Capacity of P2P On-Demand Streaming with Simple, Robust and Decentralized Control

Can Zhao, *Student Member, IEEE,* Jian Zhao, *Student Member, IEEE,* Xiaojun Lin, *Member, IEEE,* and Chuan Wu, *Member, IEEE*

***Abstract*—The performance of large-scaled peer-to-peer (P2P) video-on-demand (VoD) streaming systems can be very challenging to analyze. In practical P2P VoD systems, each peer only interacts with a small number of other peers/neighbors. Further, its upload capacity may vary randomly, and both its downloading position and content availability change dynamically. In this paper, we rigorously study the achievable streaming capacity of large-scale P2P VoD systems with sparse connectivity among peers, and investigate simple and decentralized P2P control strategies that can provably achieve close-to-optimal streaming capacity. We first focus on a single streaming channel. We show that a close-to-optimal streaming rate can be asymptotically achieved for all peers with high probability as the number of peers N increases, by assigning each peer a random set of $\Theta(\log N)$ neighbors and using a uniform rate-allocation algorithm. Further, the tracker does not need to obtain detailed knowledge of which chunks each peer caches, and hence incurs low overhead. We then study multiple streaming channels where peers watching one channel may help in another channel with insufficient upload bandwidth. We propose a simple random cache-placement strategy, and show that a close-to-optimal streaming capacity region for all channels can be attained with high probability, again with only $\Theta(\log N)$ per-peer neighbors. These results provide important insights into the dynamics of large-scale P2P VoD systems, which will be useful for guiding the design of improved P2P control protocols.**

I. Introduction

Peer-to-Peer (P2P) Video-On-Demand (VoD) streaming systems have already become a major player on today's Internet. Their success (e.g., PPLive, TVAnts, UUSee, and Zattoo) has made high-quality on-demand streaming of rich contents available to millions of users at low server costs [2]. In contrast to their commercial success, however, in-depth theoretical understanding of these systems appears to be lacking. The performance of large-scale P2P VoD systems can be extremely complex to study. As time progresses, the part of the video that a peer is interested in viewing, the cached content that it can use to serve others, and its upload capacity can all change substantially. Further, these systems are highly decentralized in nature, and each peer often only has a very limited view of the overall system through its sparsely-connected neighbors. Due to these reasons, it remains a challenging problem to understand the fundamental performance limits of highly dynamic and decentralized P2P VoD systems.

In this paper, we study a problem of fundamental interest to P2P VoD systems, i.e., what is the optimal streaming rate that all peers can reliably receive, and how to achieve this optimal rate with simple, robust and decentralized control. Note that a trivial upper-bound on the streaming rate can be obtained by dividing the total upload capacity of all peers by the total number of peers. In P2P live-streaming systems, it has been shown in our prior work that streaming rates close to this optimal value can be achieved through simple and decentralized control [3]. However, in P2P VoD systems, it is unclear whether such an optimal rate can still be attained. In contrast to live-streaming [3]–[11], each peer in a VoD system is interested in playing a different portion of the video. Further, its viewing position may jump back and forth [12], [13]. As a result, the content availability at each peer can be highly discontinuous and dynamic. One way to alleviate this difficulty is to assume that some peers (referred to as "caches") have cached the entire video beforehand, and other downloading peers request the content only from the caches. In [14]–[16], the authors have studied the optimal cache-placement problem based on this assumption. An implicit assumption along this line of work is that there exists a central entity that can perfectly balance the downloading requests among caches. Otherwise, such a global balancing problem by itself can be very challenging in a decentralized setting when the upload capacity of the peers varies.

An alternate (and perhaps practically more relevant) approach is to directly model how peers downloading the same video can use their upload capacity to help each other, which is unfortunately more difficult. Such models were proposed in, e.g., [13], [17], [18]. However, it appears difficult to establish whether they can achieve close-to-optimal streaming rates. More recently, [19]–[22] proposes an algorithm that allocates the overall upload capacity in the system sequentially from the "oldest" peer to the "youngest" peer. For each peer, its requested capacity is first allocated from older peers. If there is no sufficient upload capacity, capacity is then requested from the server. Similarly, [23] proposes a global optimization problem for rate-allocation given the age of the peers. While these algorithms have been found to exhibit good performance, the resulting rate-allocation may need to be completely re-calculated when the peers' upload capacity changes. Further,

This work has been partially supported by the NSF through Grants CNS-0643145, CNS-0721484 and CNS-0831999, and Hong Kong RGC under contracts HKU 718513 and HKU 717812. An earlier version of this paper [1] has appeared in the 32th IEEE International Conference on Computer Communications (INFOCOM 2013).

C. Zhao was with the School of Electrical and Computer Engineering, Purdue University, West Lafayette. He is now with Qualcomm Technologies, Inc., San Diego (email: vskomirain@gmail.com).

J. Zhao and C. Wu are with the Department of Computer Science, The University of Hong Kong, Hong Kong (email: {jzhao, cwu}@cs.hku.hk).

X. Lin is with the School of Electrical and Computer Engineering, Purdue University, West Lafayette (email: linx@purdue.edu).

these analyses have not accounted for the possibility that the peers' playback positions may jump back and forth, in which case even an older peer may not have the content to serve younger peers.

In summary, existing analytical studies of the streaming capacity of P2P VoD systems either require extensive centralized control, are sensitive to upload-capacity variations or do not account for the random-seek behavior of the peers. In contrast, in this paper we provide the first rigorous study of the streaming capacity of large-scale P2P VoD systems with simple decentralized control that are robust to upload-capacity variations, and random-seek behaviors. We focus on the setting of "hot" videos, i.e., there are a large number of peers interested in viewing each video. We first study a single-channel system, i.e., all users are interested in viewing the same video. Assuming that the contribution of bandwidth and cache capacities from the dedicated server(s) is minimal, we show that by using a (properly-designed) random neighbor-selection algorithm and a uniform rate-allocation algorithm, with probability approaching 1 as the total number of peers N increases, all peers can achieve a close-to-optimal streaming rate of $(1-\epsilon)\mu$, where μ is the average upload capacity per peer and ϵ is a small positive constant. In our algorithm, each peer is only assigned $\Theta(\log N)$ upstream neighbors, with which they exchange content-availability information. These neighbors are chosen uniformly randomly from a suitable *choice set* determined at the tracker (note that this is the only part of the algorithm that requires centralized knowledge). To determine the choice set, the tracker only needs to know the current downloading position of each peer, but does not need to know the detailed content/chunk availability at each peer. Further, regardless of the variation of its upload-capacity, each peer evenly distributes its upload capacity among downstream neighbors for whom it has the available chunk(s). As readers will see in Section II, our analytical studies provide key insights as to why these simple design principles can result in near-optimal performance, which was conjectured in some prior simulation-based studies [24]. Further, these insights reveal the critical and non-trivial roles that different design choices, e.g., the size of the choice set and the extent of content availability, play in the overall system.

We then turn to a multi-channel P2P VoD system where different groups of peers are interested in viewing different videos. Based on the single-channel control algorithm discussed earlier, we propose a cache-placement algorithm that can achieve (with high probability) a close-to-optimal streaming rate region for all channels (see Section III for the precise definition). Our cache placement policy shares some similarity to the "proportional-to-deficit-bandwidth" strategy in [19], which was conjectured to be close-to-optimal. However, our policy does not require a sequential rate-allocation algorithm as in [19].

Our results have a similar flavor to the results in our earlier work [3] for P2P live-streaming systems. However, as we discussed earlier and will elaborate further in Section II, P2P VoD systems are significantly different from live-streaming systems. Thus, new control algorithms and analytic techniques are required. To the best of our knowledge, this work provides the first analytic result that demonstrates how to achieve close-to-optimal streaming capacity in large-scale P2P VoD systems using simple, robust, and decentralized control.

II. A Single-Channel P2P VoD System

In this section, we focus on a system with a single channel, i.e., all users are interested in viewing the same video. We first describe the system model. We will then propose simple, robust and decentralized peer selection and rate allocation algorithms that result in at most $\Theta(\log N)$ upstream neighbors per peer. We then prove that all peers can achieve the close-to-optimal streaming rate with high probability, when N is large.

A. System Model

We consider a P2P VoD system where users/peers[1] would like to watch a common video. Let $T^{(0)}$ denote the length of the video. There is a server S and totally N peers. Let $\mathcal{N}$ denote the set of all peers in the system, i.e., $|\mathcal{N}| = N$. We assume that the number of peers N is fixed. In other words, if a peer leaves the system, a new peer is assumed to immediately join the system at a possibly random initial position. This assumption simplifies the analysis, while we believe that the insights under this assumption will also hold for a more dynamic model where peers randomly join and leave the system. In a VoD system, the viewing/downloading progress of different peers in the same channel is typically different. Peers who have already downloaded certain parts of the video can then serve the cached content to later peers. We define the downloading position of a peer as the immediately next position in the video that the peer will download. We assume that, the downloading position of each peer is *i.i.d.* according to a distribution with density function $\gamma(t)$. In other words, for a small δt, $\gamma(t)\delta t$ is the probability that the downloading position of a peer is between t and $t+\delta t$. Clearly, the exact form of the distribution $\gamma(t)$ will depend on many factors, such as a new peer's initial viewing position, the length of time before it leaves the channel, and how often and in what manner it fast-forwards/backwards. However, the exact dependency on these factors will likely be quite complicated. Instead, in this paper we do not assume a probabilistic model for these factors. Rather, we only assume that, given these random factors, the overall system is stationary and ergodic, and thus the distribution $\gamma(t)$ exists. In practice, this distribution can often be estimated by the tracker in each channel [12]. Further, we do not assume a particular form of $\gamma(t)$ in this paper. As readers will see, our results will only depend on a small number of parameters of the distribution $\gamma(t)$. Some of these parameters are defined below.

Note that the downloading position of a peer is typically larger than its viewing position, with some buffering in between to absorb any fluctuations in the downloading speed. Some peers who have finished watching a channel may stay for some period of time and serve other peers in the channel.

[1]We use the terms "user" and "peer" interchangeably throughout the rest of the paper.

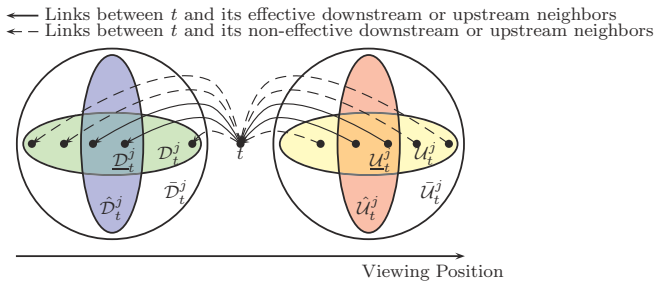


Fig. 1. Neighboring sets of peer t in channel j

We thus allow $\gamma(t)$ to have a Dirac delta function at point $T^{(0)}$. Equivalently, let $\bar{Q}$ denote the probability that a peer's downloading position is $T^{(0)}$. For ease of exposition, we assume that, with probability 1, the downloading position of each peer before $T^{(0)}$ is different from that of other peers. From now on, we will index a peer watching a channel by its downloading position t. Let $\mathcal{N}^-$ denote the set of all peers with downloading position $t < T^{(0)}$.

To model how peers serve other peers, each peer t has a set of downstream neighbors $\mathcal{D}_t$ that this peer t may upload content to. Correspondingly, each peer $t \in \mathcal{N}^-$ also has a set of upstream neighbors $\mathcal{U}_t = \{s \in \mathcal{N} | t \in \mathcal{D}_s\}$ that this peer t can potentially download the content from. However, since peer may perform random seeks, it may not have all the content "before" its downloading position. Hence, not all neighbors in the upstream neighbor set $\mathcal{U}_t$ of peer t have the requested content of peer t. We denote $\underline{\mathcal{U}}_t \subset \mathcal{U}_t$ as the set of upstream neighbors of peer t who have the data that peer t is requesting and is willing to serve peer t. Correspondingly, let $\underline{\mathcal{D}}_t = \{s | t \in \underline{\mathcal{U}}_s\} \subset \mathcal{D}_t$ denote the set of downstream neighbors that peer t can actually serve. We call $\underline{\mathcal{D}}_t$ and $\underline{\mathcal{U}}_t$ the *effective* downstream neighbors and the *effective* upstream neighbors, respectively. Let $U_t = |\mathcal{U}_t|$, $D_t = |\mathcal{D}_t|$, $\underline{D}_t = |\underline{\mathcal{D}}_t|$ and $\underline{U}_t = |\underline{\mathcal{U}}_t|$.[2] (See Fig 1 for illustration).

Let V_t denote the upload capacity of peer t. We assume that V_t is a bounded random variable between $[0, V_{\max}]$ with mean value μ, which is *i.i.d.* across all peers. Like other studies [3]–[5], [9], [10], we assume that the download capacity and the core network capacity are sufficiently large, and hence the upload capacity is the only resource bottleneck. The system performance is determined by the relationship between the targeted streaming rate and the downloading rates. Let R denote the targeted streaming rate of the video. Let $C_{s\to t}$ denote the streaming rate from peer s to peer t. Clearly, $C_{s\to t} = 0$ for any $s \notin \underline{\mathcal{U}}_t$ (or equivalently for all $t \notin \underline{\mathcal{D}}_s$). We have the following upload capacity constraint on each peer s:

$$\sum_{t\in\mathcal{N}} C_{s\to t} = \sum_{t\in\underline{\mathcal{D}}_s} C_{s\to t} \le V_s.$$

Let C_t denote the achievable downloading rate for peer t, which is then given by:

$$C_t = \sum_{s\in\mathcal{N}} C_{s\to t} = \sum_{s\in\underline{\mathcal{U}}_t} C_{s\to t}.$$

[2]As a convention, we will use script variable to denote a set (e.g., $\mathcal{U}_t$), and use a normal variable to denote its size (e.g., U_t).

To guarantee smooth playback, the downloading rate of each viewing peer must be no smaller than the targeted rate R of the video. Note that the peers whose downloading position is $T^{(0)}$ do not need to download new data, and hence we are only interested in the downloading rate of those peers in $\mathcal{N}^-$. We thus define the *streaming capacity* of the system as the largest value of R such that $C_t \ge R$ for all peers $t \in \mathcal{N}^-$.

We note that there is a simple upper bound on the streaming capacity. We assume that $\bar{Q}$ is away from 0 even with large N, and the contribution of the server capacity is negligible. In this case, it is easy to see that the largest possible streaming rate that all peers can attain is $\frac{N\mu}{N(1-\bar{Q})} = \frac{\mu}{1-\bar{Q}}$ on average. However, this upper bound completely ignores the details of the VoD system, especially whether a peer has the content and the upload capacity to help the other peer. Hence, it is unclear whether this upper bound is attainable in a large and decentralized VoD system. In practice, $\bar{Q}$ is usually not very large. Hence, in the rest of this section, we will omit the contribution of $\bar{Q}$ in the streaming capacity, and we will say that the channel achieves a close-to-optimal streaming capacity $(1-\epsilon)\mu$ with a small $\epsilon > 0$ if all peers attain a streaming rate no smaller than $(1-\epsilon)\mu$. Our goal in this section is to design simple, robust and decentralized algorithms that can achieve this close-optimal streaming capacity with high probability.

B. A Simple and Distributed Peer Selection and Rate Allocation Algorithm

In our prior work for P2P live-streaming systems [3], we proposed a simple peer selection strategy where each peer uniformly randomly selects $\Theta(\log N)$ downstream neighbors, and divides its upload capacity evenly among its downstream neighbors. This simple algorithm has been shown to achieve a close-to-optimal streaming rate for live-streaming P2P systems. Although this result serves as a useful starting point, as reader will see below, the same design would have led to very poor performance in VoD systems. Thus, we need to design a new set of control algorithms tailored to VoD systems.

(i) Peer Selection: We first explain why a uniformly-random peer-selection algorithm will not work well for VoD systems. Note that unlike live-streaming systems, in a VoD system different peers are viewing different parts of the video, and their cached content is also different. If an older peer (whose downloading position is in the later part of the video) chooses a younger peer (whose downloading position is in the earlier part of the video) as an upstream neighbor, there is a high chance that the younger peer does not have the content to help the older peer. Hence, the connection between them is of no use. This problem will be the most severe for the oldest peers that are close to the end of the video. With uniformly-random peer selection, the peers who are interested in downloading this part of the video will find that most of their selected upstream neighbors are younger and do not have the desired content. Hence, the streaming rate to these oldest peers will be very poor. Hence, we need to design a new peer selection strategy for VoD P2P systems.

A key idea of our new strategy is to restrict the random neighbor selection of each peer t to be done within a *choice*

set $\bar{\mathcal{U}}_t$, which contains peers with downloading positions larger than t. More specifically, we use the "random sequential choice-set selection strategy" as follows. Let Q be a constant such that $0 < Q < \bar{Q}$. In this strategy, the choice set $\bar{\mathcal{U}}_t$ of peer $t \in \mathcal{N}^-$ consists of the next NQ peers whose downloading positions are immediately larger than t's. If there are less than NQ peers after t and immediately before $T^{(0)}$, $\bar{\mathcal{U}}_t$ will be the set of all peers with downloading positions larger than t. In practice, the tracker can order all the peers according to their downloading positions and assign choice sets according to the above strategy. Recall our assumption that no two peers before $T^{(0)}$ are at the same downloading position. In practice, if this assumption does not hold, the tracker can always break ties arbitrarily. Then, the tracker server picks $M = \alpha \log N$ (where α is a positive constant to be determined later) peers uniformly randomly from peer t's choice set $\bar{\mathcal{U}}_t$, which constitute peer t's set of upstream neighbors $\mathcal{U}_t$. We have $\mathcal{U}_t \subset \bar{\mathcal{U}}_t$. Correspondingly, define the client set of peer t as $\bar{\mathcal{D}}_t = \{s \in \mathcal{N}^- | t \in \bar{\mathcal{U}}_s\}$. The set $\mathcal{D}_t$ of downstream neighbors of t must come from this client set and is given by $\mathcal{D}_t = \{s \in \mathcal{N}^- | t \in \mathcal{U}_s\}$. Let $\bar{U}_t = |\bar{\mathcal{U}}_t|$ and $\bar{D}_t = |\bar{\mathcal{D}}_t|$.

Remark: It appears that the tracker must maintain the current downloading position of all peers, which may incur high overhead. However, as we will explain later, by enforcing that all peers advance their downloading position at the same speed, this overhead can be significantly reduced.

(ii) Content Availability: Even with the above peer-selection strategy, the streaming rate for some peer can still be very poor. This is because peers may fast-forward/backward in a VoD system. This discontinuous random-seek behavior means that a peer t may not always have all the content before t. Thus, even if a peer only picks an older peer as an upstream neighbor, the connection and the capacity may still be wasted. Unfortunately, the random-seek behavior of peers is quite complicated to model. To the best of our knowledge, no existing analytical works on P2P VoD systems are able to take into account the impact of this random-seek behavior.

Our strategy is to develop a condition for content availability that is sufficient for achieving close-to-optimal streaming rates, yet easy to satisfy even with random-seeks. This is perhaps the most difficult part of our design. To see why such a condition is non-trivial to formulate, consider the following scenario. Suppose that the NQ peers in the choice set $\bar{\mathcal{U}}_s$ of peer s are uniformly distributed in the range $(t_0, s + \Delta)$, as shown in Fig. 2, where $t_0 > s$. Further, suppose that each peer $t \in (t_0, s + \Delta)$ only has the content in (t_0, t). The above scenario can occur when many peers random-seek to t_0 from before s (e.g., the opening of a movie ends at t_0, and thus most viewers wish to jump directly to t_0). In this case, although peer s may still have M upstream neighbors uniformly chosen from $\bar{\mathcal{U}}_s$, none of them can help peer s (because they do not have the content needed by s). Clearly, the key difficulty here is that, due to its particular position, peer s is unable to obtain any useful content from its upstream neighbors.

To address this difficulty, we introduce the following condition. Fix a positive constant $q_{\min} \in (0, 1)$. We require that, for any peer s and any one of its upstream neighbor t, the probability that peer t has the content for (and is willing to help) peer s is equal to $q_t > q_{\min}$, *independently of the position of peer* t. Note that if we were to satisfy this condition for the example in Fig. 2, we would need to make sure that, even when a peer random-seeks to t_0, the peer also downloads the additional content at position s with a probability no smaller than $q_{\min}$. A more detailed algorithm to achieve this condition is given in Appendix A. Clearly, when such a content availability condition is satisfied, the difficulty in our example in Fig. 2 will not occur. As we will see later, this condition will be sufficient for achieving a close-to-optimal streaming rate.

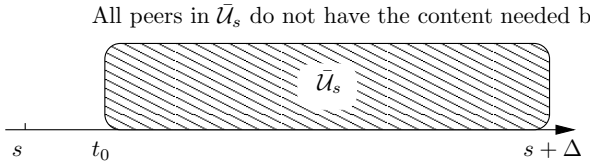


Fig. 2. Content Availability Example

We make a few remarks regarding the above condition. First, in order to meet this condition, only those peers who random-seek need to download a small amount of additional content (see Appendix A). According to real measurement studies in [12], each peer only jumps 1.6-3.4 times on average in each video. Thus, the overhead due to this additional download operation is low. For example, if the peers' viewing positions are uniformly distributed between $[0, T^{(0)})$, then the additional bandwidth will be no more than a fraction of $\frac{4Qq_{\min}}{1-Q}$ of the total required bandwidth of all users, which is not significant. Second, note that if one does not enforce the above content availability condition, an alternative option is to let the server stream data directly to such a peer s that does not receive a sufficient streaming rate from its own upstream neighbors. However, under this alternative option, peer s would have to ask the help from the server even though it did not perform any random seek. If the server capacity is not immediately available, the service to peer s would be disrupted. In contrast, under our content availability condition, only peers that random-seek need the help from the server. When a peer initiates a random seek, it expects a service disruption before the video can start at the new position. Thus, our content availability condition does not create additional disruption perceived by the user. Finally, depending on the value of $q_{\min}$, our content availability condition may even consume less additional server capacity than the alternate option. We will evaluate the required server capacity for the two options in Section IV.

(iii) Rate Allocation: To serve downstream neighbors, each peer applies a uniform rate-allocation algorithm that takes into account content-availability. Specifically, let $\hat{\mathcal{D}}_t \subset \bar{\mathcal{D}}_t$ denote the set of peers in peer t's client set $\bar{\mathcal{D}}_t$, whom peer t has the requested data and is willing to serve. We call $\hat{\mathcal{D}}_t$ the *effective client set* of peer t. Let $\hat{D}_t = |\hat{\mathcal{D}}_t|$. Thus, the *effective downstream neighbor set* $\underline{\mathcal{D}}_t$ of peer t will be the intersection of the effective client set and the downstream neighbor set of peer t, i.e., $\underline{\mathcal{D}}_t = \hat{\mathcal{D}}_t \cap \mathcal{D}_t$. Then, each peer divides its upload capacity equally among all of its effective downstream neighbors. Thus, the streaming rate from peer s

TABLE I
RELATIONSHIP BETWEEN $\bar{\mathcal{D}}_t$, $\hat{\mathcal{D}}_t$, $\mathcal{D}_t$ AND $\underline{\mathcal{D}}_t$. THE RELATIONSHIP BETWEEN $\bar{\mathcal{U}}_t, \hat{\mathcal{U}}_t, \mathcal{U}_t$ AND $\underline{\mathcal{U}}_t$ ARE SIMILAR.

$\bar{\mathcal{D}}_t$	client set of peer t (containing roughly NQ peers)
$\hat{\mathcal{D}}_t$	a subset of $\bar{\mathcal{D}}_t$ that peer t has the requested content and is willing to serve
$\mathcal{D}_t$	a subset of $\bar{\mathcal{D}}_t$ with size M that are actual downstream neighbors of peer t
$\underline{\mathcal{D}}_t$	intersection of $\hat{\mathcal{D}}_t$ and $\mathcal{D}_t$, which are the peers that peer t serves

to peer t, $C_{s\to t}$, is equal to $V_s/\underline{D}_s$ if $t \in \underline{\mathcal{D}}_s$, and $C_{s\to t} = 0$, otherwise. Correspondingly, we can define the effective choice set $\hat{\mathcal{U}}_t$ of peer t as the set of peers in the choice set $\bar{\mathcal{U}}_t$ who has the required content of peer t. We have $\underline{\mathcal{U}}_t = \hat{\mathcal{U}}_t \cap \mathcal{U}_t$. See Table I for a summary of the relationship between these notations. Note that for rate-allocation, peers only need to know the content availability information at their neighbors. There is no need for the tracker to maintain content availability information, which leads to low control overhead.

(iv) Uniform Progress: There remains one serious high-overhead problem. In a P2P VoD system, it is possible that some peer downloads content at a higher speed than others. If that is the case, the tracker needs to constantly update and re-order their downloading positions. Further, some upstream neighbors of peer t may either fall behind or advance too far ahead. As a result, the neighbors of each peer may need to be re-selected constantly. There will then be significant overhead at the tracker.

We introduce the following condition to significantly reduce the overhead. Suppose that the targeted streaming rate is $(1-\epsilon)\mu$ at the video's normal playback speed. We enforce that the downloading position of each peer will also advance ahead of its playback position at the normal playback speed of the video. In other words, even if the available download rate that a peer receives from its upstream neighbors is larger than $(1-\epsilon)\mu$, it will still download content at the speed of $(1-\epsilon)\mu$. On the other hand, if the download rate that a peer receives from its upsteam neighbors is less than $(1-\epsilon)\mu$, the server will fill in the gap. This condition ensures that the downloading positions of all peers advance at the same speed. In practice, the above design choice can be easily satisfied by the following protocol design: a peer will prefetch content for the video only up to a maximum lead-time ahead of its current playback position. Thus, once the maximum lead time is reached, the downloading position will advance at the same speed as the playback position advances, which is equal to the normal playback speed (i.e., streaming rate) of the video.

There are three benefits of this design. First, since the streaming rate of a video is known before-hand, the tracker can easily predict the advancement of each peer's downloading position. Unless a peer fast-forwards/backwards, there is no need for the tracker to update and re-order peers' downloading position. The measurement studies in [12] show that each user only seeks $1.6-3.4$ times in each video on average, which means that for our algorithm, each peer only needs to report to the server no more than 4 times on average for each video. As a comparison, the algorithms in [17]–[19], [23] require each user to report its content availability bitmap to the server periodically. Assume that each peer reports once every 5 minutes. Then, these algorithms will require each peer to report 20 times for a 1-hour video. Clearly, the signaling overhead of our proposed design is significantly lower. Second, the upstream neighbors and downstream neighbors of each peer do not need to change constantly either, unless a neighbor leaves the system or fast-fowards/backwards, in which case only this neighbor needs to be replaced via a low-complexity query to the tracker. Third, the above design choice not only leads to minimal control overhead, but also significantly simplifies our analysis because it is sufficient to focus on the streaming rates at a snapshot of time. On the other hand, some readers may be concerned that this design may unnecessarily constrain the downloading speed of those peers who could have downloaded faster. However, faster peers will likely take capacity from other slower peers, which will be unfair for the slower peers. Since our goal is to achieve the highest possible streaming rate for all peers, it is in fact more beneficial to maintain fairness. As we will show in our main result, our design is sufficient for attaining the close-to-optimal streaming capacity.

In summary, the above algorithm has the following highly-desirable features:

(i) Simplicity - Both random upstream-neighbor selection and uniform rate allocation are easy to implement in practice.

(ii) Robustness - If an upstream neighbor of peer t leaves the system, the tracker can simply assign another neighbor to t from its choice set.

(iii) Low control overhead - To carry out neighbor assignment, the tracker only needs to maintain the downloading positions of the peers. It does not need to know detailed content availability at peers. Further, the tracker can easily predict the advancement of peers' downloading position. Hence, the signaling overhead from peers to the tracker is greatly reduced.

C. Performance Analysis

We have proposed a simple and decentralized algorithm that is easy to implement, is robust to changes in the peers' upload capacity, and incurs low control overhead at the tracker. Next, we show that the above algorithm will attain close-to-optimal streaming rate. Recall from the content availability condition that $q_t \geq q_{\min}$ for all peers t, and C_t is the downloading rate of peer t.

Theorem 1. *For any $\epsilon \in (0,1)$ and $d > 1$, choose $\alpha \geq \frac{8d}{pq_{\min}\epsilon^2}$ with $p = \frac{\mu}{V_{\max}}$. Suppose that each peer chooses $M = \alpha \log N$ upstream neighbors. Then for sufficiently large N, the following holds*

$$\mathbf{P}\left(C_t \leq (1-\epsilon)\mu, \text{ for some } t \in \mathcal{N}^-\right) \leq O\left(\frac{1}{N^d}\right). \quad (1)$$

Theorem 1 shows that $\Theta(\log N)$ upstream neighbors are sufficient for achieving close-to-optimal streaming rate of $(1-\epsilon)\mu$ for all peers with high probability. Further, it provides additional insights on the required number of neighbors as a function of the system parameters. First, if we wish to achieve

a closer-to-optimal streaming rate (i.e., smaller ϵ) or a faster convergence of the probability (i.e., larger d), we need more neighbors per peer. Second, α is inversely proportional to $p = \frac{\mu}{V_{\max}}$. Hence, if there are higher levels of variation in the distribution of upload capacities (i.e., the peak rate $V_{\max}$ is large and/or a significant fraction of peers have small upload capacities), the required number of neighbors per peer must also be larger to tackle the extra level of randomness.

Another important consequence of Theorem 1 is that α is inversely proportional to $q_{\min}$. First, it is no longer necessary to ensure that an upstream neighbor of peer t always has the content that peer t requests (i.e., $q_t = 1$ for all t). According to Theorem 1, in order to ensure near-optimal streaming rates, it would be sufficient if each peer has at least $q_{\min}$ fraction of the content that its downstream peers will likely request. This relaxation significantly simplifies the system design when there are random-seeks. For example, the content availability strategy described earlier would be sufficient. On the other hand, in order to improve system performance, we should design P2P protocols with large values of $q_{\min}$, since it reduces the required number of neighbors.

For ease of exposition, we next provide a sketch of the proof of Theorem 1. The details will be provided in Appendix B. We first fix any peer t and show that the probability for its downloading rate C_t to be smaller than $(1-\epsilon)\mu$ is $\frac{1}{N^{2d}}$ (recall that $d > 1$). Theorem 1 then follows by taking the union bound. Note that peer t has exactly M upstream neighbors that may help it. Index these M upstream neighbors as $i = 1, ..., M$. Let I_i be the indicator function of the event that the i-th upstream neighbor of peer t is an effective upstream neighbor, and let $\mathbf{I} = [I_1, I_2, ...I_M]^T$. Then C_t can be represented by $C_t = \sum_{i=1}^{M} \frac{V_i I_i}{\underline{D}_i}$. We note that compared to our prior work [3] for live streaming, a main difficulty here stems from the number of effective downstream peers $\underline{D}_i$ for each upstream neighbor i. In [3], each upstream neighbor serves exactly M downstream peers. In contrast, here $\underline{D}_i$ is random and varies with an unknown parameter q_i. Further, there exists non-trivial correlation across i because the client sets of different upstream neighbors of peer t overlap. To address this difficulty, we use the following main supporting lemma.

Lemma 2. *Fix $q_{\min} > 0$. (a) Let $\tilde{\mathbf{I}} = [\tilde{I}_1, \tilde{I}_2, ..., \tilde{I}_M]^T$ be a set of M independent Bernoulli random variables such that $\mathbf{P}(\tilde{I}_i = 1) = q_i \geq q_{\min}$. (b) Let $\tilde{D}_i^+$, $i = 1, 2, ..., M$, be M positive (and possibly correlated) random variables such that $\mathbf{E}[\tilde{D}_i^+ | \tilde{\mathbf{I}}, \tilde{I}_i = 1] \leq \rho q_i M$ for some constant $\rho > 0$. (c) Let $\tilde{D}_i$, $i = 1, 2, ..., M$ be M positive (and possibly correlated) random variables such that for any $r_1, r_2, ..., r_M \geq 0$,*

$$\mathbf{E}\left[\exp\left(-\sum_{i=1}^{M}\frac{r_i}{\tilde{D}_i}\right)\middle|\tilde{\mathbf{I}}\right] \leq \prod_{i=1}^{M}\mathbf{E}\left[\exp\left(-\frac{r_i}{\tilde{D}_i^+}\right)\middle|\tilde{\mathbf{I}}\right]. \quad (2)$$

(d) Let $\tilde{V}_i$, $i = 1, 2, ..., M$, be M i.i.d. random variables independent from $\tilde{D}_i^+$'s and $\tilde{I}_i$'s such that $\mathbf{E}[\tilde{V}_i] = \mu$ and $0 < \tilde{V}_i < V_{\max}$ for all i. For any $d > 1$ and for any $\epsilon > 0$, let $\alpha \geq \frac{2dV_{\max}}{\epsilon^2 \mu q_{\min}}$. Then, there exists N_0 such that when $N > N_0$ and $M = \alpha \log N$, the following holds

$$\mathbf{P}\left(\sum_{i=1}^{M}\frac{\tilde{V}_i\tilde{I}_i}{\tilde{D}_i} \leq \frac{(1-\epsilon)\mu}{\rho}\right) \leq O\left(\frac{1}{N^{2d}}\right). \quad (3)$$

The proof is in [25]. We will soon relate $\tilde{I}_i$, $\tilde{D}_i$ and $\tilde{V}_i$ to I_i, $\underline{D}_i$ and V_i. To interpret the result of Lemma 2, note that if $\tilde{D}_i = \tilde{D}_i^+$ and $\tilde{D}_i^+$'s are independent from each other conditioned on $\tilde{\mathbf{I}}$, then the condition in (2) trivially holds. Using Jensen's inequality, it is then easy to see that $\mathbf{E}[\tilde{C}_t] \geq \mu/\rho$, where $\tilde{C}_t = \sum_{i=1}^{M} \frac{\tilde{V}_i\tilde{I}_i}{\tilde{D}_i}$. Lemma 2 implies that, as long as $M = \alpha \log N$, the probability that $\tilde{C}_t \leq (1-\epsilon)\mu/\rho$ will diminish to zero. The conditions in the lemma, however, allows the result to hold even if $\tilde{D}_i$'s are correlated, and hence is very useful.

We will use Lemma 2 to show Theorem 1. For ease of exposition, we consider instead an alternative choice-set selection strategy called "random sequential-range", which is slightly different from the "random sequential" choice set selection strategy that we originally used. In such a "random sequential-range" choice set selection strategy, each user t choose a choice set $\bar{\mathcal{U}}_t$ that contains all the other peers whose downloading position are in the range $(t, \phi'(t)]$, where $\phi'(t)$ satisfies that $\int_t^{\phi'(t)} \gamma(\tau)\mathrm{d}\tau = Q$, if $\int_t^{T^{(0)}-} \gamma(\tau)\mathrm{d}\tau \geq Q$, and $\phi'(t) = T^{(0)}$, otherwise. Correspondingly, the client set $\bar{\mathcal{D}}_t$ of each peer t contains all the peers in the range $[\psi'(t), t)$, where $\psi'(t)$ satisfies that $\int_{\psi'(t)}^{t} \gamma(\tau)(d)\tau = Q$, if $\int_0^t \gamma(\tau)\mathrm{d}\tau \geq Q$, and $\psi'(t) = 0$, otherwise. Clearly, for any $t < T^{(0)} - \psi'(T^{(0)})$, $\mathbf{E}[\bar{U}_t] = NQ$. When N is large, $\bar{U}_t$ should concentrate on NQ. Hence, we would expect that the performance of the two choice-set selection strategy are close to each other. A more general statement can be made as in the following lemma.

Lemma 3. *Let $\mathcal{X}$ be the collection of all continuous intervals $\Gamma \subset [0, T^{(0)})$. Fix $L \geq 1$. Given any $\epsilon \in (0, 1)$, define the following event*

$$\mathcal{A} = \left\{ \left| \frac{\sum_{l=1}^{L} n_l}{N} - \int_{\cup_{l=1}^{L}\Gamma_l} \gamma(\tau)d\tau \right| \leq \epsilon \int_{\cup_{l=1}^{L}\Gamma_l} \gamma(\tau)d\tau + \epsilon, \right.$$
$$\left. \text{for all disjoint } \Gamma_1, ..., \Gamma_L \in \mathcal{X} \right\},$$

where n_l is the number of peers in Γ_l. Then, for any $d > 1$, there exists N_0 such that for any $N > N_0$, $\mathbf{P}(\mathcal{A}) \geq 1 - O\left(\frac{1}{N^{2d}}\right)$.

The proof of Lemma 3 is provided in [25]. Note that if $\mathcal{A}$ happens, then the number of peers in every $\cup_{l=1}^{L}\Gamma_l$ will be close to its mean value. Lemma 3 states that such an event $\mathcal{A}$ happens with high probability. In the following, we will focus on the situation when event $\mathcal{A}$ holds. Let $\mathbf{P}_{\mathcal{A}}(\cdot)$ and $\mathbf{E}_{\mathcal{A}}(\cdot)$ denote the probability and the expectation conditioned on $\mathcal{A}$.

We are now ready to prove Theorem 1. Fix a peer t and its set of M upstream neighbors $i = 1, ..., M$. First, we note that I_i's are independent because the content availability of each upstream neighbor i is independent. Further, let q_i be the parameter introduced in the content availability condition in Section II-B. Then $\mathbf{P}(I_i = 1) = \mathbf{P}_{\mathcal{A}}(I_i = 1) = q_i \geq$

$q_{\min}$. Thus, condition (a) of Lemma 2 is met with $\tilde{I}_i = I_i$. Next, we will analyze the correlation between $\underline{D}_i$'s. Consider an upstream neighbor i. Let t_i be its current downloading position. If peer i recently random-sought to a position before t_i, let $t_{i_0} < t_i$ be the position that it first jumped to. Further, let Γ_i be the range of content from $[\psi'(t_{i_0}), t_{i_0}]$ that peer i randomly downloaded when it first jumped to t_{i_0}, according to the content availability strategy in Section II-B. Recall that the effective client set $\hat{\mathcal{D}}_i$ is a subset of $\bar{\mathcal{D}}_i$ that peer i has the requested content. $\hat{\mathcal{D}}_i$ consists of two parts: (a) all the n_1 peers in $\Gamma_i \cap [\psi'(t_i), t_{i_0})$, and (b) for the n_2 peers in $[t_{i_0}, t_i)$, each of them is in $\hat{\mathcal{D}}_i$ with probability q_i independent of others. Given $\mathcal{A}$ in Lemma 3, we must have, for any $\epsilon \in (0,1)$,

$$n_1 \le N\left(\int_{\Gamma_i \cap [\psi'(t_i), t_{i_0}]} \gamma(\tau)\mathrm{d}\tau + \epsilon\right)(1+\epsilon) \triangleq n_1^+,$$
$$n_2 \le N\left(\int_{[t_{i_0}, t_i]} \gamma(\tau)\mathrm{d}\tau + \epsilon\right)(1+\epsilon) \triangleq n_2^+.$$

Now, consider an alternative system by adding $(n_1^+ - n_1) + (n_2^+ - n_2)$ dummy peers. Construct a new set $\hat{\mathcal{D}}_i^+$ that contains all peers in $\hat{\mathcal{D}}_i$. In addition, the first group of $(n_1^+ - n_1)$ dummy peers are always added to $\hat{\mathcal{D}}_i^+$. For the second group of $(n_2^+ - n_2)$ dummy peers, each of them is in $\hat{\mathcal{D}}_i^+$ with probability q_i, independently of others. The advantage of making use of $\hat{\mathcal{D}}_i^+$ is that $\hat{D}_i^+$ only depends on I_i, t_{i_0} and Γ_i. Further, Γ_i and t_{i_0} are independent across i. Hence, $\hat{D}_i^+$'s are independent across i conditioned on $\mathcal{A}$. Further, $\hat{D}_i \le \hat{D}_i^+$ by our construction. Next, consider $\underline{\mathcal{D}}_i \subset \hat{\mathcal{D}}_i$, i.e., the set of effective downstream neighbors of i. For each peer in $\hat{\mathcal{D}}_i$, it randomly choose M upstream neighbors, one of which may be i. Further, for each dummy peers in $\hat{\mathcal{D}}_i^+$, we also let it choose peer i as an upstream neighbor with probability $\frac{M}{NQ}$. Let $\underline{D}_i^+$ be the number of effective downstream neighbors of i in this alternative system. Note that $\underline{D}_i^+$ may still be correlated across i (even though $\hat{D}_i^+$'s are independent). This is because the sets $\hat{\mathcal{D}}_i^+$ may overlap. Then, if an overlapped peer s has picked i as an upstream neighbor, it will be less likely to pick another upstream neighbor $i' \in \{1, 2, .., M\}$. Fortunately, we can show a negative dependency between $\underline{D}_i^+$'s. Specifically, if $\underline{D}_i^+$ is large, then it is likely that less peers will pick i', and hence $\underline{D}_{i'}^+$ will likely be small. This negative dependency is made precise in the following lemma (see Appendix B for proof).

Lemma 4. *For any $r_1, r_2, ..., r_M \ge 0$, $\underline{D}_i^+$'s satisfy*

$$\mathbf{E}_{\mathcal{A}}\left[\exp\left(-\sum_{i=1}^{M}\frac{r_i}{\underline{D}_i^+}\right)\middle|\mathbf{I}\right] \le \prod_{i=1}^{M}\mathbf{E}_{\mathcal{A}}\left[\exp\left(-\frac{r_i}{\underline{D}_i^+}\right)\middle|\mathbf{I}\right].$$

Note that $\underline{D}_i \le \underline{D}_i^+$ by our construction. Hence, condition (c) of Lemma 2 holds with $\tilde{D}_i = \underline{D}_i$ and $\tilde{D}_i^+ = \underline{D}_i^+$. To verify condition (b), We can show the following lemma based on the content availability condition. The proof is in [25].

Lemma 5. *Suppose $\gamma_{\min} \le \gamma(t) \le \gamma_{\max}$ for all $t \in [0, T^{(0)})$ for some $0 < \gamma_{\min} \le \gamma_{\max}$. For any $\epsilon \in (0,1)$, there exists K_0, such that for $K > K_0$, we have*

$$\mathbf{E}_{\mathcal{A}}\left[\underline{D}_i^+ \middle| \mathbf{I}, I_i = 1\right] \le (1+\epsilon)\, q_i M.$$

Thus, condition (b) of Lemma 2 holds. Finally, note that V_i's are i.i.d. and independent of all other random variables. Hence, Theorem 1 follows from Lemma 2 for the "random sequential-range" choice set selection strategy. One can then show that Theorem 1 also holds for our original policy (see [25] for details).

III. A Multi-Channel P2P VoD System

In the last section, we have focused on a single-channel P2P system. In this section, we study a multi-channel P2P system. Peers in each channel are interested in viewing a common video, which is however different across channels. Based on our single-channel algorithm, we will propose a simple and robust cache placement policy that could achieve a close-to-optimal streaming capacity for all channels.

A. System Model

We consider a P2P VoD system containing J channels. Let $\mathcal{J} = \{1, 2, ..., J\}$ denote the set of all channels, and $T_j^{(0)}$ denote the video length of channel j. Let $\mathcal{N}_j$ denote the set of peers that are watching channel j, and $N_j = |\mathcal{N}_j|$. Let $\mathcal{N}$ denote the set of all peers in the system, i.e., $\mathcal{N} = \bigcup_{j\in\mathcal{J}} \mathcal{N}_j$ and $N = |\mathcal{N}|$. We assume that $N_j = p_j \cdot N$, where p_j is the fraction of peers viewing channel j, which represents the popularity of channel j. Later on we will consider a system with large N, in which case we assume that p_j's are fixed and do not change with N, i.e., we focus on hot videos. Within each channel, we use the same model as Section II-A, except that a subscript or superscript j is added to each notation to denote the channel. For example, $\bar{Q}_j$, V_t^j and $\mathcal{D}_t^j$ represents the probability that a channel j peer's downloading position is at $T_j^{(0)}$, the upload capacity of a peer in channel j and the set of downstream neighbors of peer t in channel j, respectively. We assume that $\mathbf{E}[V_t^j] = \mu$ for all j, i.e., the upload capacity in each channel has the same distribution.

Using the results from Section II, we know that each channel j can sustain a maximum streaming rate around $(1-\epsilon)\mu$. However, in a multi-channel system, it is typical that different channels have different streaming rate requirements. Let R_j denote the targeted streaming rate for the video of channel j. Let $\mathbf{R} = [R_1, R_2, ..., R_J]^T$. Naturally, the streaming rate in some channel j may satisfy $R_j \le (1-\epsilon)\mu$, which implies that the upload capacity of peers viewing the channel is sufficient to support the targeted streaming rate. Such channels are referred to as *sufficient* channels. On the other hand, some other channel may have $R_j \ge (1-\epsilon)\mu$. We call such channels *insufficient* channels. We denote the set of insufficient channels as $\mathcal{I} = \{j \in \mathcal{J} | R_j > (1-\epsilon)\mu\}$, and the set of sufficient channels as $\mathcal{S} = \{j \in \mathcal{J} | R_j \le (1-\epsilon)\mu\}$. Seemingly, peers in an insufficient channel will not have enough upload capacity to stream the desired video.

A natural idea to improve the overall system performance is to use the extra capacity from sufficient channels to help the peers in insufficient channels. This kind of helping will obviously support a larger set of vectors $\mathbf{R}$ of streaming rate requirements. We define the *streaming capacity region* Λ of the multi-channel system as the set of streaming rate vectors, such

that for each $\mathbf{R} \in \Lambda$, under some centralized peer-selection and rate-allocation strategy, every peer in the system can receive a sufficient downloading rate R_j to view its desired channel. Assuming that the contribution of server capacity is minimal, the largest possible streaming capacity region is given by $\Lambda'_m = \left\{ \mathbf{R} \middle| \sum_{j=1}^{J} (1-\bar{Q}_j) N_j R_j \leq \sum_{i \in \mathcal{N}} \mathbf{E}[V_i] \right\}$. In other words, since the upload capacity of peers is the only in the system, the best we can do is to support those rate vectors $\mathbf{R}$ such that the summation of all demand is no greater than the summation of the overall upload capacity. Again, $\bar{Q}_j$'s are usually not very large in practice, and hence we will omit the contribution of $\bar{Q}_j$ in the rest of this section. Let

$$\Lambda_m = \left\{ \mathbf{R} \middle| \sum_{j=1}^{J} N_j R_j \leq \sum_{i \in \mathcal{N}} \mathbf{E}[V_i] \right\}.$$

We say that a multi-channel control algorithm achieves a close-to-optimal capacity region, if for any $\mathbf{R} \in (1-\epsilon)\Lambda_m$ with some $\epsilon > 0$, all peers in each channel j can sustain the streaming rate R_j.

In order for peers from a sufficient channel k to help peers in an insufficient channel j, the peers in channel k must already have the content for channel j, in addition to the content for channel k that they are interested in viewing. For this purpose, we assume that, in addition to the video from its own channel, each peer also caches, and hence can serve these cached videos to peers in those channels. (Note that although we assume that the entire video from other channels are cached in this case, a similar line of analysis can be carried out if the videos from other channels are divided into a small number of parts, and each peer only cached one parts of the videos.) Further, we assume that the cached content has already been pre-loaded, and we ignore the bandwidth resources to place these cached contents. We will then study the optimal placement probabilities for each video and how to best use the cached content. We note that a similar assumption of pre-loading cached content has been made in other prior works [14], [15], [19] that study the optimal cache placement probability. In practice, this kind of proactive deployment can be implemented in several ways. One possibility is to let the peers download the cached videos from the server during non-busy hours. Such a method is especially useful when the peers are always online, e.g., when using set-top boxes. Another possibility is to perform active push or passive replacement using a randomized algorithm [19]. The key assumption here and in [14], [15], [19] is that the cache content will change at a much slower time-scale than the content that each peer is interested in viewing. Hence, the cache replenishment process can be performed much more slowly, and thus the amount of bandwidth consumed for cache placement will be significantly smaller than the amount of bandwidth consumed for streaming.

There are, however, a common robustness issue for this line of work. The optimal cache-placement probabilities are often a function of system-wide parameters, such as the popularity of each video. When analyzing the system performance, it is often assumed that these parameters are known beforehand [14], [15], [19]. In practice, however, it can be difficult to accurately estimate these parameters before-hand. Further, as we discussed above, the cached content may need to be updated over a slow time-scale. Hence, the system parameters at the time of viewing may have already changed from those at the time of cache-placement. In summary, it is impractical to assume that the system parameters for optimal cache-placement probabilities are always known precisely beforehand. In the sequel, our goal is to develop a multi-channel control algorithm that can achieve close-to-optimal streaming capacity but is robust to imprecise estimates of key system parameters.

B. Algorithm and Performance

We start with our cache-placement algorithm, which has some similarity to the "proportional-to-deficient-bandwidth" policy in [19]. (However, note that its optimality is not rigorously shown in [19].)

(i) Cache Placement: As we discussed earlier, each peer will cache one other videos in addition to its currently-watching video. The tracker maintains which peers cache which videos. Given $\mathbf{R} \in (1-\epsilon)\Lambda_m$, the tracker determines the required number of additional helpers for each channel j, h_j^r, according to $h_j^r = \frac{p_j N R_j}{\mu\sqrt{1-\epsilon}} - \sqrt{1-\epsilon} p_j N$. Here, h_j^r can be interpreted as the deficit of upload bandwidth in channel j. Note that using h_j^r, the tracker can classify sufficient and insufficient channels: for a sufficient channel j, h_j^r is negative or zero; for an insufficient channel j, h_j^r gives a positive value. Every peer in each sufficient channel k caches a video randomly chosen from those of insufficient channels with the following distribution: the probability η_{kj} that a peer in channel k caches the video of channel j satisfies

$$\eta_{kj} = \eta_j \triangleq \frac{h_j^r}{\sum_{l \in \mathcal{I}} h_l^r}, \text{ for all } k \in \mathcal{S}, j \in \mathcal{I}. \tag{4}$$

Note that this probability only depends on R_j (video rate), μ (average upload capacity), p_j (video popularity), but is independent of N. Due to such a randomized cache placement policy, a random number of peers in each sufficient channel k cache a copy of channel j's video. Let us denote this number by $\tilde{H}_{kj}$. The total number of peers in sufficient channels that cache the video for channel j is then $\tilde{H}_j = \sum_{k \in \mathcal{S}} \tilde{H}_{kj}$. In our algorithm, the tracker randomly chooses H_{kj} peers among the $\tilde{H}_{kj}$ peers in channel k (which cache video j) to help channel $j \in \mathcal{I}$, where H_{kj} is given by

$$H_{kj} = \left\lceil \frac{|h_k^r h_j^r|}{\sum_{l \in \mathcal{S}} |h_l^r|} \right\rceil. \tag{5}$$

We call these H_{kj} peers "helpers" for channel j, and we use $\mathcal{H}_j$ to denote the set of all helpers assigned to help channel j. Note that if $H_{kj} > \tilde{H}_{kj}$, our algorithm would fail because there is not a sufficient number of peers who cache the video. However, we show in [25] that this failure probability goes to 0 as $N \to \infty$. Hence, the actual number of helpers for each channel j is $H_j \triangleq |\mathcal{H}_j| = \sum_{k \in \mathcal{J}} H_{kj}$.

***Remark*:** Although we have assumed that each peer only caches one video, the above algorithm could be easily extended to the case where each peer caches multiple videos. In

particular, if a peer in a sufficient channel k can cache multiple videos, then for each cache space, it chooses a video from those of insufficient channels with the same distribution in (4). Clearly, this design only increases the number of cached copies for videos of insufficient channels, and thus the likelihood that the tracker cannot find a sufficient number H_{kj} of "helpers" for any insufficient channel j will only decrease. Hence, our performance guarantee (shown later in Theorem 6) will still hold even when each peer can cache multiple videos.

Robustness to Imprecise Estimates of System Parameters: Along with the above cache-placement policy, our overall scheme for multi-channel P2P VoD systems achieves the following highly-desirable property of being resilient to imprecise estimates of system parameters. Note that, in the algorithm, the probabilities η_{kj} depend on several system-wide parameters, i.e., the mean upload capacity μ, the video popularity p_j, and the targeted streaming rate R_j of each channel j. In practice, these parameters, in particular the video popularity, may change over time. Hence, it would be difficult for the tracker to precisely estimate the value. Further, as we discussed earlier, the cache placement operations themselves also take time, and have to be based on predicted system parameters. These predicted system parameters can thus be quite different from the actual system parameters when peers request videos later. In such a scenario, while the number of required helpers, H_j, can be instantaneously computed from the current system configuration (e.g., the actual number of peers in each channel), the cache placement decisions $\tilde{H}_j$ must be pre-computed from possibly a different set of predicted system parameters. Nonetheless, for each insufficient channel $j \in \mathcal{I}$, our algorithm usually needs a much smaller number of helpers H_j than the expected number of caches $\mathbf{E}[\tilde{H}_j]$. Therefore, even if η_{kj} is computed from inaccurate system parameters, as long as $\tilde{H}_j \geq H_j$, we will still be able to find the required number of helpers, and the performance of our algorithm will not be negatively affected. Thus, our algorithm is robust to imprecise estimates of these parameters, as is shown in the following example.

Example: Consider a P2P VoD system with 3 different channels: channel 1 has $p_1 = 0.6$ and the streaming rate is $R_1 = 6$; channel 2 has $p_2 = 0.1$ and the streaming rate is $R_2 = 12$; channel 3 has $p_3 = 0.3$ peers and the streaming rate is $R_3 = 11$. Assume that the average upload capacity of all the peers is $\mu = 10$. One can show that $\mathbf{R} = [6, 12, 11]^T \in 0.81\Lambda_m$. It is not hard to see that channel 1 is a sufficient channel and channels 2 and 3 are insufficient channels. If we perform cache placement according to the proportions in (4), the expected number of peers in channel 1 to cache the video for channels 2 and 3 will be $\mathbf{E}[\tilde{H}_{12}] = \frac{1300}{7}$ and $\mathbf{E}[\tilde{H}_{13}] = \frac{2900}{7}$, respectively. However, the numbers of actual helpers that we need to assign to the insufficient channels are only $H_{12} = 44$ and $H_{13} = 97$, which are much smaller than $\mathbf{E}[\tilde{H}_{12}]$ and $\mathbf{E}[\tilde{H}_{13}]$. Therefore, it is not necessary to cache the videos exactly according to the proportions in (4). In particular, assume that the tracker estimates the system parameters incorrectly as $p_1^* = 0.3$, $R_1^* = 1$, $p_2^* = 0.3$, $R_2^* = 14$, $p_3^* = 0.4$, and $R_3^* = 9$, which are very different from the true parameters cited earlier. Based on these incorrect parameters, the expected number of peers chosen by our proposed algorithm to cache the video for channels 2 and 3 will then be $\mathbf{E}[\tilde{H}_{12}^*] = \frac{35400}{71} \approx 499$ and $\mathbf{E}[\tilde{H}_{13}^*] = \frac{7200}{71} \approx 101$. Since $\mathbf{E}[\tilde{H}_{12}^*] > H_{12}$ and $\mathbf{E}[\tilde{H}_{13}^*] > H_{13}$, the tracker will still be able to find enough helpers for channels 2 and 3 with high probability. Thus, the same performance guarantee of our algorithm (stated below in Theorem 6) can still be achieved.

We believe that this robustness property is a distinct advantage of this class of "proportional-to-deficit-bandwidth" strategies [19]. To the best of our knowledge, this work is the first to account for imprecise system parameters. The key intuition here is that since peers in the same channel already help each other, the number of additional helpers required is decided by the amount of deficit upload bandwidth in the channel, which is significantly smaller than the total amount of upload capacity needed. In contrast, in the model of [14], all peers must request services only from helpers that have cached the content before hand. In that case, the optimal cache-placement proportion must be accurate in order to serve the maximum number of requests. As a result, the performance is then very sensitive to imprecise estimates of system parameters.

(ii) Peer Selection and Rate Allocation: Each peer t in an insufficient channel j uniformly randomly selects M_N upstream neighbors from its choice set $\bar{\mathcal{U}}_t^j$ and uniformly randomly picks M_H upstream neighbors from its helper set $\mathcal{H}_j$, where $M_N + M_H = M$. Each peer in a sufficient channel only needs to select $M_N = M$ upstream neighbors from its choice set (i.e., $M_H = 0$ for peers in sufficient channels). Note that if a peer in a sufficient channel k is selected into the helper set $\mathcal{H}_j$ of an insufficient channel j, its upload capacity will be completely reserved for serving peers in channel j, and will not be used to serve peers in its own viewing channel. Each upstream peer still applies the uniform rate-allocation strategy. All other parts of the peer selection and rate allocation algorithms remain the same as in the single-channel case. We can show that with our simple multi-channel control algorithms, the targeted streaming rate of each channel can be attained with high probability. Specifically, let C_t^k be the achieve streaming of peer t in channel $k \in \mathcal{J}$. We have the following main result for multi-channel systems. Detailed analysis and proofs are provided in [25]

Theorem 6. *Given any $\epsilon \in (0,1)$, $d > 1$ and $\mathbf{R} \in (1-\epsilon)\Lambda_m$. Let $\epsilon' = 1 - \sqrt{1-\epsilon}$. There exists N_0 such that if $N \geq N_0$, $M = \alpha \log N$ and $\alpha \geq \frac{16d}{\min\{\rho_{\min}, p, 2\sigma_{\min} p\} q_{\min} \epsilon'^2}$, then we can find M_H and M_N such that*

$$\mathbf{P}\left(C_t^k \leq R_k, \text{ for some } k \in \mathcal{J} \text{ and } t \in \mathcal{N}_k^-\right) \leq O\left(\frac{1}{N^d}\right).$$

IV. Simulation Results

In this section, we provide simulation results of both single-channel and multi-channel systems to verify our analytical results.

Although our analytical results have not assumed a specific user-behavior model, in our simulations we will experiment with a particular user-behavior model, with which we will

verify that a steady-state distribution $\gamma(t)$ will indeed arise. Specifically, we use the following user-behavior model. The video length is set to be $T^{(0)} = 3600$ (seconds). Each user searches for upstream neighbors among the 500 peers ahead. We assume that a new peer always starts from the beginning of the video. Each peer could jump to a different location randomly after some time. More specifically, upon arrival or each jumping, each peer views the video for a random amount of time chosen from an exponential distribution with mean 1800 seconds. After this viewing period, the peer will jump to a new position between its current position and the end of the video uniformly randomly. After each jumping, each user downloads an additional $q_{\min} = 0.3$ fraction of the data in its client set, according to our content availability condition. This process (of viewing and jumping) then continues until the peer reaches the end of the video, after which the peer may stay for an additional amount of time uniformly distributed in $[0, 600]$s. Finally, the inter-arrival time between new peers follows an exponential distribution with mean $1/3$. By simulating the peers' behavior over time, we can observe the emergence of a steady-state distribution $\gamma(t)$ of the peers' viewing positions. For instance, in Fig. 3(a) each of the three curves corresponds to peers whose viewing positions are in the intervals $[0.05T^{(0)}, 0.15T^{(0)}]$, $[0.45T^{(0)}, 0.55T^{(0)}]$ and $[0.85T^{(0)}, 0.95T^{(0)}]$, respectively. The y-axis is the fraction of peers in each of the three intervals, while the x-axis is the simulation time in seconds. We can observe that these fractions indeed converge to a steady state, which suggests that a steady-state distribution $\gamma(t)$ has emerged. The density function of the resulting steady-state distribution $\gamma(t)$ is plotted in Fig. 3(b). In addition, we also measure the required additional server bandwidth due to our content availability condition in Section II-B, and compare it with the total bandwidth consumed by all the users in the system for streaming. We see that the additional bandwidth is only about 4% of the total bandwidth.

We then simulate single-channel system and study the probability that peers achieve close-to-optimal streaming capacity as the number of each peer's upstream neighbor number increases. We will compare the performance as we vary different system parameters, such as the distribution of peers' upload capacity (represented by $p = \frac{\mu}{V_{\max}}$) and the content availability at peers (represented by $q = q_{\min}$). The upload capacity of each peer is assumed to be ON-OFF, i.e., $\mathbf{P}(V_i = V_{\max}) = p$ and $\mathbf{P}(V_i = 0) = 1 - p$ for each peer i. We assume that $V_{\max} = 10$. The average upload capacity of peers is $\mu = pV_{\max}$. We vary the number of upstream neighbors per peer from $M = 10 \log N = 99$ to $M = 90 \log N = 891$, which correspond to 0.5% to 4.45% of the total number of peers N. Then, for each choice of the system parameters (p, q, ϵ) and the number of upstream neighbors per peer, we generate a single-channel P2P VoD streaming system according to our single-channel P2P control algorithms for 1000 times. In each run of the simulation, we record the smallest downloading rate among all peers and compare it with $(1 - \epsilon)\mu$. We count the number of times that this smallest downloading rate is larger than $(1 - \epsilon)\mu$ and plot the probability for that to happen. The result is shown in Fig. 3. We can observe from the simulation results that, when $p = 0.9$, $q = 0.9$, $\epsilon = 0.3$,

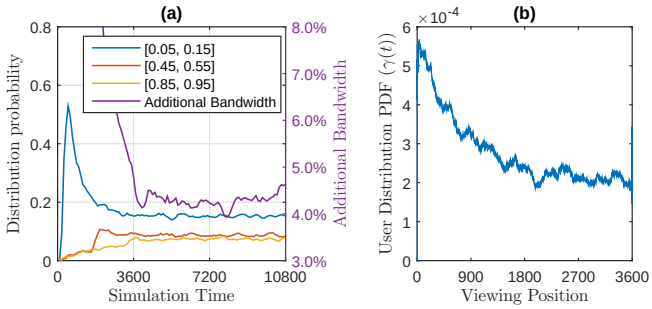


Fig. 3. Peers' viewing position approaches a steady state distribution

and when each peer selects no fewer than $10 \log N = 100$ (which corresponds to 0.5% of N) upstream neighbors, a downloading rate higher than $1 - \epsilon = 70\%$ of the average peer upload-capacity can be achieved in the entire network with probability close to 1. (We note that while $q_{\min} = 0.9$ appears to be large, it only means that each peer has 90% of the content for the range of its client set, which is of a small size $NQ = 0.05N$.) When p is reduced to 0.5 or q is reduced to 0.5, more upstream neighbors are needed to achieve the same performance. Further, under the same values of p and q, when we reduce ϵ to 0.2, more upstream neighbors are needed to achieve the same performance. These observations verify our insights following Theorem 1.

Note that our proposed algorithm has used very simple control mechanisms, such as uniformly-random neighbor-selection and uniform rate-allocation. In the following, we also simulate an adaptive algorithm to study whether more sophisticated design can further improve the system performance. Specifically, under the adaptive algorithm, each user will no longer choose its upstream neighbors from its choice set uniformly randomly. Instead, it will only choose from those peers in its choice set who still have less than qM effective downstream neighbors. As shown in Fig. 4, under the adaptive algorithm, the same probability of meeting the targeted streaming rates could be achieved with one-third less upstream neighbors (compared to the curve with the same value of p, q, ϵ). This indicates that there is potential to further improve the proposed algorithm by adding more adaptive control, which we will study in our future work.

Since our analytical results focus on large-N asymptotics, our work mainly focuses on "hot videos" that have many peers. In practice, however, one would also like to understand how large N needs to be for the proposed algorithm to be useful. In Fig. 5, we study the performance of our algorithm when the total number of peers changes from 500 to 5000. The number of neighbors for each peer is chosen as $M = 20 \log N$. We can see that when $p = 0.5, q = 0.9$ or $p = 0.9, q = 0.5$, our proposed algorithm starts to achieve close-to-optimal streaming rates with high probability once the number of users reaches around 1000, which is reasonable for "relatively hot" videos. In contrast, the performance of our algorithm degrades when $p = 0.5$, $q = 0.5$. The reason is that when $p = 0.5$ and $q = 0.5$, the choice of $\alpha = 20$ is too small and it does not satisfy our condition in Theorem 1, i.e., $\alpha \geq \frac{8d}{pq_{\min}\epsilon^2}$. Therefore, there is no performance guarantee in this case.

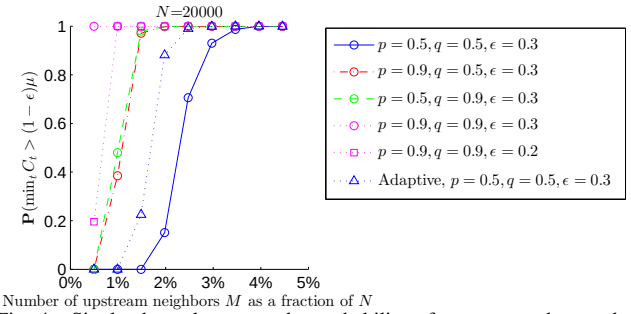


Fig. 4. Single-channel system: the probability of success as the number of upstream neighbors increases.

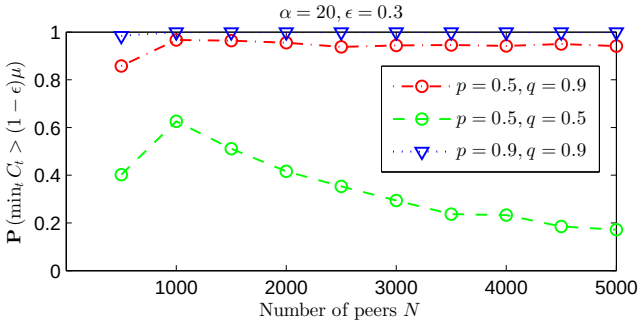


Fig. 5. Single-channel system: the probability of success as the number of peers N increases.

Next, we simulate a multi-channel P2P VoD system with 4 channels. We use the same settings as in the single-channel simulations on the distribution of peer upload capacities and the distribution of peers' downloading positions. We set $V_{\max} = 10$ and $p = 0.5$. The content availability is given by $q = q_{\min} = 0.9$. We set $N_1 = 4000$, $N_2 = 6000$, $N_3 = 3000$, $N_4 = 7000$. We choose a target streaming rate vector $\mathbf{R} = [6, 3, 7, 1]^T$, which is in $0.7\Lambda_m$ (*i.e.*, $\epsilon = 0.3$). Channels 1 and 3 are insufficient channels, and channels 2 and 4 are sufficient channels. In Fig. 6, we plot the probability that the downloading rate of a peer in channel j is greater than its target streaming rates R_j, for each of the four channels as the number of upstream neighbors per peer varies. Further, the curve with "Δ" plots the probability that all peers in all channels simultaneously sustain downloading rates greater than their corresponding target streaming rates. As we can see from Fig. 6, all channels attain with high probability their required streaming rates even with a small number of upstream neighbors. In Fig. 7, we fix $M = \alpha \log N$ with $\alpha = 50$ and plot the success probability as the total number of peers N changes. We see that the probability indeed converges to 1 when the total number of peers is larger than 20,000.

In Fig. 8, we simulate the effect of cache placement decisions based on imprecise system parameters. Specifically, the cache placement probabilities for the above system setting should be $\eta_{21} = \eta_{41} = \eta_1 = 0.4878$ and $\eta_{23} = \eta_{43} = \eta_3 = 0.5122$. (Recall from (4) that η_{kj} is the probability that a peer in sufficient channel k caches the video for insufficient channel j.) To capture the effect of imprecise estimates of system parameters, we instead simulate with a very different set of placement probabilities $[\eta'_{21}, \eta'_{23}] = [0.0697, 0.0732]$ for sufficient channel 2, and $[\eta'_{41}, \eta'_{43}] = [0.3484, 0.3659]$ for channel 4. These placement probabilities are chosen by the expression $\eta'_{kj} = (1 - \frac{R_k}{\mu(1-\epsilon)}) \cdot \eta_j$. One can verify that $\eta'_{kj} N_k = \frac{|h_k^r h_j^r|}{\sqrt{1-\epsilon}\sum_{l\in\mathcal{I}} h_l^r}$ is barely larger than H_{kj} in (5). Further, note that $\eta'_{kj}/\eta_{kj} < 1$, which implies that a smaller number of peers in channel k have the cached content to help channel j. As shown in Fig. 8, the system performance under skewed cache placement probabilities is similar to that under the original cache placement probabilities. This verifies the robustness of our strategies with respect to imprecise estimates of system parameters.

Finally, we study the cost and benefit of our content availability condition. The cost is defined as the additional amount of content that random-seeking peers need to download from the server in order to satisfy the content availability condition at a particular value of $q_{\min}$, divided by the total amount of content consumed by all streaming users. On the other hand, to calculate the benefit, we note that if a user could not get a sufficient streaming rate from its upstream neighbors, it will have to ask the server to fill in the gap. At a given value of $q_{\min}$, we then count the total amount of such content $C_G(q_{\min})$ downloaded from the server by all users to fill in the gap, i.e., $\sum_{t\in\mathcal{N}^-} \max(R - C_t, 0)$. The benefit of the content availability condition for each value of $q_{\min}$ is then calculated as $C_G(0) - C_G(q_{\min})$, again normalized by the total amount of content consumed by all streaming users. Thus, this benefit value reflects the reduction in server capacity by non-random-seeking users, compared to the case without the content availability condition (i.e., $q_{\min} = 0$). We simulate a network with 10000 peers, streaming a one-hour video. We create a similar setting as the one that we described in Section II: there is an interval (e.g., the opening of the video) for about $1/10$ of the video length that some fraction of the users wish to skip. After skipping, these peers download additional content to satisfy our content availability condition. We plot in Fig. 9 the cost and benefit, at different values of $q_{\min}$ and with different skipping probabilities. We first observe that, when $q_{\min}$ is small (less than 0.4), the benefit outweighs the cost. In other words, our content availability condition reduces the total download capacity from the server when $q_{\min}$ is relatively small.[3] However, as $q_{\min}$ increases, the cost eventually becomes larger than the benefit. Further, we observe that, when the skipping probability increases, both the benefit and cost increases. The cost increases because we have more random-seeking peers that need to download the additional content. The benefit increases because $C_G(0)$ increases, i.e., when the skipping probability increases, without the content availability condition the peers that did not jump would have even less chance to find an effective upstream neighbor. Interestingly, the ratio between the cost and the benefit remains roughly the same for different skipping probabilities, and the cross-over value of $q_{\min}$ for the cost to be approximately equal

[3]We note that the above comparison is somewhat conservative because our definition of the benefit accounts for the streaming rate from upstream neighbors, while our definition of the cost does not. If the random-seeking peers can also utilize any remaining capacity from the upstreaming neighbors to satisfy the content availability condition, the cost will be further reduced, and thus the value of $q_{\min}$ such that the benefit outweighs the cost will be even larger.

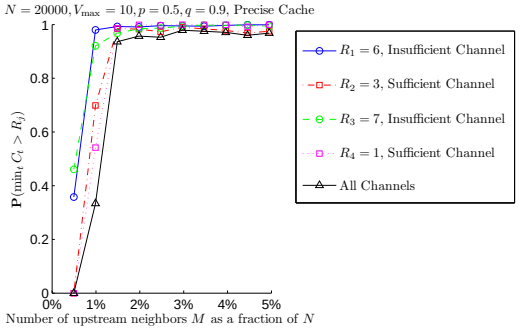


Fig. 6. Multi-channel system: the probability of success as the number of upstream neighbors increases.

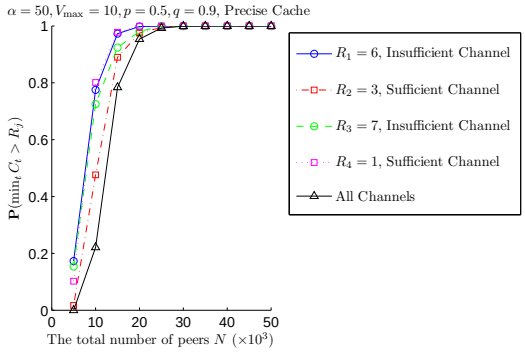


Fig. 7. Multi-channel system with precise cache placement: the probability success as the total number of peers increases. The number of upstream neighbors is chosen as $M = \alpha \log N$ with $\alpha = 50$.

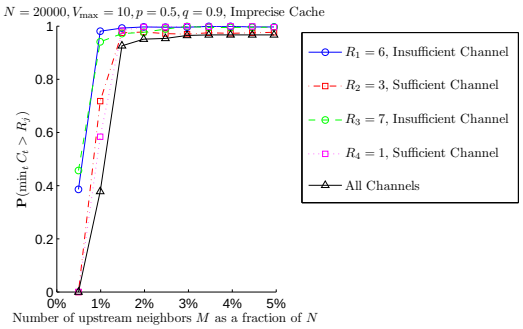


Fig. 8. Multi-channel system with imprecise cache placement: when the cache-placement algorithm is based on imprecise estimates of system parameters, the probability of success as the number of upstream neighbors increases.

to the benefit also does not vary significantly.

V. Conclusion

In this paper we provide a rigorous analytical study on the performance of large-scale P2P VoD systems with sparse connectivity and simple, robust, and decentralized control. For both single-channel and multi-channel systems, we provide easy-to-implement P2P control algorithms and show that the system can achieve close-to-optimal streaming capacity with probability approaching 1, as the total number of peers N increases. Under our control algorithms, each peer is only

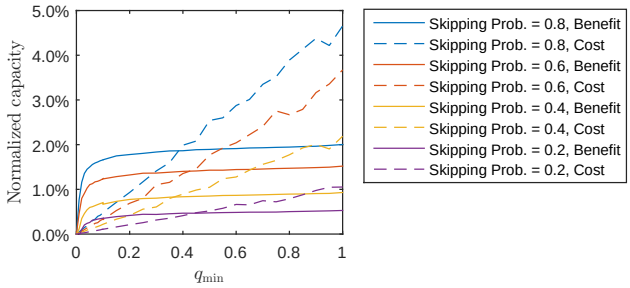


Fig. 9. The cost vs. benefit of the content availability condition

assigned $\Theta(\log N)$ upstream neighbors, with which it exchanges content availability information. Most parts of the control algorithms are decentralized. These algorithms incur low control overhead and are easy-to-implement in practice. Our analytical studies provide easy-to-verify conditions for such close-to-optimal streaming to hold, which shed important insights to guide the design of improved P2P streaming protocols. There are a number of interesting directions for further work. First, it would be interesting to study whether the required number of per-peer neighbors can be further reduced, possibly by using more sophisticated peer-selection and rate-allocation algorithms than those studied in this paper. The challenge would be how to improve the system performance while retaining the simplicity and decentralized properties. Second, in our multi-channel setting, the number of videos is assumed to be fixed as N increases. A different setting when the number of videos also approaches infinity is of significant practical interest. It remains a challenging question whether or not similar simple algorithms could achieve good performance under such a many-video case.

References

[1] C. Zhao, J. Zhao, X. Lin, and C. Wu, "Capacity of P2P On-Demand Streaming with Sparse Connectivity and Simple Decentralized Control," in *Proc. of IEEE INFOCOM*, 2013.

[2] Z. Liu, C. Wu, B. Li, and S. Zhao, "UUSee: Large-scale Operational On-Demand Streaming with Random Network Coding," in *Proc. of IEEE INFOCOM*, Mar 2010.

[3] C. Zhao, X. Lin, and C. Wu, "The Streaming Capacity of Sparsely-Connected P2P Systems with Distributed Control," in *Proc. of IEEE INFOCOM*, Apr. 2011, pp. 1449 –1457.

[4] R. Kumar, Y. Liu, and K. Ross, "Stochastic Fluid Theory for P2P Streaming Systems," in *Proc. of IEEE INFOCOM*, May 2007, pp. 919 –927.

[5] L. Massoulié, A. Twigg, C. Gkantsidis, and P. Rodriguez, "Randomized Decentralized Broadcasting Algorithms," in *Proc. of IEEE INFOCOM*, 2007, pp. 1073–1081.

[6] C. Feng and B. Li, "On Large-scale Peer-to-Peer Streaming Systems with Network Coding," in *Proc. of ACM Multimedia*, 2008.

[7] T. Bonald, L. Massoulié, F. Mathieu, D. Perino, and A. Twigg, "Epidemic Live Streaming: Optimal Performance Trade-offs," in *Proc. of ACM SIGMETRICS*, 2008, pp. 325–336.

[8] S. Liu, R. Zhang-Shen, W. Jiang, J. Rexford, and M. Chiang, "Performance Bounds for Peer-Assisted Live Streaming," in *Proc. of ACM SIGMETRICS*, 2008, pp. 313–324.

[9] S. Sengupta, S. Liu, M. Chen, M. Chiang, J. Li, and P. A. Chou, "Streaming Capacity in Peer-to-Peer Networks with Topology Constraints," *submitted to IEEE Transaction on Information Theory*, 2009.

[10] S. Liu, M. Chen, S. Sengputa, M. Chiang, J. Li, and P. A. Chou, "P2P Streaming Capacity under Node Degree Bound," in *Proc. of IEEE ICDCS*, Jun. 2010.

[11] D. Wu, C. Liang, Y. Liu, and K. W. Ross, "View-Upload Decoupling: A Redesign of Multi-Channel P2P Video Systems," in *Proc. of IEEE INFOCOM*, Apr 2009.

[12] Y. Huang, T. Z. Fu, D.-M. Chiu, J. C. Lui, and C. Huang, "Challenges, Design and Analysis of a Large-scale P2P-VoD System," *SIGCOMM Comput. Commun. Rev.*, vol. 38, pp. 375–388, Aug. 2008.

[13] K. Wang and C. Lin, "Insight into the P2P-VoD System: Performance Modeling and Analysis," in *Proc. of IEEE ICCCN*, Aug. 2009, pp. 1 –6.

[14] B. Tan and L. Massoulié, "Optimal Content Placement for Peer-to-Peer Video-on-Demand Systems," in *Proc. of IEEE INFOCOM*, Apr 2011.

[15] Y. Zhou, T. Fu, and D. M. Chiu, "On Replication Algorithm in P2P VoD," *Networking, IEEE/ACM Transactions on*, vol. 21, no. 1, pp. 233–243, Feb 2013.

[16] J. Wu and B. Li, "Keep Cache Replacement Simple in Peer-Assisted VoD Systems," in *Proc. of IEEE INFOCOM*, Apr 2009.

[17] N. Parvez, C. Williamson, A. Mahanti, and N. Carlsson, "Analysis of Bittorrent-like Protocols for On-Demand Stored Media Streaming," *SIGMETRICS Perform. Eval. Rev.*, vol. 36, pp. 301–312, Jun. 2008.

[18] N. Carlsson and D. L. Eager, "Peer-assisted On-Demand Streaming of Stored Media Using BitTorrent-like Protocols," in *Proc. of the 6th international IFIP-TC6 conference on Ad Hoc and sensor networks, wireless networks, next generation internet*, 2007, pp. 570–581.

[19] W. J. Wu and J. C. S. Lui, "Exploring the Optimal Replication Strategy in P2P-VoD Systems: Characterization and Evaluation," in *Proc. of IEEE INFOCOM*, Apr 2011.

[20] D. Ciullo, V. Martina, M. Garetto, E. Leonardi, and G. Torrisi, "Asymptotic Properties of Sequential Streaming Leveraging Users' Cooperation," *Information Theory, IEEE Transactions on*, vol. 59, no. 12, pp. 8386–8401, Dec 2013.

[21] D. Ciullo, V. Martina, M. Garetto, and E. Leonardi, "How much can large-scale Video-on-Demand benefit from users' cooperation?" in *Proc. of IEEE INFOCOM*, Apr 2013, pp. 2724–2732.

[22] D. Ciullo, V. Martina, M. Garetto, E. Leonardi, and G. Torrisi, "Peer-Assisted VoD Systems: An Efficient Modeling Framework," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 25, no. 7, pp. 1852–1863, July 2014.

[23] J. Wang, C. Huang, and J. Li, "On ISP-Friendly Rate Allocation for Peer-Assisted VoD," in *Proc. of ACM Multimedia*, 2008.

[24] C. Liang, Y. Guo, and Y. Liu, "Is Random Scheduling Sufficient in P2P Video Streaming?" in *Proc. of IEEE ICDCS*, Jun. 2008, pp. 53 –60.

[25] C. Zhao, J. Zhao, X. Lin, and C. Wu, "Capacity of P2P On-Demand Streaming with Sparse Connectivity and Simple Decentralized Control," Tech. Rep., 2012, also available at https://engineering.purdue.edu/%7elinx/papers.html.

APPENDIX A
IMPLEMENTATION OF CONTENT AVAILABILITY CONDITION

The content availability condition we introduced in Section II-B can be implemented as follows. Choose $q'_{\min}$ such that $(1-e^{-q'_{\min}})/2 = q_{\min}$. Suppose that a peer (denoted by t) randomly seeks to position $t = t_0$ first. It will first download a fraction of the content from the range that may be requested by the peers in its client set. More specifically, let $\psi(t_0)$ be the downloading position of the youngest peers in this peer's client set $\bar{\mathcal{D}}_{t_0}$. This peer then selects K intervals within $[\psi(t_0), t_0]$, each of which has a length of $q'_t(t_0-\psi(t_0))/K$, where $q'_t \geq q'_{\min} > 0$ satisfies $1-\left(1-\frac{q'_t}{K}\right)^K = q_t$. These K intervals are selected independently and uniformly randomly. At this point, it is easy to see that the above content-availability condition holds: for any peer s in $[\psi(t_0), t_0]$, the probability that peer $t = t_0$ has the required content for peer s is equal to the probability that peer s is in at least one of the K intervals, which is calculated as $1-\left(1-\frac{q'_t}{K}\right)^K = q_t$. For sufficiently large K, we will have $q_t \geq (1-e^{-q'_{\min}})/2 = q_{\min}$. Next, as peer t continues to watch the video, it downloads the content from t_0 to its current downloading position $t > t_0$. In order to meet the content availability condition for all peers in the client set $\bar{\mathcal{D}}_t$, as long as $\bar{\mathcal{D}}_t$ contains at least one peer s whose downloading position is smaller than t_0, then for all other peers in $\bar{\mathcal{D}}_t \cap (t_0, t)$, peer t is only willing to serve it with probability q_t, independently of other peers. This restriction will continue until all peers $s \in \bar{\mathcal{D}}_t$ advance past t_0. Then, peer t can serve all of its downstream neighbors (equivalently, $q_t = 1$).

APPENDIX B
DETAIL PROOF OF THEOREM 1

With Lemma 2, 3, 4 and 5, it is trivial to show (3) holds for our alternative policy: "random sequential-range" choice set selection strategy. To go back to our original "random sequential" choice set selection strategy and prove Theorem 1, we need the following lemma:

Lemma 7. *Given any $\epsilon > 0$ and $d > 1$, let*

$$\mathcal{W} = \{|\psi(t) - \psi'(t)| < \epsilon, \text{ for all } t \in \mathcal{N}\}.$$

Then, $\mathbf{P}(\mathcal{W}) \geq 1 - O\left(\frac{1}{N^{2d}}\right)$.

Proof. For any peer t such that $\psi'(t) = 0$, i.e., $\int_0^t \gamma(\tau)\mathrm{d}\tau \leq NQ$, the event $|\psi(t) - \psi'(t)| \geq \epsilon$ implies that $\psi(t) \geq \epsilon$. Thus, the number of peers in $[\epsilon, t)$ must be greater than NQ. However,

$$\int_\epsilon^t \gamma(\tau)\mathrm{d}\tau = \int_0^t \gamma(\tau)\mathrm{d}\tau - \int_0^\epsilon \gamma(\tau)\mathrm{d}\tau \leq NQ - \epsilon\gamma_{\min}.$$

Note that the number of peers in $[\epsilon, t)$ is a binomial random variable with sample size N and success probability $\int_\epsilon^t \gamma(\tau)\mathrm{d}\tau$. It is not difficulty to see that (refer to [25] for details)

$$\begin{aligned}
&\mathbf{P}\left(|\psi(t) - \psi'(t)| \geq \epsilon\right) \\
\leq &\mathbf{P}\left(\text{the number of peers in } [\epsilon, t) \text{ is greater than } NQ\right) \\
\leq &\exp\left(-\frac{(NQ - \epsilon\gamma_{\min} - NQ)^2}{NQ - \epsilon\gamma_{\min}}\right) \\
\leq &O\left(\frac{1}{N^{2d+1}}\right). \qquad (6)
\end{aligned}$$

Further, for any peer t such that $\psi'(t) > 0$, $|\psi(t) - \psi'(t)| \geq \epsilon$ implies that $\psi(t) \leq \psi'(t) - \epsilon$ or $\psi(t) \geq \psi'(t) + \epsilon$. Consider the case when $\psi(t) \leq \psi'(t) - \epsilon$. Such an event implies that the number of peers in $[\psi'(t) - \epsilon, t)$ is less than NQ. However, the number of peers in $[\psi'(t) - \epsilon, t)$ is a binomial random variable with sample size N and success probability

$$\int_{\psi'(t)-\epsilon}^t \gamma(\tau)\mathrm{d}\tau \geq \int_{\psi'(t)}^t \gamma(\tau)\mathrm{d} + \epsilon\gamma_{\min} = NQ + \epsilon\gamma_{\min}.$$

It is not difficulty to see that (refer to [25] for details)

$$\begin{aligned}
&\mathbf{P}\left(\psi(t) \leq \psi'(t) - \epsilon\right) \\
\leq &\mathbf{P}\left(\text{the number of peers in } [\psi'(t) - \epsilon, t) \text{ is less than } NQ\right) \\
\leq &\exp\left(-\frac{(NQ + \epsilon\gamma_{\min} - NQ)^2}{NQ + \epsilon\gamma_{\min}}\right) \\
\leq &O\left(\frac{1}{N^{2d+1}}\right). \qquad (7)
\end{aligned}$$

Similarly, one can show that

$$\mathbf{P}\left(\psi(t) \geq \psi'(t) + \epsilon\right) \leq O\left(\frac{1}{N^{2d+1}}\right). \qquad (8)$$

Finally, combining (6), (7) and (8), and taking the union bound, we have

$$\begin{aligned}\mathbf{P}(\mathcal{W}) =&\mathbf{P}\left(\bigcap_{t\in\mathcal{N}}|\psi(t)-\psi'(t)|<\epsilon\right)\\ =&1-\mathbf{P}\left(\bigcup_{t\in\mathcal{N}}|\psi(t)-\psi'(t)|\geq\epsilon\right)\\ \geq&1-N\mathbf{P}\left(|\psi(t)-\psi'(t)|\geq\epsilon\right)\geq 1-O\left(\frac{1}{N^{2d}}\right).\end{aligned}$$

□

Next, fix any peer t. Recall that for any peer $i\in\mathcal{U}_t$, $\hat{\mathcal{D}}_i$ consists of two parts: (a) all the n_1 peers in $\Gamma_i\cap[\psi(t_i),t_{i_0})$, and (b) for the n_2 peers in $[t_{i_0},t_i)$, each of them is in $\hat{\mathcal{D}}_i$ with probability q_i independent of others. For any $\epsilon>0$, given $\mathcal{A}$ in Lemma 3 and $\mathcal{W}$ in Lemma 7, we have

$$\begin{aligned}n_1\leq&N\left(\int_{\Gamma_i\cap[\psi(t_i),t_{i_0}]}\gamma(\tau)\mathrm{d}\tau+\epsilon\right)(1+\epsilon)\\ \leq&N\left(\int_{\Gamma_i\cap[\psi'(t_i)-\epsilon,t_{i_0}]}\gamma(\tau)\mathrm{d}\tau+\epsilon\right)(1+\epsilon)\\ \leq&N\left(\int_{\Gamma_i\cap[\psi'(t_i),t_{i_0}]}\gamma(\tau)\mathrm{d}\tau+\epsilon\gamma_{\max}\right)(1+\epsilon)\end{aligned}$$

and

$$n_2\leq N\left(\int_{[t_{i_0},t_i]}\gamma(\tau)\mathrm{d}\tau+\epsilon\right)(1+\epsilon).$$

Since $\epsilon>0$ can be arbitrarily, replacing ϵ by $\frac{\epsilon}{\gamma_{\max}}$ we have

$$\begin{aligned}n_1&\leq N\left(\textstyle\int_{\Gamma_i\cap[\psi'(t_i),t_{i_0}]}\gamma(\tau)\mathrm{d}\tau+\epsilon\right)(1+\epsilon)=n_1^+,\\ n_2&\leq N\left(\textstyle\int_{[t_{i_0},t_i]}\gamma(\tau)\mathrm{d}\tau+\epsilon\right)(1+\epsilon)=n_2^+.\end{aligned}$$

Now, we can construct an alternative system in the same way as we did in Section II-C. We will have $\underline{D}_i\leq\underline{D}_i^+$, $i=1,2,...,M$, where $\underline{D}_i^+$'s satisfy Lemma 4 and Lemma 5, with the conditioning event $\mathcal{A}$ replaced by $\mathcal{A}\cap\mathcal{W}$. Let $\mathbf{P}_{\mathcal{AW}}(\cdot)$ and $\mathbf{E}_{\mathcal{AW}}(\cdot)$ denote the probability and the expectation conditioned on $\mathcal{A}\cap\mathcal{W}$. We thus have, for any $r_1,r_2,...,r_M\geq 0$,

$$\begin{aligned}&\mathbf{E}_{\mathcal{AW}}\left[\exp\left(-\sum_{i=1}^M\frac{r_i}{\underline{D}_i}\right)\middle|\mathbf{I}\right]\\ \leq&\mathbf{E}_{\mathcal{AW}}\left[\exp\left(-\sum_{i=1}^M\frac{r_i}{\underline{D}_i^+}\right)\middle|\mathbf{I}\right]\\ \leq&\prod_{i=1}^M\mathbf{E}_{\mathcal{AW}}\left[\exp\left(-\frac{r_i}{\underline{D}_i^+}\right)\middle|\mathbf{I}\right].\end{aligned}$$

Now applying Lemma 2 and taking $\rho=(1+\epsilon)$, we have

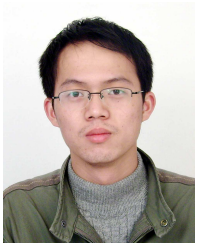
$$\mathbf{P}_{\mathcal{AW}}\left(\sum_{i=1}^M\frac{V_iI_i}{\underline{D}_i}\leq\frac{(1-\epsilon)\mu}{1+\epsilon}\right)\leq O\left(\frac{1}{N^{2d}}\right).$$

Note that $C_t=\sum_{i=1}^M C_{it}=\sum_{i=1}^M\frac{V_iI_i}{\underline{D}_i}$ and ϵ is arbitrary. Hence, for any $\epsilon>0$, $\mathbf{P}_{\mathcal{AW}}\left(C_t\leq(1-\epsilon)\mu\right)\leq O\left(\frac{1}{N^{2d}}\right)$. Consequently,

$$\begin{aligned}&\mathbf{P}\left(C_t\leq(1-\epsilon)\mu\right)\\ \leq&\mathbf{P}_{\mathcal{AW}}\left(C_t\leq(1-\epsilon)\mu\right)+\mathbf{P}(\mathcal{A})+\mathbf{P}(\mathcal{W})\leq O\left(\frac{1}{N^{2d}}\right).\end{aligned}$$

The result of Theorem 1 thus follows by taking the union bound over all peers.

Can Zhao received his B.S. degree in Electrical Engineering from Tsinghua University, Beijing, China in 2007, and his Ph.D. degree in Electrical and Computer Engineering from Purdue University, West Lafayette, Indiana, in 2012. His research interests are mathematical modeling and evaluation of communication networks, including ad hoc networks and peer-to-peer networks.



Jian Zhao received his B.Eng. degree in 2009 from Department of Electronic Engineering and Information Science, University of Science and Technology of China, China, and his Ph.D. degree in 2014 from the Department of Computer Science, the University of Hong Kong, China. His research interests include peer-to-peer video streaming and cloud computing. He is a member of IEEE.

Xiaojun Lin (S'02 / M'05) received his B.S. from Zhongshan University, Guangzhou, China, in 1994, and his M.S. and Ph.D. degrees from Purdue University, West Lafayette, Indiana, in 2000 and 2005, respectively. He is currently an Associate Professor of Electrical and Computer Engineering at Purdue University.

Dr. Lin's research interests are in the analysis, control and optimization of wireless and wireline communication networks. He received the IEEE INFOCOM 2008 best paper award and 2005 best paper of the year award from *Journal of Communications and Networks*. His paper was also one of two runner-up papers for the best-paper award at IEEE INFOCOM 2005. He received the NSF CAREER award in 2007. He was the Workshop co-chair for IEEE GLOBECOM 2007, the Panel co-chair for WICON 2008, the TPC co-chair for ACM MobiHoc 2009, and the Mini-Conference co-chair for IEEE INFOCOM 2012. He is currently serving as an Area Editor for (Elsevier) Computer Networks journal, and has served as a Guest Editor for (Elsevier) Ad Hoc Networks journal.

Chuan Wu received her B.E. and M.E. 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 cloud computing, peer-to-peer networks and online/mobile social network. She is a member of IEEE and ACM.