming rates of the downstream nodes deteriorate drastically. Note that the upload rate of a peer is upper bounded by its download rate, a peer with a small download rate wastes its upload bandwidth capacity. To fully utilize peer's upload capacity, bandwidth heterogeneity is an important issue in designing efficient P2P streaming systems.
- *Peer churn*: Peers are inherently dynamic and may depart the network at any time; therefore, their upload capacity is not reliable for peers. Peer's download rate may be highly variable and suffer from intermittent connectivity. To avoid potential interruptions of video playback encountered by streaming clients, the streaming overlay should be repaired quickly due to peer departures.

To address the above two challenges, we propose an adaptive P2P video streaming framework, which consists of two major components: *source rate adaptation* and *adaptive topology formation*. In video streaming, it is crucial to provide users with satisfactory perceptual visual quality. Video coding plays an important role in determining visual quality of the video. With a smaller compression ratio, peers enjoy less loss of visual quality, but require a higher download rate to support continuous playback. Hence, the resulting visual quality, which peers can perceive, is determined by both video compression quality and network transport capability. Because network available bandwidth is usually unknown in advance, it is difficult (if still possible) to select an appropriate video compression rate and corresponding playback rate before video transmission. To achieve a good trade-off between video quality and network transport, we propose a source rate adaptation scheme to automatically tune the playback rate based on local measurement of peer's download rate from the source. After an appropriate playback rate is selected, the next question to answer is how to ensure peers to download above this rate for smooth media playout. To enable peers to maintain their desired data rate after multiple overlay relays, we design an adaptive topology formation algorithm to construct a homogeneous overlay in terms of link capacity, namely link-level homogeneous overlay. This overlay is constructed for two main purposes: first, it eliminates the bottleneck for video streaming over P2P networks; second, by assigning high capacity peers with more upload connections, peer's upload bandwidth is fully utilized to provide the best achievable video quality in the global network. We presented our preliminary results in [5] and this paper is a significant extension of [5] in developing a complete analytical model and much more simulation results. In summary, we make the following contributions in this paper:

- We propose a novel adaptive peer-to-peer streaming framework, which adapts the overall video playback rate based on local network measurements and maximizes the perceptual visual quality of peers. The bandwidth supply of P2P networks is usually unknown and keeps changing due to peer churn, the proposed adaptive system is able to automatically balance the bandwidth supply and demand for high-quality video streaming with full bandwidth utilization.
- We construct a link-level homogeneous (LLH) overlay network using a Markov chain Monte Carlo method, where the per-connection bandwidth converges to the same value. In such an LLH network, the amount of download rate of peers can be controlled by setting the number of download connections they have as each connection possesses an equal bandwidth capacity. Hence, we can achieve load-balanced bandwidth allocation by controlling the overlay topology. Note that this LLH approach is also effective in improving the download performance of file-sharing networks [6].
- We design a distributed algorithm to achieve the LLH topology formation for P2P streaming. We develop a fluid-flow model to analyze the performance of the proposed algorithm and show that the network converges to the link-level homogeneity. The accuracy of the model is evaluated via extensive simulations.

The rest of the paper is organized as follows. We end this section with an overview of the related work. In Section 2, we present the proposed video streaming framework in detail. In Section 3, we analyze the proposed topology formation method using a fluid-flow based analysis. We conduct a simulation

study to evaluate the topology formation algorithms in Section 4. Finally, we conclude the paper in Section 5.

### 1.1 Related work

Wide-range applications of P2P video streaming and their impact on changing the Internet traffic pattern have spurred a large body of research since 2000. Given the limited bandwidth supply of peers, it is critical to ensure that the overall bandwidth is sufficient to meet the bandwidth demand of all the peers. One common approach is to deploy the call admission control (CAC) [7, 8]. Kumar et al. analyzed the video streaming quality in P2P streaming systems with two types of peers using a fluid-flow model [7]. With the assumption of the full utilization of peer's upload bandwidth, they computed the maximal achievable video playback rate. When the overall bandwidth is not sufficient to feed all peers' playback, they propose to adopt the admission control in order to ensure that the bandwidth demand of the peers does not exceed the bandwidth supply. Huang et al. studied how to construct a profitable P2P VoD network [8]. They classify the network into the surplus mode, the balanced mode or the deficit mode, respectively, based on the criteria whether the overall bandwidth supply is larger than, equal to or less than the bandwidth demand of the peers. They argued that the optimal system performance is achieved in the balanced mode only. Unlike [7, 8], we propose to adaptively tune the video compression rate based on the local rate measurement. Hence, our streaming system does not reject any subscribers. Instead, all the users share the service degradation when the bandwidth supply is not sufficient.

Apart from ensuring the sufficiency of overall bandwidth supply for the system, it is also necessary to allocate the bandwidth and enforce quality of service (QoS) at the peer level. To address this issue, many theoretical works take the mathematical programming approach [9–11]. This usually leads to a joint topology formation and resource allocation optimization formulation. The optimal solution is chosen out of all the possible connection combinations between peers, and data transmission rates are assigned for these connections. A disadvantage of this mathematical programming approach is that it normally requires iterative adaptations and intensive communications between peers, which are expensive in large-scale, highly dynamic P2P networks. This concern has motivated the recent works [12–14] on the incremental topology formation [9]. Given peer churn, the network evolves accordingly by locally refining the overlay topology of a peer with its neighbors, whereas all the other peers remain unchanged.

On constructing an optimal mesh topology for video streaming, Magharei and Rejaie [12] showed that each peer has an acceptable range of outgoing connections proportional to its upload bandwidth. Using simulations, they demonstrated that the network performs best with the optimal bandwidth utilization and minimum buffer requirements when the per-connection bandwidth is the same. In such an overlay topology, video flows will not encounter any bottleneck connections. This desirable property is also maintained in our proposed bandwidth allocation scheme. Nevertheless, our work is accomplished independently and parallel to [12]. In addition, how to construct such a link-level homogeneous topology is not described in [12] and we propose a distributed algorithm to achieve such an overlay in this paper.

Sung et al. described an incentive scheme using for P2P video streaming [15]. In [15], an overlay multicast tree is formed with each branch of the tree having the same capacity. Such an overlay is similar to our proposal and [12]. By maintaining the in-coming branches to a peer proportional to the branches it serves, the proposed system in [15] encourages peers to download at a rate proportional to their upload rates, and provides incentives for peers to upload.

In [13], Small et al. suggested to construct the overlay mesh topology using a greedy algorithm called Outreach. In Outreach, when a peer is bootstrapped, it first connects to a deputy peer, which is a normal node in the overlay. Based on the information stored at the deputy peer, a newly-arrived peer selects multiple peers with the largest available bandwidth as its parent nodes. Constrained by the myopic operations, Outreach may select parental peers with sub-optimal performance, so a global optimal performance can be difficult to achieve.

In Chunkyspread [14], Venkatraman et al. proposed to form the balanced overlay network towards a full bandwidth utilization. Each peer has a predefined transmission load which is proportional to its upload capacity. The workload is maintained locally within a certain region. However, this transmission load of peers does not take other peers' bandwidth capacity into consideration. Hence, Chunkyspread can hardly achieve a good load balance for the global network.

In [16], Kwong and Tsang proposed a basic random walk framework for the P2P topology formation and provided mathematical analysis for the heterogeneous P2P networks. Then, they advanced to a more general framework [17] by introducing two control parameters in the peer sampling process. However, in [16, 17], the

authors did not consider any P2P applications on top of the constructed overlay topology and the research efforts purely focused on the topology-formation aspects. In this paper, we apply the topology formation framework in [16, 17] for P2P video streaming applications. To address the challenges in P2P streaming, the main goal is to form a topology with a link-level homogeneous property so as to eliminate overlay-link bottlenecks in delivering video streams between peers. We also propose a distributed algorithm to achieve such a link-level homogeneous overlay and conduct extensive simulations for performance evaluation.

## 2 System design

In this section, we present the proposed video streaming system in detail. We start with the notations for assisting the discussions and then elaborate the design philosophy and system components. Finally, we describe the algorithm implementation of the proposed adaptive topology formation algorithm and the source rate maximization scheme.

### 2.1 Notations

The notations used in the paper are summarized in Table 1. We model the P2P streaming overlay network using a directed graph $G(t) = \{V(t), E(t)\}$ at time $t$, where $V(t)$ denotes the set of participating peers, including the video server $s$, and $E(t)$ denotes the set of overlay links between peers. $N(t)$ is the peer population size, where $N(t) = |V(t)|$ and $N(t) \geq 2$. In this paper, we use the terms "peer" and "node" interchangeably. In this overlay streaming network, each node may download streaming content from multiple parent nodes; at the same time, it may upload streaming content to multiple child nodes. Let $I_i(t)$ denote the number of download connections or in-degree of peer $i$. Let $O_i(t)$ denote the number of upload connections or out-degree of peer $i$. Let $C_i$ denote the upload capacity of peer $i$. Let $d_i(t)$ denote the achieved download rate of peer $i$, where $d_i(t)$ is computed by summing the flow rates of all in-degree connections. To characterize the impact of peer churn, all these variables, except $C_i$, are functions of time $t$. We assume that for each node $i$ the upload capacity $C_i$ is a known system parameter. Note that the proposed topology formation algorithm is adaptive to the changes of peers' capacities in the network.

Without loss of generality, we only consider one video channel which all the peers are watching. The source node $s$ remains in the network with an in-degree $I_s(t) = 0$ at any time and a download rate $d_s(t) = 0$. The source streaming rate $r(t)$ can be controlled and adjusted by the server node $s$ to fully utilize the upload capacity contributed by peers and maximize the video quality.

**Table 1** Notations

| | |
|---|---|
| $V(t)$ | Set of peer nodes including the server node at time $t$ |
| $E(t)$ | Set of overlay connections at time $t$ |
| $N(t)$ | Number of nodes including the server node in the network at time $t$, $N(t) = \|V(t)\|$ |
| $\mathcal{N}$ | Average number of nodes in the network in the stable state |
| $I_i(t)$ | In-degree of node $i$ at time $t$ |
| $O_i(t)$ | Out-degree of node $i$ at time $t$ |
| $d_i(t)$ | Download rate of node $i$ at time $t$ |
| $t_i$ | The time when node $i$ joins the network |
| $C_i$ | Upload capacity of node $i$ |
| $\langle C \rangle$ | Average node upload capacity |
| $r(t)$ | Video playback rate at time $t$ |
| $\delta$ | Capacity per out-degree value of nodes at the link-level homogeneous state |
| $\delta_D$ | Capacity per out-degree value achieved by the distributed algorithm |
| $\delta_s(t)$ | Capacity per out-degree of the server at time $t$ |
| $m$ | Number of random walkers generated when a peer joins the network |
| $s$ | Server node |
| $\pi_i(t)$ | Probability that node $i$ is selected as a parent node using the distributed algorithm |
| $Z(t)$ | Sum of capacity square per out-degree values of the network at time $t$ |
| $\mathcal{Z}$ | Average sum of capacity square per out-degree values of the network at the stable state |
| $p_{ij}$ | Probability that a random walker at node $i$ is forwarded to node $j$ |
| $P$ | Transition matrix of the random walkers with elements $p_{ij}$ |
| $\lambda$ | Mean arrival rate of nodes to the network |
| $\mu$ | Mean departure rate of each node; $\frac{1}{\mu}$ is the mean sojourn time of each node |
| $n_{\min}$ | Minimum number of neighbors of a node in the Markovian plane |
| $n_{\max}$ | Maximum number of neighbors of a node in the Markovian plane |
| $TTL$ | Time-To-Live, number of steps that the random walkers traverse the network |
| $A_i(t)$ | Neighbor set of peer $i$ on the Markovian plane at time $t$ |
| $k_i(t)$ | Degree of peer $i$ on the Markovian plane at time $t$, $k_i(t) = \|A_i(t)\| + 1$, including the self-loop |
| $\|\cdot\|$ | Cardinality of a set |
| $\langle\cdot\rangle$ | Expectation of a random variable |

### 2.2 System overview

The proposed P2P video streaming system is shown in Fig. 1. There are two software components in the system: a rate-adaptive source and a peer streaming engine. The rate-adaptive source has an adaptive-rate video encoder, e.g., MPEG (Motion Picture Experts Group) [18] over a UDP/IP protocol suite. This adaptive encoder is able to tune the amount of video bits to compress different video frames by selecting an appropriate quantizer scale parameter (QSP) [19], which is controlled by the rate controller. At the receiver side, the downloaded video is decoded by the MPEG decoder and finally sent to the media player for video playback at the playback rate $r(t)$.

In our design, we assume that each node can always retrieve the desired video content from any selected parent node. This could be attained by implementing the random linear network coding [20]. Instead of forwarding the received video streams directly, each peer forwards the coded video blocks, which are the linear combinations of the downloaded video blocks. As the video content is evenly spread in the coded streams [11, 20], the receivers can download video blocks without any duplicate.

We also assume that the transmission bottleneck is the upload capacity of peers, instead of the network core or the download side. In the case of the service deployment of P2P video applications within an ISP, the bandwidth bottleneck normally occurs at the access link since the ISP generally provisions sufficient bandwidth inside its network. A recent measurement study [21] partially confirms that the bottlenecks are in the residential access and not in the network core. In practice, the backbone network is usually over-provisioned and optimized using traffic engineering techniques [22].

In our system, video streams are carried using the UDP transport. Therefore, peers are able to allocate the upload bandwidth explicitly among the out-going connections. In this work, we adopt the equal bandwidth allocation among out-going connections; therefore, the upload bandwidth of each out-going link from peer $i$ can be computed as $\frac{C_i}{O_i(t)}$.

The proposed system aims to fully utilize the upload capacities of peers and maximize the video playback rate given the constraint on the upload capacities of peers. The system is governed by the following optimization problem $J$ in Eqs. 1–3. Equation 2 ensures that the total bandwidth demand from peers does not exceed the overall bandwidth supply. Equation 3 ensures that the video playback rate does not exceed the download rate of any peers for smooth playback.

$$J: \text{ maximize } \quad r(t) \tag{1}$$

$$s.t. \quad \sum_{i \in V(t)} C_i \geq \sum_{i \in V(t) \backslash \{s\}} d_i(t), \tag{2}$$

$$d_i(t) \geq r(t), \quad \forall i \in V(t) \backslash \{s\}, \tag{3}$$

where $V(t)\backslash\{s\}$ represents all the receiving peers excluding the server $s$.

Note that Problem $J$ is a linear programming formulation. By examining the Karush-Kuhn-Tucker (KKT) conditions of Problem $J$, we can compute the optimal playback rate of Eq. 1 at time $t$ as

$$r^*(t) = \frac{\sum_{j \in V(t)} C_j}{N(t) - 1}, \tag{4}$$

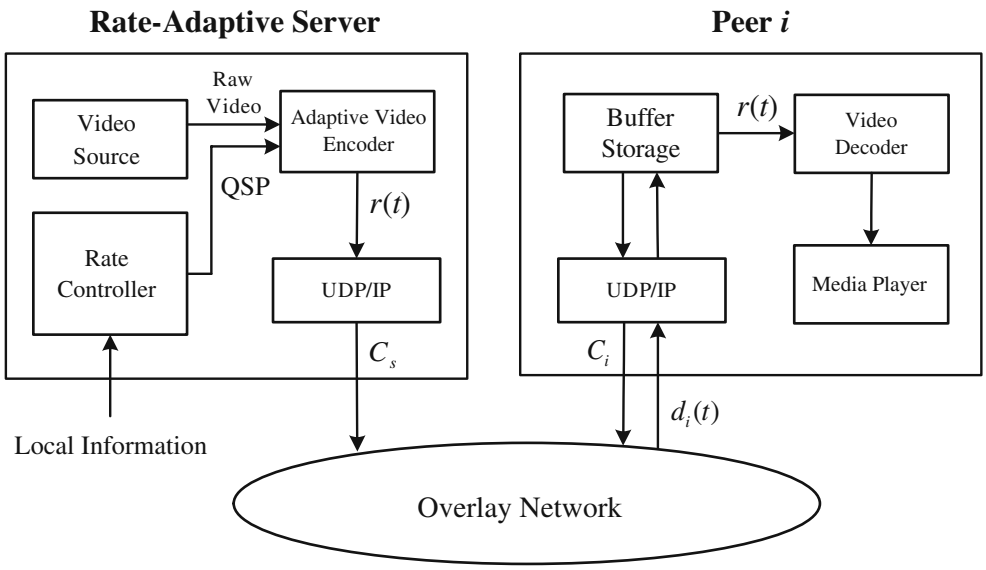


**Fig. 1** The proposed P2P video streaming system

and $\forall i \in V(t)\backslash\{s\}$,

$$d_i^*(t) = r^*(t) = \frac{\sum_{j\in V(t)} C_j}{N(t)-1}. \quad (5)$$

The derivation steps of Eq. 4 can be found in Appendix A. Equations 4 and 5 show that in the optimal state every peer is required to achieve the same download rate as the video playback rate. This is intuitive as in this case the playback rate is maximized to the rate all the nodes can support and bandwidth is fully utilized. This result has also been shown in multiple contexts [7, 23, 24].

Note that Eq. 5 only shows the property of the peers' download rate. Two crucial issues still remain: i) how to ensure that the server adapts the source rate following Eq. 4 because both $\sum_{i\in V(t)} C_i$ and $N(t)$ are unknown (Although $C_i$ of each individual peer is assumed known, to compute $\sum_{i\in V(t)} C_i$ in the network-wide is difficult, if still possible, when the network is dynamic and of large-scale.); ii) how to ensure that peers download at the optimal rate by Eq. 5 because the download rates are often throttled by the hidden bottlenecks in the overlay and are not easily predictable. To address the above two issues, we propose to construct a link-level homogeneous overlay in which all the overlay connections have an equal bandwidth allocation as shown in Eq. 6.

$$\frac{C_i}{O_i(t)} = \delta, \quad \forall i \in V(t), \quad (6)$$

where

$$\delta = \frac{\sum_{i\in V(t)} C_i}{\sum_{i\in V(t)} O_i(t)}. \quad (7)$$

Therefore, the download rate of peer $i$ is computed as

$$d_i(t) = I_i(t)\delta, \quad \forall i \in V(t)\backslash\{s\}, \quad (8)$$

because each in-coming flow is eliminated from any bottlenecks and achieves the allocated rate $\delta$. In addition, Eq. 8 shows that we can allocate peers with their desired rates by controlling their in-degrees. Hence, to achieve Eq. 5, we need that every peer has an equal in-degree, *i.e.*, $I_i(t) = m$, $\forall t > 0$ and $i \in V(t)\backslash\{s\}$ in Eq. 8. Therefore, we have

$$d_i(t) = m\delta = \frac{\sum_{j\in V(t)} C_j}{N(t)-1} = d_i^*(t), \quad \forall i \in V(t)\backslash\{s\}, \quad (9)$$

because $\sum_{i\in V(t)} O_i(t) = \sum_{i\in V(t)\backslash\{s\}} I_i(t) = m\,(N(t)-1)$ and $\delta = \frac{\sum_{i\in V(t)} C_i}{m(N(t)-1)}$.

With the link-level homogeneity, to select the optimal video playback rate by Eq. 4, the server does not need to collect any explicit feedback from peers. The server converges to the link-level homogeneous state with $\lim_{t\to\infty} \frac{C_s}{O_s(t)} = \delta$. If the server estimates $\delta$ by measuring its own capacity per out-degree value, the server can compute the download rates of its child peers and adjust the playback rate to achieve $r^*(t) = d_i^*(t)$ accordingly based on Eqs. 5 and 9. In this manner, the server node achieves the source rate adaptation.

In our streaming framework, we do not differentiate peers; hence, we enforce a universal playback rate $r(t)$. However, with some modification, we can enable a differentiated service in which different peers achieve different download rates and diverse video quality, respectively. This could be achieved by assigning peers with different in-degree values in Eq. 8 according to their required video quality. This differentiated video playback can be realized by using the multiple description codes (MDC).

In summary, the proposed streaming framework ensures that peers download at the optimal rate shown in Eq. 5 by constructing the link-level homogeneous overlay. Because the server also behaves as a normal uploading peer and converges to such a desirable link-level homogeneous state, the server can measure the video streaming rate to the peers and determine the appropriate video rate accordingly. In the rest of this section, we describe two algorithms on constructing such a link-level homogeneous overlay. The topology formation algorithm mainly determines how a peer joins the streaming topology upon its arrival and how the topology repairs itself due to peer departures from the network.

## 2.3 Algorithm design for topology formation

To construct a link-level homogeneous overlay, we first propose a centralized algorithm as a benchmark scheme, which requires a central controller to coordinate peers. This algorithm is efficient and implementable in a small-sized network; however, it is not scalable for a large-sized network. Therefore, we also propose a fully distributed algorithm, which shares the same design philosophy without requiring any global network information.

### *2.3.1 Centralized algorithm*

To implement this centralized algorithm, we assume that a central controller is used to collect network information and control the topology formation. The global

network information includes peers' upload bandwidth and the current out-degrees of all the participating peers.

*Peer joining* When peer $i$ joins the network at time $t$, to connect it to the overlay network, the controller sorts all the participating peers, including the server node, in a descending order according to their capacity per out-degree values as $\left\{\frac{C_{i_1}}{O_{i_1}(t)}, \frac{C_{i_2}}{O_{i_2}(t)}, \ldots, \frac{C_{i_{|V(t)|}}}{O_{i_{|V(t)|}}(t)}\right\}$, where $\frac{C_{i_1}}{O_{i_1}(t)} \geq \frac{C_{i_2}}{O_{i_2}(t)} \geq \cdots \geq \frac{C_{i_{|V(t)|}}}{O_{i_{|V(t)|}}(t)}$. For those peers with zero out-degree, we manually set their out-degree values to be $\varepsilon$ for the computation purpose, where $\varepsilon$ is a constant and $0 < \varepsilon < 1$. Note that the value of $\varepsilon$ only affects the initial positions of peers in this ordered list and does not impact the final sorting order once those peers have at least one child node. Then, the controller selects the first $m$ peers from this ordered list. When peer $i$ connects to the network, its initial out-degree is 0 and its initial in-degree is $m$.

*Peer departing* When peer $i$ departs from the network, all its child peers lose a download link, and their download rates decrease accordingly. To compensate this rate reduction, the controller will rebuild a new link for these peers. The rebuilding procedure is similar to the joining procedure, whereas on each rebuilding event, only one peer currently with the largest capacity per out-degree value is chosen to be connected as the parent node.

Using the centralized algorithm, as peers are continuously joining and departing, the capacity per out-degree values of all live peers converge to the same equilibrium value in Eq. 6. We illustrate the basic idea using a simple example shown in Fig. 2. Suppose that there are five peers $A$, $B$, $C$, $D$ and $E$. Their capacity per out-degree values are shown along the vertical axis. Initially, the network is heterogeneous in terms of capacity per out-degree or link bandwidth. However, when new peers join, peers with large capacity per out-degree are connected and their link bandwidth decreases. In this example, $A$ and $D$ are connected by the new peer. The out-degrees of $A$ and $D$ increase accordingly; hence, their out-going link bandwidth decreases. When their child nodes depart, peers with small capacity per out-degree (*e.g.*, peers $B$ and $E$) will have their capacity per out-degree increased. Finally, as peers continuously join and depart, the network converges to the link-level homogeneous state with all the peers' capacity per out-degree approaching the same value. Note that the convergence of peers' capacity per out-degree is driven by the peer arrivals and departures. The proposed algorithm is resilient to peer churn and can converge faster when the network becomes more dynamic. This effect will be demonstrated in Section 4 via simulations.

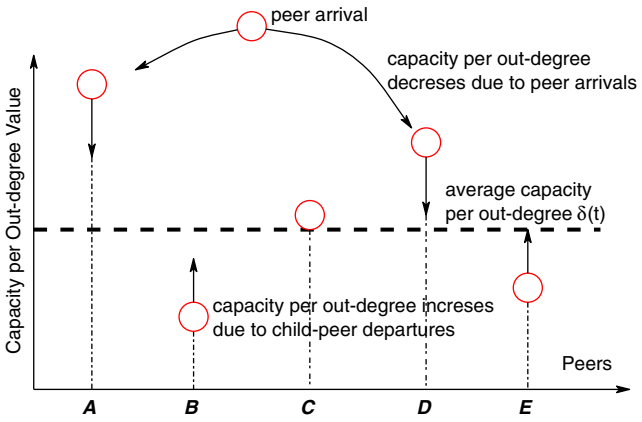


**Fig. 2** Evolution of the capacity per out-degree values in an example network

### 2.3.2 Distributed algorithm

The previous centralized algorithm achieves the link-level homogeneity in the formed topology; however, this algorithm is not scalable for a large-scale dynamic peer-to-peer network. We now propose a distributed algorithm to construct a link-level homogeneous overlay topology using a probabilistic sampling procedure. The analysis of this distributed algorithm is conducted in Section 3.

In this distributed algorithm, instead of seeking help from the central controller, the joining or rebuilding peers distributively find the appropriate parent peers in the network using the local information. Specifically, when peer $i$ connects to the network (in the joining or rebuilding procedure), it selects an existing node $j \in V(t)$ with the following probability

$$\pi_j(t) = \frac{\frac{C_j^2}{O_j(t)}}{\sum_{k \in V(t)} \frac{C_k^2}{O_k(t)}} = \frac{\frac{C_j^2}{O_j(t)}}{Z(t)}, \tag{10}$$

where

$$Z(t) \triangleq \sum_{k \in V(t)} \frac{C_k^2}{O_k(t)}. \tag{11}$$

By selecting parent peers following Eq. 10, peer $j$ with a large capacity but a small out-degree is selected with a high probability. Here, the choice of $\pi_j(t)$ is due to the requirements of the desirable link-level homogeneity which will be shown in the analysis section.

*Distributed node selection using a Markov chain Monte Carlo (MCMC) method* To implement the random peer selection procedure in Eq. 10 in a distributed manner, we use a Markov chain Monte Carlo (MCMC) method based on a random walk algorithm [25]. The basic idea is to construct a discrete-time ergodic Markov chain, embedded in the overlay topology. The states of the Markov chain are the nodes in the overlay graph, and the steady-state distribution of the formed chain equals to the targeted probability distribution defined in Eq. 10. The peer selection process is implemented using the distributed random walk algorithm. To select a peer by Eq. 10, a peer first issues a random walker starting at a randomly selected peer (or state) of the network. This walker is then routed among connected nodes based on the transition probability matrix of the Markov chain. The random walk is actually a process which starts from a random initial state of the Markov chain and then converges to the stable state after a sufficient number of state transitions. When the random walk process converges, the walker stays at a node according to the steady-state probability of the Markov chain and therefore accomplishes the selection.

To implement the random walk algorithm, we explicitly maintain two planes separately in the overlay network as shown in Fig. 3, namely, the Markovian plane and the overlay plane. The Markovian plane maintains the Markov chain and is only used to route the random walkers. The overlay plane is the formed overlay topology used for video delivery and this topology exhibits the link-level homogeneity. To ensure the convergence of the random walk algorithm, the Markov chain embedded in the Markovian plane must be ergodic, i.e., aperiodic, reversible and irreducible [26]. Therefore, we construct the Markovian plane to be an undirected graph with a self-loop connection at each node [27]. This setting ensures that the Markov chain is aperiodic.

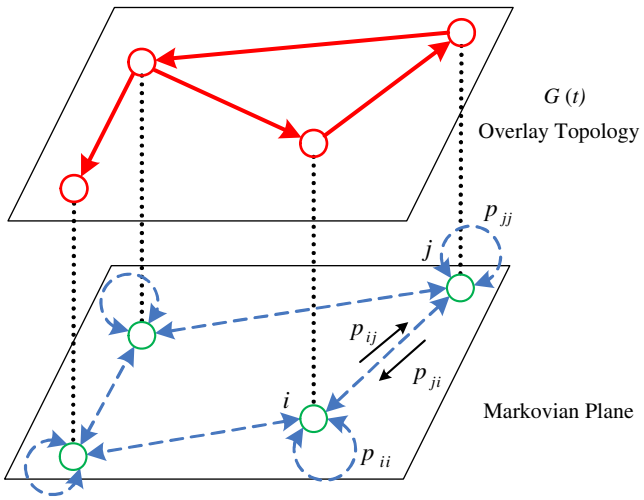


**Fig. 3** Distributed peer selection using a double-plane architecture

The main reason that we construct a Markovian plane is to help the random walk converge quickly. Because the length of the random walks significantly impacts the cost and the accuracy of peer selection, the Markovian plane should be constructed to minimize the cost of random walks with good accuracy [28]. To this end, we construct the Markovian plane as regular as possible in which the degree of each node is kept within a fixed region $[n_{\min}, n_{\max}]$, where $n_{\min}$ and $n_{\max}$ are predefined constants and $n_{\min} < n_{\max}$. This is because random regular graphs have been shown to be efficient in mixing time [29]. Using simulations, they showed that the random regular graphs have the smallest mixing time among the studied unstructured topologies. The maintenance of the Markovian plane is similar to the management of the virtual connections used for the rarest first algorithm in BitTorrent [30]. This mechanism appears to be of robustness and low cost in practice.

After constructing the graph of the Markovian plane, we still need to assign the connections with an appropriate transition probability matrix $P = [p_{ij}]$ so that the constructed Markov chain converges to the targeted probability distribution in Eq. 10. Thus, we use the celebrated Metropolis-Hastings algorithm as follows. Denote $A_i(t)$ as the neighbor set of peer $i$ on the Markovian plane at time $t$. The degree of peer $i$ is $|A_i(t)| + 1$, including the self-loop connection. Given the targeted steady-state probability $\pi(t) = \{\pi_i(t), \forall i \in V(t)\}$ in Eq. 10, $P$ is given as

$$p_{ij} = \begin{cases} \frac{1}{|A_i(t)|+1} \min\left\{ \frac{C_j^2 \cdot O_i(t) \cdot (|A_i(t)|+1)}{C_i^2 \cdot O_j(t) \cdot (|A_j(t)|+1)}, 1 \right\}, & j \in A_i(t), \\ 1 - \sum_{k \in A_i(t)} p_{ik}, & j = i, \\ 0, & \text{otherwise}, \end{cases} \tag{12}$$

where $p_{ij}$ is the transition probability from state $i$ to state $j$. As shown in Eq. 12, peer $i$ only needs to have the local information to compute the transition probability $p_{ij}$.

*Algorithm implementation* We are now ready to present the implementation of the distributed algorithm in detail. On peer arrivals and departures, both

the overlay plane and the Markovian plane are incrementally updated.

*Peer joining* To join the network, peer $i$ first contacts a bootstrap server and fetches a peer list. Note that the peer list accommodates more than $n_{\max}$ nodes and is timely obtained from the server. This mechanism is commonly used in P2P applications, such as BitTorrent. Then, peer $i$ randomly chooses $m$ initial nodes from this peer list and forwards a random walker to each of them. Each walker is then relayed among peers and traverses the Markovian plane based on the transition probability in Eq. 12. This process is described in Algorithm 1. The walker stops after some predefined Time-to-Live ($TTL$) steps, with each self-loop traversal counted as one step. Peers, which receive the stopped walkers, are selected as parent nodes to upload video stream to peer $i$ in the overlay topology.

**Algorithm 1** Peer joining algorithm using random walkers

```
Issue a walker to peer i_0 in the peer list
Set n ← 0
while n ≤ TTL − 1 do
  Select a peer j ∈ V(t) with probability p_{i_n j} based
    on Eq. 12 in the Markovian plane
  n ← n + 1
  i_n ← j
end while
Select peer i_n as a parent node
```

After updating the overlay topology, peer $i$ joins the Markovian plane by randomly connecting to $\frac{1}{2}n_{\max}$ peers selected from the peer list, where $\frac{1}{2}n_{\max} > n_{\min}$. Then both peer $i$ and the selected parent nodes update their local transition probabilities using Eq. 12.

*Peer departing* When peer $i$ departs from the network, each of its child peers in the overlay graph loses a parent node. To compensate for the degraded download rate, these child peers select a new parent node by issuing a walker to rebuild a new link. This procedure is similar to the joining procedure. Hence, each node in the overlay topology is responsible to maintain its own in-degree to be $m$, whereas its out-degree is adapted automatically in this distributed algorithm. As a result, the download performance of each peer is guaranteed by the constant number of the download connections and the guaranteed flow rate of each in-coming connection.

When a node departs, all its neighbor peers lose a link in the graph of the Markovian plane. To repair the graph, if the nodes' degrees are less than $n_{\min}$, they rebuild a new link by randomly selecting a peer in the Markovian plane. Otherwise, they do nothing. In this process, if a peer's degree in the Markovian plane has already reached $n_{\max}$, when the peer receives a connection setup request, it will deny this connection request. Therefore, the degree of peers in the Markovian plane is controlled within $[n_{\min}, n_{\max}]$. The pseudo codes associated with the above operations are described in Algorithm 2.

**Algorithm 2** Formation of the Markovian plane at peer $i \in V(t)$

```
INPUT  k_i(t): degree of peer i on the Markovian
plane at time t
Step 1: run the joining procedure
  Send requests and connect to peers randomly
  selected in the peer list until k_i(t) = (1/2) n_max
Step 2: run the rebuilding procedure, when
  neighboring nodes depart
if k_i(t) < n_min then
  Send connection requests to peers in the peer list
  Rebuild new connections until k_i(t) = n_min
end if
Step 3: run topology maintenance on receiving
connection requests
if k_i(t) < n_max then
  Accept the request
else
  Reject the request
end if
```

### 2.4 Bandwidth adaptive playback rate control

On adjusting the playback rate, the server first estimates the converged download rates of peers using Eq. 8 by measuring its own capacity per out-degree value. Based on this measurement, the video server adjusts the playback rate by Eq. 13.

$$r(t) = m\delta_s(t), \tag{13}$$

where $\delta_s(t)$ is the capacity per out-degree value of the server node at time $t$.

## 3 Algorithm analysis

In Section 2, we proposed a distributed algorithm to construct a link-level homogeneous overlay topology for high-performance P2P streaming. In this section, we analyze this link-level homogeneous property of the constructed topology $G$ in that the node capacity per out-degree value achieved by our proposed distributed

algorithm, $\delta_D$, converges to a constant value. Because we have assumed that the network bottleneck is at the upload link of the peers, the performance of this link-level homogeneity can be demonstrated using the metric, capacity per out-degree, of each peer at the upload side. In what follows, we first describe the network model. Then, we study the transient evolution of the out-degree of individual peers, and show that peer's capacity per out-degree value converges to an equilibrium point for the whole network. Finally, we discuss the possible algorithm extensions to enhance the performance of the topology formation and examine the tradeoff of the parameter settings in the algorithm.

### 3.1 Network model

We assume that each newly-arrived peer can connect to some parent peers very quickly. The search time of parent nodes is negligible. Each peer stays in the P2P network for a random amount of time duration, which is denoted as the peer life time. After this life time, the peer departs from the network. We also assume that both the peer arrival process and the departure process are independent. The peer arrival process is assumed to follow the Poisson process with the average rate $\lambda$, and the life time is independently and exponentially distributed with mean $\frac{1}{\mu}$. With the above assumptions, this P2P network can be modeled as an $M/M/\infty$ queue. Note that these modeling assumptions were also used in [31]. In [31] Pandurangan et al. showed that the network converges exponentially fast to a certain *stable state* when the mean peer arrival rate equals the mean departure rate. To simplify the analysis, we approximate the peer population $N(t)$ with the average population of peers $\mathcal{N} = \langle N(t) \rangle$, because our analysis focuses on the system performance at the stable state, in which $\lambda = \mathcal{N}\mu$ (that is, the peer arrival rate is equal to the total peer departure rate so that the number of peers in the network remains constant on average).

At the stable state, although the average peer population $\mathcal{N}$ converges to the value of $\frac{\lambda}{\mu}$, a node's out-degree still exhibits some dynamic behaviors when this node stays in the network. Let $t_i$ denote the instant when node $i$ joins the network. The out-degree $O_i(t)$ of node $i$ is a function of the joining instant $t_i$ and current time $t$. To highlight this, we denote the out-degree of node $i$ by $O_i(t_i, t)$ instead of $O_i(t)$ in this section. We start our analysis with the transient process of $O_i(t_i, t)$ at the stable state of the network and show its convergence to the link-level homogeneous property.

### 3.2 Transient evolution and equilibrium value of $O_i(t_i, t)$

The transient evolution of $O_i(t_i, t)$ can be characterized by the differential equation as

$$\frac{dO_i(t_i,t)}{dt} = (\lambda m + N(t)\mu m)\,\pi_i(t) - O_i(t_i,t)\mu, \tag{14}$$

where we approximate the discrete out-degree value by a continuous variable. Equation 14 computes the changing rate of $O_i(t_i, t)$ over time $t$ due to peer arrivals and departures. The first term on the right-hand-side of Eq. 14, $(\lambda m + N(t)\mu m)$, denotes the overall walkers generated by the arrival and departure peers. In particular, $\lambda m$ is the average walker generation rate by the newly-arrived peers. Note that peers arrive at the average rate $\lambda$ and each new peer issues $m$ walkers at its arrival using the distributed algorithm. $N(t)\mu m$ denotes the average walker generation rate by the rebuilding peers. The overall peer departure rate is $N(t)\mu$ because there are $N(t)$ peers and each peer has the average departure rate $\mu$. As shown in Appendix B, if each peer is of a constant in-degree $m$, the average peer's out-degree is approximately $m$. Therefore, each peer departure triggers $m$ peers to generate walkers to rebuild links. Hence, the overall generation rate of walkers due to peer departures is $N(t)\mu m$. Because the walkers finally stay at peer $i$ with probability $\pi_i(t)$, $O_i(t_i, t)$ thus increases at the rate $(\lambda m + N(t)\mu m)\,\pi_i(t)$. Because peer $i$ has $O_i(t_i, t)$ child peers with each peer departing the network at the average rate $\mu$, the second term on the right-hand-side of Eq. 14 computes the changing rate of $O_i(t_i, t)$ due to the departures of the child peers of peer $i$.

Next, we will show that the out-degree values of peers converge to an equilibrium value as shown in Eq. 15. Specifically, based on the transient evolution of $O_i(t_i, t)$ as shown in Eq. 14, when peer $i$ stays in the network long enough, its out-degree converges to a constant value as

$$\lim_{t\to\infty} O_i(t_i,t) = C_i \frac{4m}{\pi\,\langle C\rangle}. \tag{15}$$

The derivation of Eq. 15 is as follows. At the stable state, $\lambda = \mathcal{N}\mu$. Substituting it into Eq. 14 and approximating $N(t)$ by the average number of peers $\mathcal{N}$, we have

$$\frac{dO_i(t_i,t)}{dt} = 2\lambda m\pi_i(t) - O_i(t_i,t)\mu. \tag{16}$$

Substituting Eq. 10 into Eq. 16, we have

$$\frac{dO_i(t_i,t)}{dt} = 2\lambda m\frac{\frac{C_i^2}{O_i(t_i,t)}}{Z(t)} - O_i(t_i,t)\mu, \tag{17}$$

where

$$Z(t) = \sum_{j \in V(t)} \frac{C_j^2}{O_j(t_j, t)}. \tag{18}$$

As shown in Eq. 18, $Z(t)$ is a function of multiple random variables, including $V(t)$, $t_j$ and $C_j$. We assume that these random variables are independent. To solve Eq. 17, we approximate $Z(t)$ using its expectation $\langle Z(t) \rangle$ as

$$\langle Z(t) \rangle = \left\langle \sum_{j \in V(t)} \frac{C_j^2}{O_j(t_j, t)} \right\rangle. \tag{19}$$

We compute Eq. 19 by summing the expected capacity square per out-degree of peers which are still alive in the network at time $t$. On average during the time period of $[0, t]$, $\lambda t$ peers have joined the network in total. Nevertheless, some of these peers may have already departed from the network at time $t$. Let us first consider a randomly selected peer $j$, which has the joining time $t_j \in [0, t]$. The probability that peer $j$ is still in the network at time $t$ is $e^{-\mu(t-t_j)}$. Note that the arrival time $t_j$ is uniformly distributed in $[0, t]$ because in a Poisson process the arrival time of a random element is uniformly distributed in $[0, t]$ (see the remark of Section 5.3.5 on page 318 in [32]). We also assume that the upload capacities of peers follow some probability distribution.

$$\begin{aligned}
\langle Z(t) \rangle &= \left\langle \sum_{j \in V(t)} \frac{C_j^2}{O_j(t_j, t)} \right\rangle \\
&= \left\langle \int_0^t \lambda t e^{-\mu(t-t_j)} \frac{C_j^2}{O_j(t_j, t)} \frac{1}{t} dt_j \right\rangle \\
&= \left\langle \int_0^t \lambda e^{-\mu(t-t_j)} \frac{C_j^2}{O_j(t_j, t)} dt_j \right\rangle.
\end{aligned} \tag{20}$$

Equation 20 is now only the expectation with respect to $C_j$. At the stable state, Eq. 20 becomes

$$\mathcal{Z} = \lim_{t \to \infty} \langle Z(t) \rangle = \lim_{t \to \infty} \left\langle \int_0^t \lambda e^{-\mu(t-t_j)} \frac{C_j^2}{O_j(t_j, t)} dt_j \right\rangle, \tag{21}$$

where $\mathcal{Z}$ is a constant unrelated to time $t$ as shown in Appendix C. Substituting Eq. 21 into Eq. 17, we have

$$\frac{dO_i(t_i, t)}{dt} = 2\lambda m \frac{\frac{C_i^2}{O_i(t_i, t)}}{\mathcal{Z}} - O_i(t_i, t)\mu. \tag{22}$$

Solving Eq. 22, we have

$$O_i(t_i, t) = C_i \sqrt{\frac{2m\mathcal{N}}{\mathcal{Z}} \left(1 - e^{-2\mu(t-t_i)}\right)}, \ t \geq t_i. \tag{23}$$

Hence, the out-degree $O_i(t_i, t)$ converges to

$$\lim_{t \to \infty} O_i(t_i, t) = C_i \sqrt{\frac{2m\mathcal{N}}{\mathcal{Z}}}. \tag{24}$$

As shown in Eq. 23, when $\mu$ increases, the out-degree value converges faster to the equilibrium point $C_i\sqrt{\frac{2m\mathcal{N}}{\mathcal{Z}}}$. This suggests that if the network becomes more churning when peers depart at a higher rate, the peers' out-degrees converge faster. This effect is because the convergence to the link-level homogeneous state is driven by peer arrivals and departures. To compute $\mathcal{Z}$ in Eq. 24, we substitute Eq. 23 into Eq. 21 which yields

$$\mathcal{Z} = \frac{\pi^2 \langle C \rangle^2}{8m} \mathcal{N}. \tag{25}$$

The derivation steps of Eq. 25 are shown in Appendix C. Plugging Eq. 25 into Eq. 24, we obtain Eq. 15 in that when $t \to \infty$, $O_i(t_i, t)$ converges to a constant, $C_i \frac{4m}{\pi \langle C \rangle}$. Therefore, the capacity per out-degree value achieved by our proposed distributed algorithm is

$$\delta_D = \lim_{t \to \infty} \frac{C_i}{O_i(t_i, t)} = \langle C \rangle \frac{\pi}{4m}. \tag{26}$$

Equation 26 shows a desired LLH property that the capacity per out-degree of peer $i$ approaches to a constant. The download rate of peer $i$, $i \in V(t) \backslash \{s\}$, using our proposed distributed algorithm, is hence

$$d_i(t) = m\delta_D = \langle C \rangle \frac{\pi}{4}. \tag{27}$$

With $m$ increasing, $\delta_D$ becomes smaller. However, the achieved download rate of peers is not affected by $m$ as shown in Eq. 27.

### 3.3 Algorithm enhancement

As $\delta = \frac{\langle C \rangle}{m}$ shown in Eq. 7, it is important to note that $\delta_D < \delta$ with $\delta_D = \frac{\pi}{4}\delta$, in that the converged capacity per out-degree value using the distributed algorithm is smaller than that of the centralized algorithm. In this part, we propose two directions which could reduce the gap between $\delta_D$ and $\delta$, but requires more signalling overheads.

One possible reason of the gap between $\delta_D$ and $\delta$ is that the randomized peer selection scheme used in the decentralized algorithm introduces some probabilistic sampling errors. Specifically, using Eq. 10 to probabilistically select parent peers, some overloaded peers with small capacity per out-degree value may still be selected. These sampling errors can only be repaired when the child peers of the overloaded peers depart;

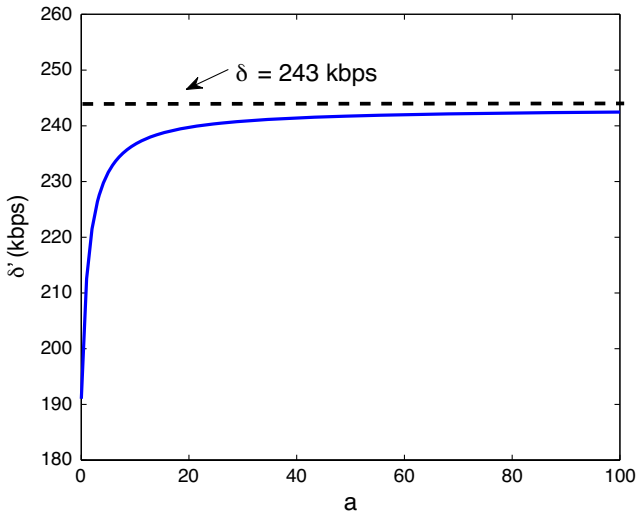


**Fig. 4** The effect of $a$ on $\delta'$

hence, the random peer selection slows the convergence to link-level homogeneity. To remedy this, we can enhance the accuracy of the peer selection with the modified probability $\pi_j(t)$ in Eq. 10 as

$$\pi_j(t) = \frac{\frac{C_j^{\alpha+1}}{O_j^{\alpha}(t_j,t)}}{\sum_{k\in V(t)} \frac{C_k^{\alpha+1}}{O_k^{\alpha}(t_k,t)}}, \tag{28}$$

where $\alpha$ is a positive real number and $\alpha$ equals 1 in Eq. 10. By increasing $\alpha$, the peers with larger capacity per out-degree values tend to be selected with higher probabilities; the overloaded peers will hence be selected with smaller probabilities (see [17] for a more detailed discussion on $\alpha$). In an extreme case, when $\alpha \to \infty$, $\pi_j(t) \to 1$ if peer $j$ has the largest capacity per out-degree value and 0 for other peers. This is the same as the centralized algorithm. However, a larger TTL value is required because more peers need to be examined in the selection process and hence a higher overhead is incurred.

Another way to reduce the gap between $\delta_D$ and $\delta$ is by enabling peers to quickly repair the errors if the overloaded parent peers are connected. To the end, we can enable peers to proactively disconnect a parent peer and re-select a new one at a mean rate $\mu'$ using the distributed algorithm. In this case, the evolution of the out-degree of peer $i$ is

$$O_i(t_i,t) = C_i\sqrt{\frac{\mathcal{N}}{\mathcal{Z}'}\left(\frac{2m\mu + m\mu'}{\mu+\mu'}\right)\left(1-e^{-2(\mu+\mu')(t-t_i)}\right)}, \quad t \geq t_i. \tag{29}$$

As indicated by Eq. 29, this proactive disconnection has the effect of increasing the departure rate of child peers; hence, the overloaded peers with too many child peers can quickly reduce their workloads and their capacity per out-degree converges to the equilibrium value as

$$\delta' = \lim_{t\to\infty} \frac{C_i}{O_i(t_i,t)} = \frac{\langle C\rangle A\,(1+a)}{m\,(2+a)}, \tag{30}$$

where $a = \frac{\mu'}{\mu}$, $A = \int_0^1 \frac{1}{\sqrt{1-x^{2(1+a)}}}dx$.

The derivation steps of Eqs. 29 and 30 are shown in Appendix D. The value of $\delta'$ with different $a$ is plotted in Fig. 4 where $m = 5$ and $\langle C\rangle = 1216$ kbps, which are computed based on the parameter settings in Tables 2 and 3. Note that in this example, the ideal capacity per out-degree is $\delta = \frac{\langle C\rangle}{m} = 243$ kbps. When $a \to \infty$, $A \to 1$ as shown in Eq. 52 in Appendix D. Therefore, by Eq. 30, we have $\delta' \to \frac{\langle C\rangle}{m} = \delta$. By making the re-selection rate $\mu'$ sufficiently large, we can tune the converged capacity per out-degree value to approach the value achieved by the centralized algorithm. However, this proactive operation should be conducted carefully since the network becomes even more dynamic with much more frequent link disconnections and reconnections. The perceived video quality may suffer as a result. Hence, $a$ achieves a tradeoff between the convergence of the link-level homogeneity and the video quality. When $a = 0$, our algorithm exploits the network dynamic itself to achieve the link-level homogeneity for the P2P network.

**Table 2** Default settings of the simulator

| $m$ | $n_{\min}$ | $n_{\max}$ | $TTL$ | $\varepsilon$ | $\lambda$ |
|---|---|---|---|---|---|
| 5 | 10 | 40 | 10 | 0.5 | 10 |

**Table 3** Capacity distribution of peers in the simulation

| Upload capacity (kbps) | 256 | 512 | 896 | 2048 | 5120 |
|---|---|---|---|---|---|
| Percentage | 25% | 30% | 20% | 15% | 10% |

## 4 Performance evaluation

### 4.1 Simulation setup

We conduct a simulation study to evaluate the performance of the proposed adaptive P2P streaming framework. The simulation is conducted using a session-level, event-driven simulator coded in C++. In each simulation run, there are 50,000 peer arrivals.

The peer arrivals follow the Poisson process with a mean rate $\lambda$ (peers per second). Each peer arrival is associated with two system parameters: peer life time and peer index. The peer life time follows the exponential distribution with a mean $\frac{1}{\mu}$ (seconds). Once the life time of the peer is over, the peer departs from the network. The peer index increases incrementally as peers arrive continuously. In this live streaming system, there exists a server node indexed with 0. This server always stays alive in the network. In the simulation experiments, the network size initially starts with one, where there is only one server node. As the network size increases, the overall departure rate, $N(t)\mu$, also increases and the network size becomes stable when the mean overall peer arrival rate ($\lambda$) equals to the mean overall peer departure rate ($\mathcal{N}\mu$). The average peer population is then $\mathcal{N} = \frac{\lambda}{\mu}$ at the stable state. In our experiments, we keep the average peer population in the stable state to be 5000 and control $\lambda$ to achieve different peer churn levels in terms of peer arrivals and departures. The simulation results are averaged over 10 individual simulation runs. The default parameter settings of the simulator are shown in Table 2. The bandwidth distribution of peers is configured as shown in Table 3 to capture the peers' heterogeneous upload capacities.

In our experiments, we examine whether the link-level homogeneity can be achieved using the proposed topology formation algorithms. In particular, given different levels of bandwidth heterogeneity and peer churn, will this link-level homogeneity still be maintained in the formed overlay topology? Our evaluation is conducted from two performance perspectives: network-level and peer-level. The network-level performance provides macro understanding on the proposed algorithms whereas we also take a close investigation on the streaming performance of some randomly selected peers. Our experimental results also characterize the impact of various factors, including peer churn, upload bandwidth distribution, the number of random walkers and so on, on the link-level homogeneity of the constructed streaming topology.

## 4.2 Network-wide link-level homogeneity

Ideally, utilizing the proposed topology formation algorithms, the resultant overlay topology should lead to a constant capacity per out-degree value of each peer. In this subsection, we examine the capacity per out-degree value of each peer of the two proposed algorithms and two heuristic algorithms at randomly-selected snapshots after the simulation reaches a stable state. Consequently, we are also able to obtain peers' download rates and the playback rate adapted at the server.

### *4.2.1 Centralized algorithm*

In the centralized topology formation, a central controller manages the construction of the overlay topology. This controller needs to collect the peer information across the whole network. This requirement may be prohibited in a large-scale P2P network; nevertheless, the performance of the centralized algorithm provides a benchmark. In Fig. 5, we plot the distribution of capacity per out-degree value of peers using the centralized algorithm at $t = 3500$ s. The peers are sorted according to the increasing order of peer indices along the $x$-axis. Most peers' capacity per out-degree values converge to the theoretical value $\delta = 243$ kbps, which is computed using Eq. 7.

In Fig. 6, we plot the download rate of peers at the same time instant. All the peers have roughly the same download rate of $1,216$ kbps. This rate matches well with the theoretical value computed using Eq. 5. This value also matches the playback rate adjusted by the server, $1,216$ kbps. In addition, this value is the optimal streaming encoding rate as shown in Eq. 4, at which peers can achieve the maximal achievable visual quality.

### *4.2.2 Distributed algorithm*

In Figs. 7 and 8, we examine the network-level performance of the proposed distributed algorithm. In Fig. 7,

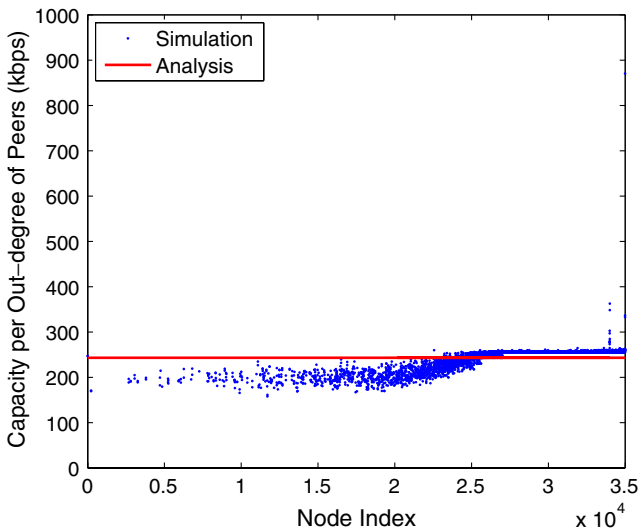


**Fig. 5** Capacity per out-degree value using the centralized algorithm at $t = 3500$ s

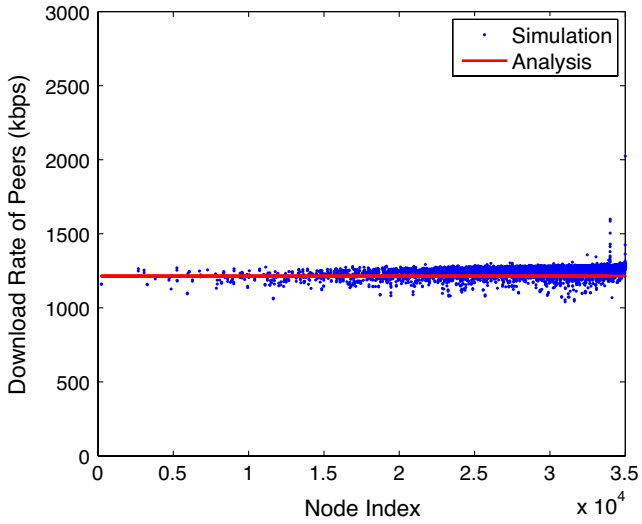


**Fig. 6** Download rate using the centralized algorithm at $t = 3500$ s

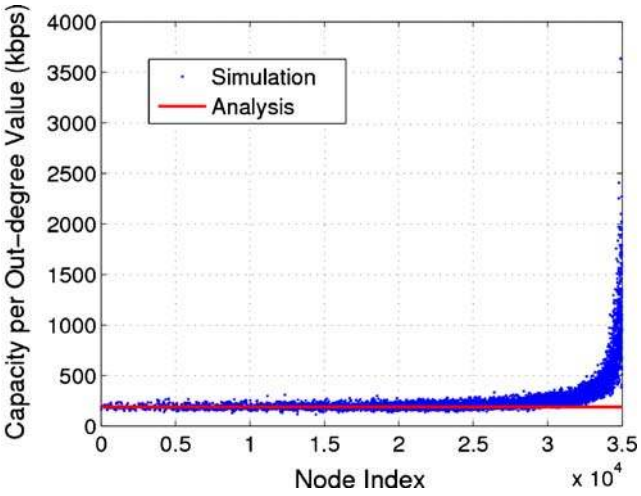


**Fig. 8** Distribution of capacity per out-degree using the distributed algorithm at $t = 3500$ s

we plot the histogram of the capacity per out-degree values of peers using the distributed algorithm. This histogram curve starts with a sharp impulse followed by a fast decaying tail. This curve suggests that in the constructed topology most participating peers have roughly an equal capacity per out-degree value. The peers at the end of the tail have a very large capacity per out-degree value. A closer look indicates that these peers are newly-joined peers in the network. Because these peers have not been connected by any other peers as their child nodes, their out-degree values are small. This observation can be shown clearly in Fig. 8, where we plot the capacity per out-degree value of the existing peers as the function of the peer index. We observe that the present peers with the index between [0, 34000] have roughly equal capacity per out-degree values. However, the newly-arrived peers with the index between [34000, 35000] have very large capacity per out-degree values, and contribute in the tail of the curve in Fig. 7. Note that the peer indices also indicate the time instants when the peers join the network, in that a peer with a large index joins the network late. Peers which stay in the network long have converged capacity per out-degree value. Comparing Figs. 5 and 8, we can observe that using the centralized algorithm, the capacity per out-degree of peers converges much faster than that using the distributed algorithm. Note that the converged capacity per out-degree value matches well with the theoretical value computed using Eq. 26, where $\delta_D = 191$ kbps.

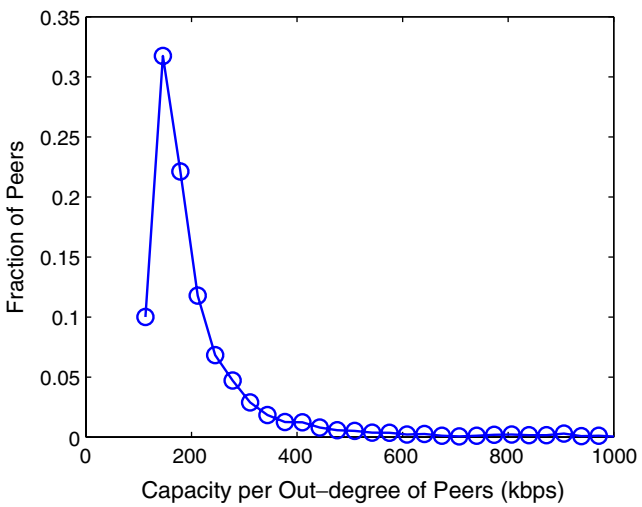


**Fig. 7** Histogram of capacity per out-degree using the distributed algorithm at $t = 3500$ s

In Fig. 9, we plot the download rate of peers using the distributed algorithm. The download rate of each peer is computed by summing all its parent nodes' capacity per out-degree values. Most of the peers have roughly the same download rate; nevertheless, those newly-arrived peers with the peer index between [34000, 35000] have large capacity per out-degree values. On the other hand, as the capacity per out-degree values of these newly-arrived peers are transient and converging, those large download rates gradually converge to the equilibrium. The playback rate in Fig. 9 is computed using Eq. 13. Many peers have download rates larger than this value and all the peers have download rates larger than 80% of the playback rate. In addition, as shown in Fig. 9, the playback rate is almost the same as the expected download rate of peers which is 955 kbps computed using Eq. 27. This result shows

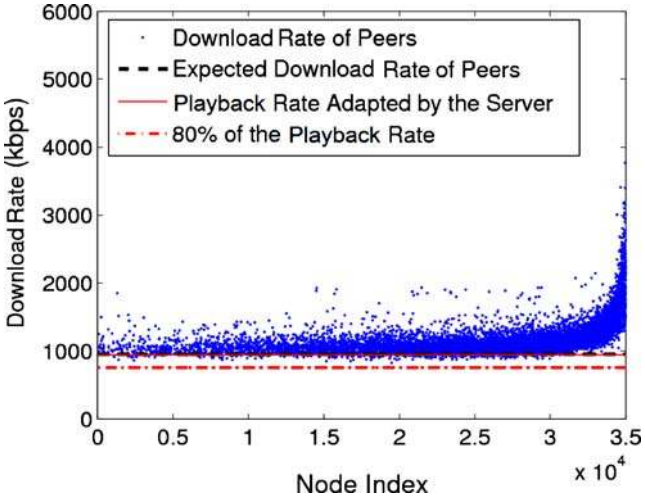


**Fig. 9** Download rate using the distributed algorithm

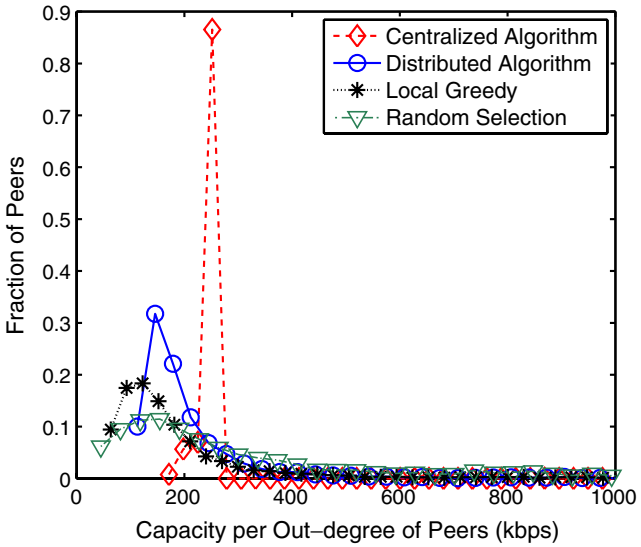


**Fig. 10** Capacity per out-degree using the four topology formation algorithms

that our adaptive playback rate control is effective to adapt to the optimal video quality shown in Eq. 4 using local measurements at the source.

In Fig. 6, the maximum playback rate is 1216 kbps using the centralized algorithm; nevertheless, using the distributed algorithm, as shown in Fig. 9, the maximum playback rate is 955 kbps. The distributed algorithm only achieves a sub-optimal performance in terms of the delivered video playback rate compared with the centralized algorithm. The achieved optimality ratio is then $\frac{955}{1216} = 0.78$. Unlike the distributed algorithm, the capacity per out-degree curve of the centralized algorithm does not have the heavy tail. This indicates that the centralized algorithm is much more efficient in utilizing the bandwidth of the newly-arrived peers and converges faster than the distributed algorithm.

#### *4.2.3 Comparisons with random and greedy algorithms*

We also compare the proposed centralized and distributed algorithms with two heuristic algorithms, the random algorithm and the greedy algorithm. In the random selection algorithm, each peer randomly selects its neighbors and then joins the overlay topology. This random topology formation scheme is commonly used in the data-driven content delivery such as BitTorrent [30] and CoolStreaming [28]. In the greedy algorithm, each peer maintains a neighbor list. When a peer needs to connect to a parent peer, it first measures the capacity per out-degree value of all its neighbors, and then selects the one with the largest value as its parent node. Similar greedy algorithms are adopted in [9, 13].

In Fig. 10, we plot the capacity per out-degree values of peers using the four topology formation algorithms, respectively. The centralized algorithm performs the best with a sharp impulse, with nearly 90% of the peers achieving the same capacity per out-degree value. Our distributed algorithm is less efficient than the centralized algorithm; however, it still significantly outperforms the random algorithm and the greedy algorithm. Unlike the myopic behaviors of the greedy algorithm, our sampling approach takes the bandwidth supply into consideration to construct a more efficient topology.

### 4.3 Convergence of the distributed algorithm

The previous experiments show that our distributed algorithm achieves a good link-level homogeneous property. In the remaining experiments, we evaluate the convergence property and the impact of various factors on our distributed topology formation algorithm. Figure 11 plots the capacity per out-degree value of peers at different time instant $t$. At different time instants, the capacity per out-degree value of peers converges to the same value.

We then study the dynamic evolution of peer's out-degree to the link-level homogeneous state. Specifically, we report the simulation results of the peer-level performance and investigate a typical peer in depth. This peer is randomly selected from the network. To facilitate the investigation on its out-degree evolution, this selected peer always stays in the network and never departs. The final simulation results are the average of 10 independent simulation runs.

We examine the evolution of the out-degree of the investigated node. The simulation results are compared with the theoretical curve based on Eq. 23 as $t \to \infty$. In

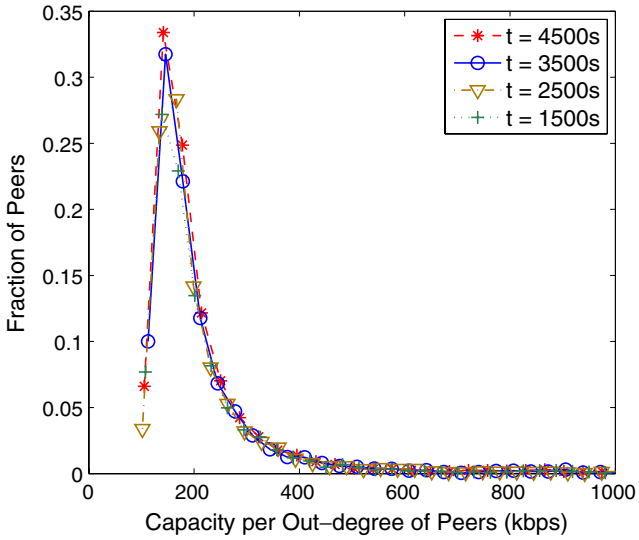


**Fig. 11** Capacity per out-degree at different time $t$ using the distributed algorithm

this experiment, we initially set the capacity of the investigated node to be 2048 kbps. At $t = 1,500$ seconds and $t = 3,000$ s, we change its capacity to 896 kbps and 5120 kbps, respectively. Figure 12 shows the evolution of the out-degree of this peer under these capacity changes. We observe that the out-degree of this peer evolves adaptively and converges to the theoretical values.

## 4.4 Impact of various factors on the distributed algorithm

In what follows, we examine the impact of peer churn, upload bandwidth distribution and the number of random walkers on the link-level homogeneity of the constructed streaming topology using the proposed distributed algorithm.

### *4.4.1 Impact of peer churn*

Besides bandwidth heterogeneity, another difficulty in designing an efficient P2P streaming network is due to peer churn. We define the churn rate as the frequency of arrival and departure events in a unit of time in our experiments. As the churn rate increases, the network becomes more dynamic; therefore, the streaming system must adapt and be repaired faster to minimize video playback freezing. In these experiments, we investigate the impact of peer churn on the network performance. In particular, we show that as the peer arrival rate increases, the convergence of peers' out-degrees can also speed up, which suggests that our system can adapt faster to combat the increasingly flash-crowd environment to maintain the streaming performance.

In our experiments, we control the peer churn rate by tuning the arrival rate $\lambda$. As $\lambda$ increases, the mean departure rate $\mu$ also increases as $\mu = \frac{\lambda}{\mathcal{N}}$ where $\mathcal{N}$ is fixed to be 5000. Based on Eq. 23, with the peer arrival rate increasing, the peer out-degree converges faster. To examine this effect, we ran the experiments with $\lambda = 10$ peers/second and $\lambda = 40$ peers/second, respectively. In Fig. 13, we observe that with $\lambda$ increasing the out-degree of the investigated peer can converge faster to the equilibrium value. In particular, the simulation results match well with the theoretical analysis in Eq. 23. This exhibits that our proposed distributed algorithm is resilient to the high churns and can adapt faster with an increasing churn rate.

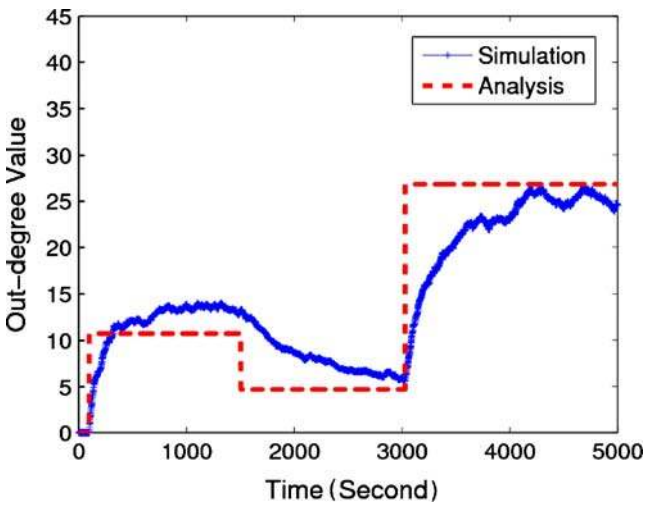


**Fig. 12** Evolution of out-degree of the investigated peer

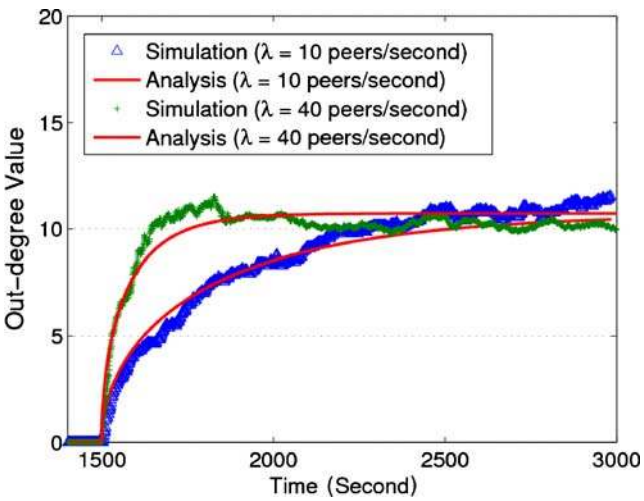


**Fig. 13** Impact of peer arrival rate on the investigated peer's out-degree

#### 4.4.2 Impact of changing capacity distribution

We examine the impact of changing capacity bandwidth on the proposed distributed algorithm. We first ran the simulation using the default parameter settings. At $t = 2000$ s when the 20000-th peer joins the network, we change the capacity distribution in Table 3 by increasing the corresponding capacity values by a factor of 3. Therefore, on average the capacity of the peers, which arrive after the capacity changes, increases by three times. This experiment is used to simulate the scenario as reported in [4], where peers in different regions and countries may join the network at different time periods due to the diurnal effect. Peers in different countries may have diverse bandwidth capacities. As a result, the capacity distribution of peers in the network may be changed significantly. Therefore, there may not exist a single video playback rate suitable for the whole broadcast duration. As such issues are quite general in P2P video streaming, we believe that the proposed adaptive-rate video streaming is particularly useful.

Now suppose that the peers arriving at $t \in [0, 2000]$ seconds (with the peer index approximately between $[0, 20000]$) are from a small-capacity region, *i.e.*, regions with less access bandwidth. The peers, which arrive after 2000 s, are from a large-capacity region, *i.e.*, regions populated with broadband network access. We simulate the case where the composition of peers changes over time when peers from different regions join the network to watch the video at different times.

In Fig. 14, we observe that in the initial state, where all the peers are from small-capacity regions, the download rate of the investigated peer is approximately 1 Mbps. At $t = 2000$ s, as peers from large-capacity regions starts to join the network, the overall bandwidth available in the network increases gradually and the download rate of the investigated peer increases as well. After a transient period, at around $t = 4000$ s, most peers from the small-capacity region depart from the network; peers in the network are mainly from large-capacity regions. Therefore, the overall bandwidth stops increasing and the download rate of the investigated peer also converges to a new equilibrium of 3 Mbps. Note that this rate is three times the previous download rate when all the peers in the network are from small-capacity regions. During this process, we also observe that the playback rate is adaptive to this change and increases from around 1 Mbps to 3 Mbps. This observation verifies that our proposed algorithm can effectively adapt to the dynamic capacity changes and fully utilize the available bandwidth to maximize the video quality. In practice, the composition of peers changes slowly and smoothly, and the playback rate adaptation is conducted over a longer time scale than peer churn. Therefore, during this rate adaptation process, peers usually do not experience any rapid video quality fluctuation.

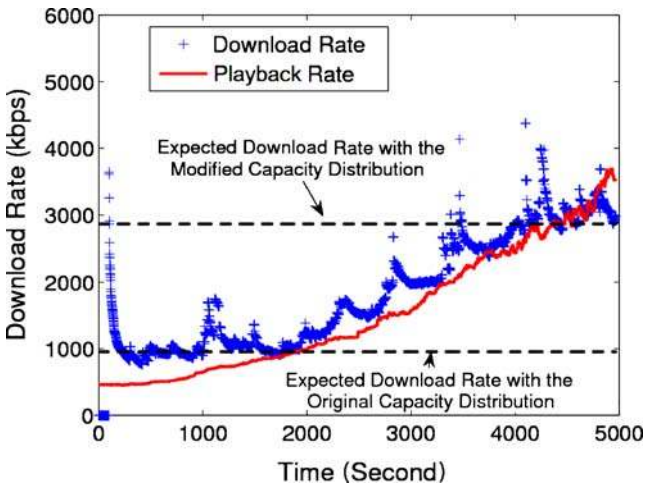


**Fig. 14** Impact of changing bandwidth distribution

#### 4.4.3 Impact of the number of random walkers

The number of random walkers used in the distributed algorithm has impact on the link-level homogeneous property of the constructed topology. In Fig. 15, we increase $m$ from 5 to 9. When $m$ increases, the converged capacity per out-degree value of peers becomes smaller, which can be explained using Eq. 26. On the other hand, the download rate of peers remain unchanged

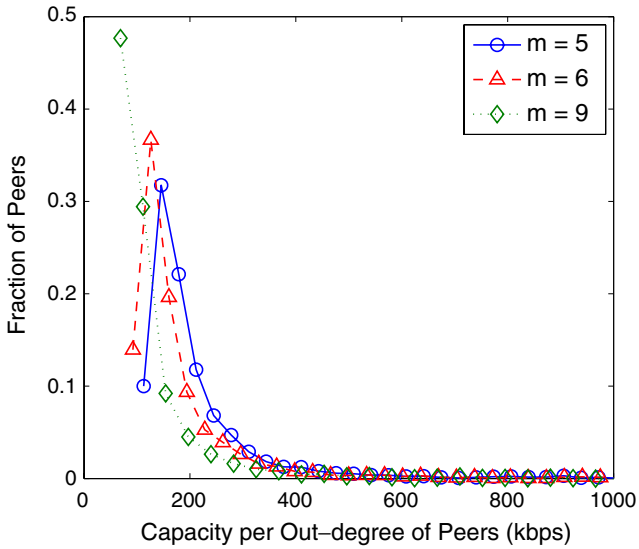


**Fig. 15** Distribution of capacity per out-degree values with increasing $m$ at $t = 3500$ s

regardless of the value of $m$ as shown in Eq. 27. An advantage of using a large value of $m$ is to maintain a downloading diversity to prevent a simultaneous failure of the parent nodes. However, a large value of $m$ can incur a high overhead in topology maintenance.

## 5 Conclusion

In this paper we propose an adaptive P2P video streaming system. This system automatically maximizes the video playback quality given the limited bandwidth supply in the network. On providing peers with maximized video playback quality, we construct a link-level homogeneous overlay topology to eliminate the bottleneck along the delivery paths of video flows. Equipped with the source rate adaptation and the homogeneous overlay construction, the proposed framework is able to fully utilize the bandwidth supply and provide users with a good video viewing experience. In particular, in order to construct a link-level homogeneous topology, we propose two algorithms for the implementation: the centralized version and the distributed version. We believe that this approach is also effective for the general resource allocation problems in overlay networks to improve the end-to-end overlay throughput.

## Appendix

### A Solution of problem $J$ in Eqs. 1–3

Problem $J$ is a linear programming problem. The KKT conditions [33] are both necessary and sufficient for Problem $J$. Let $d_i^*(t)$, $\forall i \in V(t)\backslash\{s\}$, be the optimal solution of Problem $J$. We introduce the Lagrange multiplier $\varphi$ to relax the constraint of Eq. 2. $\varphi$ is the shadow price if upload capacities are not fully utilized. We also introduce the Lagrange multiplier $\psi_i$ to relax the constraint of Eq. 3. $\psi_i$ is the shadow price if peer $i$ does not achieve the required download rate. Then, we have

$$L(d_i(t), r(t); \varphi, \psi_i) = r(t) + \varphi\left(\sum_{i\in V(t)} C_i - \sum_{i\in V(t)\backslash\{s\}} d_i(t)\right) + \sum_{i\in V(t)\backslash\{s\}} \psi_i\left(d_i(t) - r(t)\right).$$

We obtain the KKT conditions as follows.

$$\sum_{i\in V(t)\backslash\{s\}} d_i(t) \le \sum_{i\in V(t)} C_i; \quad d_i(t) \ge r(t),\ \forall i \in V(t)\backslash\{s\}, \tag{31}$$

$$\varphi \ge 0; \quad \psi_i \ge 0,\ \forall i \in V(t)\backslash\{s\}, \tag{32}$$

$$\varphi\left(\sum_{i\in V(t)} C_i - \sum_{i\in V(t)\backslash\{s\}} d_i(t)\right) = 0; \quad \psi_i\left(d_i(t) - r(t)\right) = 0, \quad \forall i \in V(t)\backslash\{s\}, \tag{33}$$

$$\frac{\partial L}{\partial d_i} = -\varphi + \psi_i = 0,\ \forall i \in V(t)\backslash\{s\}, \tag{34}$$

$$\frac{\partial L}{\partial r} = 1 - \sum_{i\in V(t)\backslash\{s\}} \psi_i = 0. \tag{35}$$

By Eq. 34, we have $\varphi = \psi_i, \forall i \in V(t)\backslash\{s\}$. If $\varphi = 0$, we have $\psi_i = 0, \forall i \in V(t)\backslash\{s\}$; however, this conflicts with Eq. 35. Therefore, $\varphi = \psi_i > 0, \forall i \in V(t)\backslash\{s\}$. Because $\varphi > 0$ and $\psi_i > 0$, by Eq. 33, we have $\sum_{i\in V(t)} C_i - \sum_{i\in V(t)\backslash\{s\}} d_i(t) = 0$ and $d_i(t) - r(t) = 0, \forall i \in V(t)\backslash\{s\}$, respectively. Therefore,

$$\sum_{i\in V(t)} C_i - (N(t) - 1)r(t) = 0.$$

Finally, we have

$$d_i^*(t) = r^*(t) = \frac{\sum_{j\in V(t)} C_j}{N(t) - 1},\ \forall i \in V(t)\backslash\{s\}. \tag{36}$$

### B Derivation of the average out-degree of nodes

In the following, we show that if each node is of a constant in-degree $m$, the average out-degree of nodes is approximately $m$.

Note that $\sum_{i\in V(t)} O_i(t_i, t) = \sum_{i\in V(t)\backslash\{s\}} I_i(t)$ as the overall out-degree equals to the overall in-degree, the expected out-degree of nodes is

$$\frac{1}{N(t)} \sum_{i\in V(t)} O_i(t_i, t) = \frac{1}{N(t)} \sum_{i\in V(t)\backslash\{s\}} I_i(t).$$

Because $\sum_{i\in V(t)\backslash\{s\}} I_i(t) = (N(t) - 1)m$ as each node has in-degree $m$ except the server $s$, we have

$$\frac{1}{N(t)} \sum_{i\in V(t)} O_i(t_i, t) = \frac{1}{N(t)}(N(t) - 1)m.$$

When $N(t) \gg 1$, we have

$$\frac{1}{N(t)} \sum_{i\in V(t)} O_i(t_i, t) \approx m. \tag{37}$$

## C Derivation of Eq. 25

To derive $\mathcal{Z}$, substituting Eq. 23 into Eq. 21 yields

$$\mathcal{Z} = \lim_{t\to\infty}\left\langle \int_0^t \lambda e^{-\mu(t-t_i)} \frac{C_i^2}{O_i(t_i,t)} dt_i \right\rangle = \lim_{t\to\infty}\left\langle \lambda C_i \int_0^t \frac{e^{-\mu(t-t_i)}}{\sqrt{\frac{2m\mathcal{N}}{\mathcal{Z}}\left(1-e^{-2\mu(t-t_i)}\right)}} dt_i \right\rangle. \quad (38)$$

Note that the $\langle\cdot\rangle$ operation is taken with respect to the peer capacity $C_i$ in Eq. 38. Therefore, we have

$$\mathcal{Z} = \langle C\rangle \lim_{t\to\infty}\int_0^t \frac{\lambda}{\sqrt{\frac{2m\mathcal{N}}{\mathcal{Z}}}} \frac{e^{-\mu(t-t_i)}}{\sqrt{1-e^{-2\mu(t-t_i)}}} dt_i. \quad (39)$$

By denoting $\tau = t - t_i$, Eq. 39 becomes

$$\mathcal{Z} = \frac{\lambda\langle C\rangle}{\sqrt{\frac{2m\mathcal{N}}{\mathcal{Z}}}} \lim_{t\to\infty}\int_0^t \frac{e^{-\mu\tau}}{\sqrt{1-e^{-2\mu\tau}}} d\tau. \quad (40)$$

Then, changing the integration variable $x = e^{-\mu\tau}$ and plugging it into Eq. 40, we have

$$\mathcal{Z} = \frac{1}{\mu}\frac{\lambda\langle C\rangle}{\sqrt{\frac{2m\mathcal{N}}{\mathcal{Z}}}} \int_0^1 \frac{1}{\sqrt{1-x^2}} dx. \quad (41)$$

Note that $\int_0^1 \frac{1}{\sqrt{1-x^2}} dx = \frac{\pi}{2}$ and plugging it into Eq. 41; therefore, we have

$$\mathcal{Z} = \frac{\pi}{2\mu}\frac{\lambda\langle C\rangle}{\sqrt{\frac{2m\mathcal{N}}{\mathcal{Z}}}}. \quad (42)$$

Plugging $\lambda = \mathcal{N}\mu$ into Eq. 42, we obtain

$$\mathcal{Z} = \frac{\pi}{2}\frac{\langle C\rangle}{\sqrt{\frac{2m\mathcal{N}}{\mathcal{Z}}}}\mathcal{N}. \quad (43)$$

Solving Eq. 43 for $\mathcal{Z}$, we finally have

$$\mathcal{Z} = \frac{\pi^2\langle C\rangle^2}{8m}\mathcal{N}. \quad (44)$$

## D Derivation of Eqs. 29 and 30

If the nodes reselect their parent nodes periodically, Eq. 14 becomes

$$\frac{dO_i(t_i,t)}{dt} = \left(\lambda m + N(t)\mu m + N(t)m\mu'\right)\pi_i(t) - O_i(t_i,t)\mu - O_i(t_i,t)\mu', \quad (45)$$

where the term $N(t)m\mu'$ appears because the nodes reselect each parent at an average rate $\mu'$; $O_i(t_i,t)\mu'$ is the additional rate of out-degree reduction because node $i$ has $O_i(t_i,t)$ child nodes which are disconnected from node $i$ at the average rate $\mu'$ due to this reselection operation. Solving Eq. 45, we have

$$O_i(t_i,t) = C_i\sqrt{\frac{\mathcal{N}}{\mathcal{Z}'}\left(\frac{2\mu m + m\mu'}{\mu+\mu'}\right)\left(1-e^{-2(\mu+\mu')(t-t_i)}\right)}, \quad t \ge t_i, \quad (46)$$

where $\mathcal{Z}' = \lim_{t\to\infty}\left\langle \sum_{i\in V(t)} \frac{C_i^2}{O_i(t_i,t)} \right\rangle$. As shown in Eq. 46, the convergence rate to the link-level homogeneous state increases by $\mu'$. $\mathcal{Z}'$ can be computed in the same manner as in Appendix C by substituting Eq. 46 into Eq. 21.

$$\mathcal{Z}' = \lim_{t\to\infty}\left\langle \int_0^t \lambda e^{-\mu(t-t_i)} \times \frac{C_i^2}{C_i\sqrt{\frac{\mathcal{N}}{\mathcal{Z}'}\left(\frac{2\mu m+m\mu'}{\mu+\mu'}\right)\left(1-e^{-2(\mu+\mu')(t-t_i)}\right)}} dt_i \right\rangle. \quad (47)$$

Denoting $a = \frac{\mu'}{\mu}$ and substituting it into Eq. 47, we have

$$\mathcal{Z}' = \langle C\rangle\sqrt{\frac{\mathcal{Z}'(1+a)}{\mathcal{N}m(2+a)}} \lim_{t\to\infty}\int_0^t \lambda e^{-\mu(t-t_i)} \times \frac{1}{\sqrt{\left(1-e^{-2\mu(1+a)(t-t_i)}\right)}} dt_i,$$

$$\sqrt{\mathcal{Z}'} = \langle C\rangle\sqrt{\frac{\mathcal{N}(1+a)}{m(2+a)}} \lim_{t\to\infty}\int_0^t \frac{\lambda e^{-\mu(t-t_i)}}{\mathcal{N}} \times \frac{1}{\sqrt{1-e^{-2\mu(1+a)(t-t_i)}}} dt_i. \quad (48)$$

Denoting $A = \lim_{t\to\infty}\int_0^t \frac{\lambda e^{-\mu(t-t_i)}}{\mathcal{N}} \frac{1}{\sqrt{1-e^{-2\mu(1+a)(t-t_i)}}} dt_i$ and plugging $\lambda = \mathcal{N}\mu$, we have

$$A = \lim_{t\to\infty}\int_0^t \mu e^{-\mu(t-t_i)} \frac{1}{\sqrt{1-e^{-2\mu(1+a)(t-t_i)}}} dt_i.$$

Changing the integration variable $x = e^{-\mu(t-t_i)}$, we obtain a simplified expression of $A$ in Eq. 49.

$$A = \int_0^1 \frac{1}{\sqrt{1-x^{2(1+a)}}} dx. \quad (49)$$

Plugging Eq. 49 into Eq. 48, we have

$$\mathcal{Z}' = \frac{A^2\langle C\rangle^2(1+a)}{m(2+a)}\mathcal{N}. \quad (50)$$

Substituting Eq. 50 into Eq. 46, we have

$$O_i(t_i,t) = C_i\frac{m(2+a)}{\langle C\rangle A(1+a)}\sqrt{\left(1-e^{-2(\mu+\mu')(t-t_i)}\right)}, \ t \ge t_i.$$

Hence, as $t \to \infty$, the converged capacity per out-degree value due to the proactive reselection of parent nodes is

$$\delta' = \lim_{t\to\infty} \frac{C_i}{O_i(t_i, t)} = \frac{\langle C \rangle A\,(1+a)}{m\,(2+a)}. \tag{51}$$

Note that the integration with respect to $x$ in Eq. 49 is conducted in [0, 1). As $a \to \infty$, $x^{2(1+a)} \to 0$. Therefore,

$$\lim_{a\to\infty} A = \lim_{a\to\infty} \int_0^1 \frac{1}{\sqrt{1-x^{2(1+a)}}} dx = 1. \tag{52}$$

## References


1. Chu Y-H, Rao SG, Seshan S, Zhang H (2002) A case for end-system multicast. IEEE J Sel Areas Commun 20(8): 1456–1471
2. Zhang X, Liu J, Li B, Yum T-SP (2005) CoolStreaming/DONet: a data-driven overlay network for peer-to-peer live media streaming. In: Proc IEEE conference on computer communications (INFOCOM), vol 3, Miami, pp 2102–2111
3. Liu J, Rao SG, Li B, Zhang H (2008) Opportunities and challenges of peer-to-peer internet video broadcast. Proc IEEE 96(1):11–24
4. Hei X, Liang C, Liang J, Liu Y, Ross KW (2007) A measurement study of a large-scale P2P IPTV system. IEEE Trans Multimedia 9(8):1672–1687
5. Luan H, Kwong KW, Huang Z, Tsang DHK (2008) P2P live streaming towards best video quality. In: Proc IEEE consumer communications and networking conference (CCNC), Las Vegas, pp 458–463
6. Luan H, Tsang DHK, Kwong KW (2006) Media overlay construction via a Markov chain Monte Carlo method. SIGMETRICS Perform Eval Rev 34(3):9–11
7. Kumar R, Liu Y, Ross KW (2007) Stochastic fluid theory for P2P streaming systems. In: Proc IEEE conference on computer communications (INFOCOM), Anchorage, pp 919–927
8. Huang C, Li J, Ross KW (2007) Can internet video-on-demand be profitable? In: Proc ACM conference on applications, technologies, architectures, and protocols for computer communications (SIGCOMM), Kyoto, pp 133–144
9. Cui Y (2005) Content distribution in overlay multicast. PhD dissertation, University of Illinois at Urbana-Champaign, Champaign
10. Wu C, Li, B (2005) Optimal peer selection for minimum-delay peer-to-peer streaming with rateless codes. In: Proc ACM workshop on advances in peer-to-peer multimedia streaming (P2PMMS), Hilton, Singapore, pp 69–78
11. Wang M, Li Z, Li B (2005) A high-throughput overlay multicast infrastructure with network coding. In: Proc IEEE international workshop on quality of service (IWQoS), University of Passau
12. Magharei N, Rejaie R (2007) PRIME: peer-to-peer receiver-driven mesh-based streaming. In: Proc IEEE (INFOCOM), Anchorage, pp 1415–1423
13. Small T, Li B, Liang B (2007) Outreach: peer-to-peer topology construction towards minimized server bandwidth costs. IEEE J Sel Areas Commun 25(1):35–45
14. Venkataraman V, Yoshida K, Francis, P (2006) Chunkyspread: heterogeneous unstructured tree-based peer-to-peer multicast. In: Proc IEEE international conference on network protocols (ICNP), pp 2–11
15. Sung Y-W, Bishop M, Rao S (2006) Enabling contribution awareness in an overlay broadcasting system. In: Proc ACM conference on applications, technologies, architectures, and protocols for computer communications (SIGCOMM), Pisa, pp 411–422
16. Kwong KW, Tsang DHK (2008) Building heterogeneous peer-to-peer networks: protocol and analysis. IEEE/ACM Trans Netw 2:281–292
17. Kwong KW, Tsang DHK (2007) Application-aware topology formation algorithm for peer-to-peer networks. In: Proc IEEE international conference on communications (ICC), Glasgow, pp 73–79
18. ISO-IEC/JTC1/SC29/WG11, DIS11172-1 (1992) Coding of moving pictures and associated audio for digial storage media up to 1.5 Mbps part 2, video. International Standard ISO-IEC/JTC1/SC29/WG11, DIS11172-1
19. Wu D, Hou YT, Zhu W, Zhang Y-Q, Peha JM (2001) Streaming video over the internet: approaches and directions. IEEE Trans Circuits Syst Video Technol 11(3):282–300
20. Gkantsidis C, Rodriguez P (2005) Network coding for large scale content distribution. In: Proc IEEE conference on computer communications (INFOCOM), vol 4, pp 2235–2245
21. Dischinger M, Haeberlen A, Gummadi KP, Saroiu S (2007) Characterizing residential broadband networks. In: Proc ACM SIGCOMM conference on internet measurement (IMC), San Diego, pp 43–56
22. Kwong KW, Guérin R, Shaikh A, Tao S (2007) Improving service differentiation in IP networks through dual topology routing. In: Proc ACM international conference on emerging networking experiments and technologies (CoNEXT), New York
23. Mundinger J, Weber R, Weiss G (2008) Optimal scheduling of peer-to-peer file dissemination. J Sched 11(2):105–120
24. Chiu DM, Yeung R, Huang J, Fan B (2006) Can network coding help in P2P networks? In: Proc international symposium on modeling and optimization in mobile. Ad Hoc and Wireless Networks (WiOpt), Boston, pp 1–5
25. Gilks W, Richardson S, Spiegelhalter D (1995) Markov chain Monte Carlo in practice (interdisciplinary statistics), 1st edn. Chapman & Hall/CRC, London
26. Ross SM (2006) Introduction to probability models. Academic, London
27. Jerrum M, Sinclair A (1997) The Markov chain Monte Carlo method: an approach to approximate counting and integration. In: Approximation algorithms for NP-hard problems. PWS, Boston, pp 482–520
28. Zhong M, Shen K, Seiferas J (2005) Non-uniform random membership management in peer-to-peer networks. In: Proc IEEE conference on computer communications (INFOCOM), vol 2, pp 1151–1161
29. Zhong M, Shen K (2006) Random walk based node sampling in self-organizing networks. SIGOPS Oper Syst Rev 40(3):49–55
30. Cohen B (2003) Incentives build robustness in BitTorrent. In: Workshop on economics of peer-to-peer systems
31. Pandurangan G, Raghavan P, Upfal E (2003) Building low-diameter peer-to-peer networks. IEEE J Sel Areas Commun 21(6):995–1002
32. Ross SM (2007) Introduction to probability models, 9th edn. Academic, Amsterdam
33. Boyd S, Vandenberghe L (2004) Convex optimization. Cambridge University Press, Cambridge

**Hao Luan** received the B.E. degree in Xi'an Jiaotong University, China in 2004 and the M.Phil. degree in electronic engineering from the Hong Kong University of Science and Technology, Kowloon, Hong Kong in 2007. He is now pursuing the Ph.D. degree at the University of Waterloo, ON, Canada. His current research interests focus on wired and wireless multimedia streaming, QoS routing in multihop wireless networks and vehicular network design.

**Xiaojun Hei** received the B.Eng degrees in Information Engineering from Huazhong University of Science and Technology, Wuhan, P.R. China, in June 1998. He also obtained his MPhil. Degree in Electrical and Electronic Engineering from the Hong Kong University of Science and Technology in August 2000. Then, he received his Ph.D. degree in the Department of Electronic and Computer Engineering at the Hong Kong University of Science and Technology in August 2008. From October 2008, he has been with the Internet Technology and Engineering R&D Center of the Department of Electronics and Information Engineering, Huazhong University of Science and Technology, P.R. China. He is now an associate professor in the Department of Electronics and Information Engineering, Huazhong University of Science and Technology. Between September 2005 and September 2007, he worked on P2P networking in the Department of Computer and Information Science, Polytechnic University. He is co-author (with Yong Liu and Keith W. Ross) of the best paper in multimedia communications for 2008 awarded by the Multimedia Communications Technical Committee of the IEEE Communications Society.

**Kin-Wah Kwong** received the B.E. degree (first class honors) and the M.Phil. degree in Electronic Engineering from the Hong Kong University of Science and Technology, in 2003 and 2005, respectively. He is now pursuing the Ph.D. degree at the University of Pennsylvania. His current research interests span the fields of network design and management, routing, traffic engineering, P2P networks and distributed systems.

**Danny H. K. Tsang** received the B.Sc. degree in mathematics and physics from the University of Winnipeg, Canada, in 1979, and the B.Eng. and M.A.Sc. degrees both in electrical engineering from the Technical University of Nova Scotia, Canada, in 1982 and 1984, respectively. He also received the Ph.D. degree in electrical engineering from the Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia, in 1989.

Upon the completion of his Ph.D. degree, he joined the Department of Mathematics, Statistics and Computing Science, Dalhousie University, Canada, where he was an Assistant Professor in the Computing Science Division. He has been with the Department of Electronic and Computer Engineering at the Hong Kong University of Science and Technology (HKUST), Kowloon, Hong Kong, since the summer of 1992. He is currently a Professor in the department. His current research interests include Internet quality of service, P2P video streaming over the Internet, and wireless multimedia networks. During his leave from HKUST in 2000–2001, he assumed the role of Principal Architect at Sycamore Networks in the United States. He was responsible for the network architecture design of Ethernet MAN/WAN over SONET/DWDM networks. He invented the 64B/65B encoding scheme (US Patent number 6 952 405) for Transparent GFP in the T1X1.5 standard which was later advanced to become the ITU G.GFP standard. The coding scheme has now been adopted by International Telecommunication Union (ITU)s Generic Framing Procedure recommendation GFP-T (ITU-T G.7041/Y.1303)).

Dr. Tsang was the General Chair of the IFIP Broadband Communications'99 held in Hong Kong and received the Outstanding Paper from Academe Award at the IEEE ATM Workshop'99. He also served as Technical Program Committee (TPC) members of ITC18-20, ACM Multimedia 2002, and INFOCOM 1994–96 and 2006–2007. He was an Associate Editor for the Journal of Optical Networking published by the Optical Society of America and also served as the Guest Editor for the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS special issue on Advances in P2P Streaming Systems. He currently serves as the General Chair of ICST AccessNets 2009 and an Associate Technical Editor for the IEEE Communications Magazine.