On the Performance of Content Delivery under Competition in a Stochastic Unstructured Peer-to-Peer Network

Yuh-Ming Chiu and Do Young Eun
Department of Electrical and Computer Engineering
North Carolina State University, Raleigh, NC
{ychiu, dyeun}@ncsu.edu

Abstract—Peer-to-peer (P2P) network is widely used for transferring large files nowadays. Measurement results show that most downloading peers are patient as the average download session is usually very long. It is sometimes even longer than downloading from a dedicated server using a modem. Existing results in the literature indicate that the stochastic fluctuation and the heterogeneity in the service capacity of each peer are two of the major reasons that make the average download time far longer than expected. In those studies, it has been often assumed that there is only one downloading peer in the network, ignoring the interaction and competition among peers.

In this paper, we investigate the impact of the interaction and competition among peers on downloading performance under stochastic, heterogeneous, unstructured P2P settings, thereby greatly extending the existing results on stochastic P2P networks made only under a single downloading peer in the network. To analyze the average download time in a P2P network with multiple competing downloading peers, we first introduce the notion of system utilization tailored to a P2P network. We investigate the relationship between the average download time, system utilization and the level of competition among downloading peers in a stochastic P2P network. We then derive an achievable lower bound on the average download time and propose algorithms to give the peers the minimum average download time. Our result can much improve the download performance compared to earlier results in the literature. The performance of the different algorithms is compared under NS-2 simulations. Our results also provide theoretical explanation to the inconsistency of performance improvement by using parallel connections (parallel connections sometimes do not outperform a single connection) observed in some measurement studies.

Index Terms—Computer network performance, P2P networks

1 Introduction

Peer-to-peer (P2P) technologies, such as BitTorrent [2], Gnutella [3], or Kaaza [4], have been widely used for file transfer over the Internet. In those applications, file download time is one of the most important performance metrics. Theoretically, a P2P network can make its users download faster compared to a traditional client-server network because a P2P network is inherently scalable. Each node in a P2P network can act both as a server and a client *simultaneously*. As a result, the aggregate system service capacity increases with the number of downloading nodes (demand) in a P2P network [5]. It is widely believed that only the physical access bandwidth of each downloading peer can limit the download performance.

However, measurement studies show rather counter-intuitive results. It is shown in [6], [7] that downloading a 100MB file from a Gnutella or Kaaza network generally takes hours often up to a week. Considering the fact that earlier P2P studies predicted that the download performance is only limited by each peer's physical access bandwidth and most people nowadays have broadband access, the typical download time for a 100MB file should be less than an hour if the prediction is correct. The gap between prediction and measured performance motivated us to investigate more about the file download process in a P2P system.

Most people believe that the existence of free-riders is the main reason that degrades the performance of a P2P network. Accordingly, many incentive algorithms have been developed so as to encourage peers to contribute to the network [8], [9], [10], [11], [12], [13]. However, even if all peers in the network are altruistic, we are still far away from enjoying the performance predicted by [5], [14]. Results in [15] suggest that *both* the stochastic temporal fluctuation and the heterogeneity in service capacity of each peer can make the average download time significantly longer than expected, even when all peers in the network are willing to share. Although some other earlier studies have also noticed the impact of stochastic fluctuation and the heterogeneity in service capacity of each peer, those studies often have more limited viewpoints. For example, the authors in [16] consider only the stochastic fluctuation in service capacity of each peer but they do not consider the network heterogeneity. On the other hand, the authors in [17],

A preliminary version of this work appeared in [1].
This work was supported in part by NSF CAREER Award CNS-0545893

[18] try to develop an optimal peer selection method that exploits the heterogeneity of the network but do not consider the temporal fluctuation in service capacity of each peer.

More important, all the results in [15], [16], [17], [18] have been established under the assumption that *there is only one downloading peer in the network*. This is critical, since in a real P2P network there will be multiple peers uploading and downloading at the same time and a peer's service capacity will be shared by its competing peers. In other words, the downloading peers will have to compete for the limited resource each single source peer can offer. In this setting, the download performance is determined not only by the stochastic fluctuation and heterogeneity in service capacity that each peer offers; how each peer makes its peer selection choice under such a competitive environment is also very important.

In this paper, we jointly consider all the factors presented in [15], [16], [17], [18] *and* the impact of interaction/competition among the downloading peers. In order to capture this impact of interactions among the downloading peers, we first introduce the notion of system utilization – defined as the average fraction of the aggregated system capacity that is *actually* consumed by all the downloading peers, which is heavily dependent upon the peer selection decision. With the introduction of system utilization, we show how the average "share" of system capacity a downloading peer receives is not always equal to the aggregated capacity of the source peers divided by the number of downloading peers, even if each source peer distributes its capacity evenly among its connected downloading peers. Further, we extend the results in [15] that the average file download time is often greater than the file size divided by the average received capacity of each downloading peer in our much more complex network setting. We derive an *achievable* lower bound on the average download time based on the average network capacity, the system utilization, and the level of competition in the network.

It was shown in [15] that switching a single connection "periodically" among possible source peers "uniformly at random" can reduce the average download time to some extent. In this paper, in a network with *multiple downloading peers*, we show that the algorithm in [15] is far from being optimal, but can still be used as the first step toward developing an optimal peer selection strategy. We modify the notion of periodic switching and start by letting each downloading peer generate *a random number of parallel connections periodically* instead of having just one single connection all the time. Each downloading peer chooses its source peers according to some probability. We can then formulate the probability assignment problem as a convex optimization problem and compute the optimal connection probabilities in a centralized fashion. Our solution is applicable to much wider ranges of network scenarios with stochastic fluctuation and heterogeneity under peer competitions, to which none of the approaches in [15], [16], [17], [18] applies.

While the aforementioned centralized algorithm to solve for the optimal peer selection probability is impractical in a fully distributed P2P network, the algorithm will serve its purpose as a benchmark with optimal performance. We then propose a heuristic distributed algorithm that only depends upon the network performance metric *observed by each downloading peer*. Through realistic network simulations using NS-2 [19], we compare the performance of different peer selection strategies. We use the performance of periodic uniform peer selection derived from [15] as the baseline for performance comparison. Our NS-2 simulation results show that both the optimal centralized algorithm and the proposed distributed algorithm outperforms the periodic uniform peer selection in all cases. Our centralize algorithm can even reduce the average download time by 50% compared to periodic uniform peer selection. In addition, our results in this paper also provide theoretical explanation for the observations in [20], [21] that using parallel connections does not always give users better performance than using a single connection. We present conditions under which parallel connections is helpful and provide a guideline for determining a suitable number of parallel connections.

The paper is organized as follows. In Section 2, we present our system model. In Section 3, we introduce a performance metric – system utilization, which is determined by the interaction between downloading peers, and derive the relationship between the average download time and the system utilization. In Section 4, we present a centralized optimal peer selection algorithm that maximizes the system utilization for benchmark purpose and a simple distributed algorithms. In Section 5, We present the simulation results and we conclude in Section 6.

2 System model

In a P2P network, all peers can be categorized into two groups for a given file of interest: the peers who are uploading data to other peers (source peers) and those who are downloading from others (downloading peers). We denote the set of source peers and the set of downloading peers by $\mathcal{S}$ and $\mathcal{D}$, respectively. Each peer can either be in one of the two sets, $\mathcal{S}$ or $\mathcal{D}$, or in both. For example, a free-rider is in $\mathcal{D}$ only and a peer who contributes to the network without downloading anything from other peers is in $\mathcal{S}$ only. In most cases, a peer is counted in both sets $\mathcal{D}$ and $\mathcal{S}$. We use indices i and j to represent the peers in sets $\mathcal{D}$ and $\mathcal{S}$ throughout the paper, respectively.

We model the network as a discrete time system with the length of a unit time-slot set to Δ. Considering a P2P network as a discrete time system is widely used in performance analysis of P2P networks in the literature [5], [22]. Thus the duration of each slot is $[(t-1)\Delta, t\Delta), t \in \mathbb{N}$. For analytical simplicity, Δ is

normalized to 1 and each time slot can then be simply indexed by t. There are two major data dissemination models, push-based and pull-based, often used in the literature to describe P2P systems. In a push-based system, the source peers control when and what to send to the downloading peers [23], [24]. The downloading peers can make as many data or connection requests as they want and then wait passively for the source peers to service them. On the other hand, the downloading peers in a pull-based system have the power to decide when and what to download from the source peers [25]. A piece of data is transferred immediately from the source peers to the downloading peers upon requests. Since each downloading peer is interested in improving its own downloading performance, it is intuitive to give the downloading peers more control in its own download process. Hence we view our network as a pull-based system. Since the downloading peers can select its source peers actively, we denote the subset $S_i(t) \subseteq \mathcal{S}$ be the set of source peers that peer i connects to at time slot t.

Note that peers i and j are actively connected if and only if the connection between i and j is carrying data traffic in our definition. For each connection, it is well known that the available bandwidth fluctuates over time [26], [27] due to the workloads of both end points or the network congestion status. Recent studies have shown that most peers in a P2P network utilize broadband connections. Since either cable or DSL lines offer asymmetrical bandwidth and the upstream bandwidth is usually much smaller than downstream bandwidth, it is reasonable to assume that the bottleneck lies in the service capacity of the source peers. Let $C_j(t)$ denote the *total service capacity* a source j can offer at time slot t. We assume that all downloading peers connecting to the same source peer will share the source's service capacity equally, i.e., if there are currently M downloading peers connecting to source j, each downloading peer will get a data rate of $C_j(t)/M$ during time slot t.

Define an indicator function

$$I_{ij}(t) = \begin{cases} 1, & j \in \mathcal{S}_i(t) \\ 0, & \text{otherwise} \end{cases}$$

showing whether downloading peer i connects to source j at time t. Then, $\sum_{i\in\mathcal{D}} I_{ij}(t)$ is the number of downloading peers that are connected to source j at time t. The data rate that the downloading peer i receives from source j at time t becomes

$$R_{ij}(t) = \frac{I_{ij}(t)}{\sum_{i\in\mathcal{D}} I_{ij}(t)} C_j(t).$$

The aggregated capacity, $R_i(t)$, that a downloading peer i receives from the entire network during time slot t will then be

$$R_i(t) \triangleq \sum_{j\in\mathcal{S}_i(t)} R_{ij}(t) = \sum_{j\in\mathcal{S}} \frac{I_{ij}(t)}{\sum_{i\in\mathcal{D}} I_{ij}(t)} C_j(t), \quad (1)$$

The second equality in (1) comes from the fact that $I_{ij}(t) = 0$ for $j \notin \mathcal{S}_i(t)$, i.e., sources not connected to downloading peer i will not contribute to $R_i(t)$.

To simplify the analysis, we first impose the following assumptions.

(A1) Different downloading peers make their own choices independent of other peers.

(A2) There is no restriction on the number of connections a source peer can have, i.e., all connection attempts of a downloading peer always succeed. However, we limit the average number of parallel connections a downloading peer can open to L.

(A3) A downloading peer i selects each source peer j with some probability, p_{ij}, i.e. the connection decision $I_{ij}(t)$ is generated from p_{ij}. The connection probability p_{ij} and the fluctuation in $C_j(t)$ of a source peer j are independent.

We have (A1) because it is unlikely that each downloading peer will broadcast its own peer selection strategy to its neighbors unless the set of downloading peers conspire together.

We have (A2) to reduce capacity over-saturation of the source peers. Capacity over-saturation occurs when a source peer has too many active connections simultaneously. The capacity offered over each connection becomes extremely low. The performance bottleneck in terms of receive capacity (from a downloading peer's point of view) is the capacity over-saturated source peers. It is intuitive to limit the number of parallel connections each source peer can have to eliminate capacity over-saturation. It can be done in two ways. We can either put such limitation on the source peers or the downloading peers. To see how limiting the number of parallel connection each downloading peer can have helps to prevent capacity over-saturation from happening, let's consider the following simple example. Suppose that there are $N = 100$ source peers and $M = 200$ downloading peers. Each downloading peer is limited to have one connection. Each downloading peer selects its source peer uniformly at random. The number of parallel connections each source peer will have is then a binomial random variable with $n = M, p = 1/N = 0.01$. Therefore, each source peer will have $M/N = 2$ parallel connections on average with variance of 1.98. In other words, it is very unlikely for a source peer to have 5 or more parallel connections. In the above case, we demonstrate clearly that we can prevent capacity over-saturation by limiting the number of parallel connections a downloading peer can have.

We have (A3) because selecting source peers using probability is in general better than selecting source peers deterministically based on some system metrics. We define deterministic peer selection as a downloading peer making its connection decisions based on some measured network statistics. The decision is always connect/disconnect if the measured statistics is greater/smaller than some threshold. To demonstrate the idea of deterministic peer selection, let's consider the following example. Suppose that there are 3 source peers and a downloading peer. The downloading peer

can have two parallel connections simultaneously. The capacity of source 1, 2, and 3 are denoted by $C_1(t)$, $C_2(t)$, and $C_3(t)$, respectively. The table below shows the capacity fluctuation of each peer. We can either make our connection decision base on the average capacity (assume it is known) of each source peer or on the received capacity in the previous time slot.

	t=1	2	3	4	5	6	...
$C_1(t)$	**4**	**1**	4	**1**	4	**1**	...
$C_2(t)$	**3**	3	**3**	**3**	**3**	**3**	...
$C_3(t)$	1	**2**	**1**	2	**1**	2	...

If we use the received capacity in the previous slot as the decision threshold. One possible strategy is to disconnect from the source peer that offers the lowest capacity and then make new connections to the source peer that is not utilized in the previous slot. The boldface numbers in the above table represents the capacity a downloading peer receives from the its two connections over time under such strategy. In this case, the average received capacity is less than 4. However, if we connect to the two source peers that offer higher average capacity (peers 1 and 2). The average received capacity is 4.

In the above example, we can see that selecting peers according to their average capacity is better. It is possible that the approach of making decisions utilizing the most recent measurement is better in some cases but we lack the systematic analysis of selecting peers based on the most recent measurements in the literature. Hence, the most common approach adopted in the literature is selecting source peers based on their average capacity. However, making deterministic peer selection based on the average capacity of the source peers is not as good as using a uniform random variable to choose a source peer as shown in [15]. Hence we adopt the notion of using probability to choose source peers.

3 Download Time Analysis

3.1 System Utilization

We first define the system utilization of a P2P network because it is important in determining the average download time. Let the service capacity of source $j \in \mathcal{S}$ be stationary with $\mathbb{E}\{C_j(t)\} = c_j$. We have the following:

Definition 1. *The system utilization is*

$$\rho \triangleq \frac{\mathbb{E}\left\{\sum_{i\in\mathcal{D}} R_i(t)\right\}}{\mathbb{E}\left\{\sum_{j\in\mathcal{S}} C_j(t)\right\}} = \frac{\sum_{i\in\mathcal{D}}\sum_{j\in\mathcal{S}}\mathbb{E}\left\{R_{ij}(t)\right\}}{\sum_{j\in\mathcal{S}} c_j} \quad (2)$$

which is the ratio between the aggregated average service capacity of the entire network and the aggregated average capacity actually consumed by all the downloading peers.

In any network, from an administrator's point of view, the system would perform the best if the "utilization" of the system is maximized. In other words, when the system is under-utilized, i.e. some source capacity being wasted, downloading peers can increase their performance by utilizing those unused resources and thus reduce their download time.

To make our analysis more tractable, let's assume that each downloading peer has the same probability of connecting to a source j, i.e.,

$$\mathbb{P}\{I_{ij}(t) = 1\} = \mathbb{P}\{j \in \mathcal{S}_i(t)\} = p_j, \quad \forall\ i \in \mathcal{D}. \quad (3)$$

Note that p_j ($j \in \mathcal{S}$) is *not* a probability distribution since $\mathbb{E}\{\sum_{j\in\mathcal{S}} I_{ij}(t)\} = \sum_{j\in\mathcal{S}} p_j$ is the average number of connections of downloading peer i, which can be larger than 1 for multiple connections (i.e., parallel downloading). In other words, $I_{ij}(t)$ is *not* independent over j as they are constrained by the number of connections of a downloading peer (usually up to some constant). We note however that $I_{ij}(t)$ is $i.i.d.$ over i ($i \in \mathcal{D}$) under (A1) and (3).

We now have the following.

Proposition 1. *Under (3), we have*

$$\rho = \frac{\sum_{j\in\mathcal{S}}\left[1-(1-p_j)^{|\mathcal{D}|}\right]c_j}{\sum_{j\in\mathcal{S}} c_j}. \quad (4)$$

Proof: See Appendix A □

To better understand the meaning of the system utilization ρ in (4), let's consider a simple system with two source peers and two downloading peers where the capacities of two source peers are identical. Suppose that each downloading peer connects to each source with probability $p_j = \mathbb{E}\{I_{ij}\} = 0.5$, but under the constraint that $I_{i1}(t) + I_{i2}(t) = 1, i = 1, 2$, i.e., each downloading peer maintains a connection to exactly one of the two sources at any time. Then, from (4), we have $\rho = (1 - (1 - 0.5)^2) = 0.75$. This is indeed true as there are only two possible cases – (i) two downloading peers connect to different sources ($\rho = 1$), or (ii) two downloading peers connect to the same source ($\rho = 0.5$), each of which takes place with the same probability.

From Proposition 1, we see that $\rho = 1$ only when $p_j = \mathbb{P}\{I_{ij}(t) = 1\} = 1$ for all j. This suggests that each downloading peer should connect to *all possible source peers* in the network from purely the system utilization point of view. However, in reality, a downloading peer may have many potential source peers and it is impractical to connect to all of them simultaneously, hence we usually have $p_j < 1$.

3.2 Achievable Minimum Average Download Time

Before we show how we can derive the relation between the average download time and the system utilization, we first need a formal definition for the file download time.

Definition 2. *Let $C(t)$ denote the service capacity that a downloading peer "receives" from the network at time t ($t = 1, 2, \ldots$). The file download time T is the first time that the*

size of the accumulated received data exceeds the file size F. In other words, we have the following equation:

$$T = \min\left\{T > 0 \Big| \sum_{t=1}^{T} C(t) \geq F\right\} \tag{5}$$

The random variable T is the first hitting time of the cumulative process $\sum_{t=1}^{T} C(t)$ to reach level F. If $\{C(t),\ t \in \mathbb{N}\}$ are *independent and identically distributed* $(i.i.d.)$, then by assuming an equality in (5), we obtain from Wald's equation [28] that

$$F = \mathbb{E}\left\{\sum_{t=1}^{T} C(t)\right\} = \mathbb{E}\{C(t)\}\mathbb{E}\{T\}. \tag{6}$$

The expected download time, measured in slots, then becomes $\mathbb{E}\{T\} = F/\mathbb{E}\{C(t)\}$, which has been widely used in the literature.

However, the equality in (6) does not generally hold. Recall that the service capacity of each source peer can be different and can fluctuate over time. First, suppose that a downloading peer is only able to make *a single connection* and waits patiently for its session to complete. It was shown in [15] that the temporal correlation in the fluctuation of service capacity can make the average download time over any connection longer. In other words, if a downloading peer selects its source peer by a random variable $J \in [1, \ldots, |\mathcal{S}|]$, the average download time T_j, given that $J = j$, usually have the following relationship: $\mathbb{E}\{T_j\} \geq F/\mathbb{E}\{C_j(t)\}$, where equality is achieved only if each of $C_j(t), \forall j \in \mathcal{S}$ is *independent or weakly correlated* over t.

Even if the equality in (6) holds for whichever connection the downloading peer chooses, the heterogeneity of the network (different service capacities offered by different source peers) also makes the average download time longer. Suppose now $\mathbb{E}\{T_j\} = F/\mathbb{E}\{C_j(t)\}$ for each j. Then, observe from Jensen's inequality that $\mathbb{E}_J\left\{\frac{F}{\mathbb{E}\{C_J(t)|J\}}\right\} \geq \frac{F}{\mathbb{E}_J\{\mathbb{E}\{C_J(t)|J\}\}}$ where the equality is achieved when $\mathbb{E}\{C_j(t)\} = \mathbb{E}\{C_k(t)\}, \forall j,k \in \mathcal{S}$.

Hence, combining the effect of both temporal correlation and network heterogeneity, we arrive to

$$\begin{aligned}\mathbb{E}\{T\} &= \mathbb{E}_J\left\{\mathbb{E}\left\{T_J|J\right\}\right\} \\ &\geq \mathbb{E}_J\left\{\frac{F}{\mathbb{E}\left\{C_J(t)|J\right\}}\right\} \geq \frac{F}{\mathbb{E}_J\left\{\mathbb{E}\{C_J(t)|J\}\right\}},\end{aligned} \tag{7}$$

where the first inequality comes from the temporal correlation in the fluctuation of the capacity of each source peer and the second inequality comes from the network heterogeneity. Note that (7) is for the case in which the network has only one downloading peer and it only utilizes a single connection. We now investigate the performance of the case in which the network has multiple downloading peers in competition, each of which utilizes parallel connections to multiple source peers at the same time.

Assume that each downloading peer i can now have an average of L parallel connections, i.e. $\mathbb{E}\{\sum_{j\in\mathcal{S}} I_{ij}(t)\} = L$, and all downloading peers follow the "connect-and-wait" strategy for each of the connections. We have the following.

Theorem 1. *Let $\vec{c} = \{c_j\},\ j = 1, \ldots, |\mathcal{S}|$ represent the vector of average capacities of the source peers. Then*

$$\mathbb{E}\{T_i\} \geq \frac{F}{A(\vec{c})}\frac{\nu}{\rho} \tag{8}$$

where

$$\nu = \frac{|\mathcal{D}|}{|\mathcal{S}|} \quad \textit{and} \quad A(\vec{c}) \triangleq \frac{1}{|\mathcal{S}|}\sum_{j\in\mathcal{S}} c_j. \tag{9}$$

We define the parameter ν as the competition level, which is the average number of downloading peers each source peer has to service. Note that $\nu/A(\vec{c}) = \left(\sum_{j\in\mathcal{S}} c_j\right)/|\mathcal{D}|$ can be interpreted as the commonly perceived average "share" of system capacity that each downloading receives from the network .

Proof: Assume that a file of size F is divided into $|\mathcal{S}_i|$ pieces because the downloading peer utilizes parallel connections to download from the source set $\mathcal{S}_i$. Let F_{ij} denotes the size of the piece that the downloading peer i requests from source peer j. Since we are considering the specific downloading peer i, so we suppress the subscript i in F_{ij}. Clearly, we have $\sum_{j\in\mathcal{S}_i} F_j = F$, where $0 \leq F_j \leq F$. Note that p_j is independent of j and let T_{ij} denotes the time required to complete transferring piece F_j from j, as defined in (5), and we have

$$\begin{aligned}\mathbb{E}\{T_i\} &= \mathbb{E}\left\{\max_{j\in\mathcal{S}_i} T_{ij}\right\} = \mathbb{E}_{\mathcal{S}_i}\left\{\mathbb{E}\Big\{\max_{j\in\mathcal{S}_i} T_{ij}\Big|\mathcal{S}_i\Big\}\right\} \\ &\geq \mathbb{E}_{\mathcal{S}_i}\left\{\max_{j\in\mathcal{S}_i}\mathbb{E}\{T_{ij} \mid \mathcal{S}_i\}\right\} = \mathbb{E}_{\mathcal{S}_i}\left\{\max_{j\in\mathcal{S}_i}\mathbb{E}\{T_{ij}\}\right\},\end{aligned} \tag{10}$$

where $\mathbb{E}_{\mathcal{S}_i}$ is the expectation over the random set $\mathcal{S}_i$, and the last equality follows from the fact that S_i is randomly generated at $t = 0$, i.e., T_{ij} and S_i are independent.

For each piece of a file that peer i downloads from peer j, we generally have $T_{ij} \geq F_j/\mathbb{E}\{R_{ij}(t)\}$ [15]. Substitute $\mathbb{E}\{R_{ij}(t)\}$ with the result in (27) and we have

$$\mathbb{E}\{T_{ij}\} \geq \frac{F_j}{\mathbb{E}\{R_{ij}(t)\}} = |\mathcal{D}|\frac{F_j}{\mu_j c_j},$$

where we let $\mu_j = \left\{1 - (1-p_j)^{|\mathcal{D}|}\right\}$. Here, we can view μ_j as the probability that source peer j is utilized in any random time slot since $(1-p_j)^{|\mathcal{D}|}$ represents the probability that source peer j is connected to no downloading peers. Further, we can also view μ_j as the fraction of time it is contributing to the system. As the result $\mu_j c_j$ is then the average capacity that a source j can contribute to the system. Hence, equation (10) implies

$$\mathbb{E}\{T_i\} \geq |\mathcal{D}|\mathbb{E}_{\mathcal{S}_i}\left\{\max_{j\in\mathcal{S}_i}\frac{F_j}{\mu_j c_j}\right\}. \tag{11}$$

For any give set $\mathcal{S}_i$, we know that $F_j = F \cdot \mu_j c_j / \sum_{j\in\mathcal{S}_i}\mu_j c_j$ is a solution to the following optimiza-

tion problem:

$$\min\left\{\max_{j\in\mathcal{S}_i}\left\{\frac{F_j}{\mu_j c_j}\right\}\right\},\ \text{s.t.}\ \sum_{j\in\mathcal{S}_i} F_j = F,\ F_j \geq 0.$$

Thus, assume that F_j is allocated proportional to $\mu_j c_j$ and $\sum_{j\in\mathcal{S}_i} F_j = F$, we have

$$\max_{j\in\mathcal{S}_i}\left\{\frac{F_j}{\mu_j c_j}\right\} = \frac{F}{\sum_{j\in\mathcal{S}_i}\mu_j c_j}. \quad (12)$$

Further, note that $I_{ij}(0) = 1_{\{j\in\mathcal{S}_i\}}$ in the "connect-and-wait" strategy for each connection and we have

$$\mathbb{E}_{\mathcal{S}_i}\Big\{\sum_{j\in\mathcal{S}_i}\mu_j c_j\Big\} = \mathbb{E}\Big\{\sum_{j\in\mathcal{S}}\mu_j c_j I_{ij}(0)\Big\}$$
$$= \sum_{j\in\mathcal{S}}\mu_j c_j \mathbb{P}\{I_{ij}(0)=1\} = \sum_{j\in\mathcal{S}}\mu_j c_j p_j \leq \rho A(\vec{c})|\mathcal{S}|. \quad (13)$$

Recall from (4) that $\rho\sum_{j\in\mathcal{S}} c_j = \sum_{j\in\mathcal{S}}\mu_j c_j$ and $p_j \leq 1$, we have the inequality in (13). From (11), (12) and (13), observe that

$$\mathbb{E}\{T_i\} \geq |\mathcal{D}|\mathbb{E}_{\mathcal{S}_i}\left\{\max_{j\in\mathcal{S}_i}\frac{F_j}{\mu_j c_j}\right\} = |\mathcal{D}|\mathbb{E}_{\mathcal{S}_i}\left\{\frac{F}{\sum_{j\in\mathcal{S}_i}\mu_j c_j}\right\}$$
$$\geq \frac{F|\mathcal{D}|}{\mathbb{E}_{\mathcal{S}_i}\{\sum_{j\in\mathcal{S}_i}\mu_j c_j\}} \geq \frac{F|\mathcal{D}|}{\rho A(\vec{c})|\mathcal{S}|} = \frac{F}{A(\vec{c})}\frac{\nu}{\rho}. \quad (14)$$

This completes the proof. □

The function $A(\vec{c})$ is *the average system service capacity* from the perspective of a downloading peer. Note that the results in [15] that $\mathbb{E}\{T\} \geq F/A(\vec{c})$ is now just "a special case" of Theorem 1. The schemes described in [15] is the case when $|\mathcal{D}| = L = 1$ and $p_j = 1/|\mathcal{S}|$, which corresponds to $\rho = 1/|\mathcal{S}|$ from (4). Therefore, $\nu/\rho = 1$ in (8), and we have the result in [15].

From the derivation of Theorem 1, we can see that the naive "connect-and-wait" strategy needs to satisfy several conditions in order to achieve the minimum possible average download time, i.e. the equality in (8). First, the size for each piece downloaded from different sources must be determined at the beginning of a session such that F_j is proportional to $\mu_j c_j$, i.e $F_j = F\cdot(\mu_j c_j/\sum_{j\in\mathcal{S}_i}\mu_j c_j)$. In reality, it is hard to measure both c_j and μ_j accurately, and hence it is difficult to pre-allocate F_j. Even if we were able to allocate each F_j precisely proportional to $\mu_j c_j$, the possible temporal correlation of fluctuation in the service capacity of each source peer will result in a strict inequality in (11) and so does the heterogeneity in the average capacity with different source peers for the inequality in (14). The naive "connect-and-wait" strategy for each of the parallel connections is thus unlikely to achieve an equality in (8), and therefore we need to consider different algorithms to achieve such equality if at all possible.

4 Achieving the Minimum Average Download Time

Theorem 1 shows the minimum possible average download time in a stochastic P2P network. In this section, we first show that there exists a way to achieve the equality in (8) under all conditions. We then propose a centralized algorithm that maximizes the system utilization ρ, thereby further minimizing the (now achievable) minimum average download time (see (8)). Finally, we propose some possible distributed algorithm implementations that may achieve near optimal performance.

4.1 Dynamic Peer Selection

As our first step towards developing an algorithm that minimizes the average download time, we show switching peers periodically, i.e. a downloading peer i changes its source set $\mathcal{S}_i(t)$ after each t, can achieve the *equality* in (8). As opposed to the "connect-and-wait" strategy that the set of source peers remains "static" after a session begins, switching peer periodically causes the set of source peer to change dynamically during a download session. We will use the term dynamic peer selection and periodic switching interchangeably in the following sections. In our network setting, the source set $S_i(t)$ of each downloading peer i can change *both in its elements and its size* over time. Each downloading peer can have connection preference for one source peer over another. The capacity of each source peer is shared among the downloading peers connected to it to model the competition.

Under our time-varying dynamic source peer selection scheme, $\mathcal{S}_i(t)$ $(t = 1, 2, \ldots)$ is a *sequence* of random sets rather than a *constant* set randomly generated at $t = 0$ as the $\mathcal{S}_i(t)$ in Theorem 1. The service capacity that a downloading peer i receives from the entire network, $R_i(t)$, then becomes *the sum of random variables over a random set*. Here, we do not enforce a strict limit on the number of parallel connections in *each time slot*. Rather, we assume that the average number is limited to L, i.e. $\mathbb{E}\{\sum_{j\in\mathcal{S}} I_{ij}(t)\} = \sum_{j\in\mathcal{S}} p_j = L$. To show that we can have an equality in (8), we need to first show that the temporal correlation in received service capacity of each downloading peer is much reduced by our time-varying dynamic peer selection.

Note that the correlation function of a stationary random process $X(t)$ is defined by

$$\phi_X(\tau) = \frac{\text{Cov}(X(t), X(t+\tau))}{\text{Var}(X(t))}$$

and we have the following.

Theorem 2. *For any given downloading peer i, Let $X(t)$ and $Y(t)$ denote the $R_i(t)$ under the "connect-and-wait" and the dynamic peer selection, respectively. In general, we have*

$$\phi_Y(\tau) \leq \max_{j\in\mathcal{S}}\{p_j\}\phi_X(\tau). \quad (15)$$

For the special case of uniform (blind) selection, i.e $p_j = L/|\mathcal{S}|$, we have the following inequality:

$$\phi_Y(\tau) \leq \frac{L}{|\mathcal{S}|}\phi_X(\tau). \quad (16)$$

Proof: See Appendix B. □

Note that [15] considers only the case of blind selection with $L = 1$. Theorem 2 is in a much general form and shows that the temporal fluctuation in *each downloading peer's received capacity* is either uncorrelated or weakly correlated over time so that Wald's equation (6) holds true as long as $\max_j\{p_j\}$ is not too large.

Now, suppose that $\max_j\{p_j\}$ is properly selected to be some suitable value, say, 0.5. The correlation in the received capacity from the network is *at least* reduced by 50% compared to "connect-and-wait" strategy from (15). Hence, the receive capacity $R_i(t)$ for downloading peer i will be weakly correlated over time. Replacing $C(t)$ in (6) directly by $R_i(t)$ in (1) gives

$$\mathbb{E}\{T_i\} = \frac{F}{\mathbb{E}\{R_i(t)\}} = \frac{F}{\mathbb{E}\left\{\sum_{j\in\mathcal{S}} \frac{I_{ij}(t)}{\sum_{i\in\mathcal{D}} I_{ij}(t)} C_j(t)\right\}}$$

$$= \frac{F|\mathcal{D}|}{\sum_{j\in\mathcal{S}}\left\{1-(1-p_j)^{|\mathcal{D}|}\right\}c_j} \cdot \frac{A(\vec{c})}{A(\vec{c})} \quad (17)$$

$$= \left(\frac{F}{A(\vec{c})}\right)\left(\frac{|\mathcal{D}|}{|\mathcal{S}|}\right)\rho^{-1} = \frac{F}{A(\vec{c})}\frac{\nu}{\rho}. \quad (18)$$

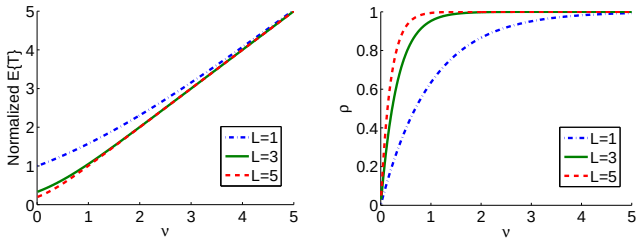
Equation (17) is a direct application of (27) and (18) is the result of rearranging the terms in (17). Equation (18) clearly shows that the equality in (8) is achieved by using dynamic peer selection.

We argue that $\max_{j\in\mathcal{S}}\{p_j\}$ is most likely to be strictly less than 1 in practice. If $\max_{j\in\mathcal{S}}\{p_j\} = 1$, it means that the downloading peers will connect to some source peers permanently throughout their entire download sessions. Note that the service capacity of a source peer fluctuates over time. A permanent connection will prevent the downloading peer from switching to some other potential source peers that can offer better capacity at the times when the service capacity of the currently connected source peer plummets.

Further, note that limiting the value of $\max_j\{p_j\}$ also implies limiting the average number of parallel connections a downloading peer can use, i.e. $\sum_{j\in\mathcal{S}} p_j = L \leq |\mathcal{S}| \cdot \max_{j\in\mathcal{S}}\{p_j\}$. Although it is a common belief that it is always better to open more parallel connections if a downloading peer wants to complete its download session quicker, such idea is generally not true as will be discussed in Section 4.2. If we are to connect to all possible source peers in the network, we lose the benefit of switching peers (correlation reduction) and hence the average download time will be much larger than the RHS of (8). Further, the measurement results in [20], [21] also show that all downloading peers utilizing parallel connections often do not give better performance than all peers using a single connection. The authors of [20], [21] suggest that the number of parallel connections should be limited. Theorem 2 actually agrees with this claim. In the next section, we will investigate the impact of utilizing parallel connections in more detail.

4.2 Impact of Parallel Connections

Here, we assume that each downloading peer changes its connections periodically using selecting peers uniformly at random, i.e. $p_j = p = L/|\mathcal{S}|$, $\forall j \in \mathcal{S}$. The average number of parallel connections used by each downloading peer L is chosen to be some small number so that that equation (18) holds, and we plot the relation between the average download time $\mathbb{E}\{T\}$, system utilization ρ, and the level of competition ν for different average number of parallel connections L according to (18) as Figure 1(a) and (b). The network has a fixed number of 40 source peers and the capacity of each source ranges from 1MB/min to 20MB/min with an increment of 0.5MB/min. The number of downloading peers varies from 1 to 200, hence $\nu \in [0.025, 5]$.



(a) $\mathbb{E}\{T\}$ vs ν. (b) ρ vs. ν

Fig. 1. The relation between the average download time $\mathbb{E}\{T\}$, system utilization ρ, and the level of competition ν for different average number of parallel connections L. $\mathbb{E}\{T\}$ is normalized over $F/A(\vec{c})$ and $\mathbb{E}\{X\}$ denotes the expectation of a random variable X.

By closely examine both Figure 1(a) and 1(b), we argue that the fundamental effect of parallel downloading is to increase the system utilization rather than directly reducing the average download time. Figure 1(a) shows that the benefit that downloading peers gain form utilizing parallel connections varies under different network settings. The common belief that parallel connections can help to reduce the average download time is true only when the competition in the network is very low. For example, when $\nu \leq 2$, we see that utilizing parallel connections indeed gives better performance over using a single connection in Figure 1(a). The reason is that parallel connections can much increase the system utilization. Taking $\nu = 1$ as an example, Figure 1(b) shows that the system utilization is increased from 60% to around 95% by increasing L from 1 to 3. On the other hand, there is no noticeable performance difference between using a single connection and using parallel connections when $\nu \geq 2$. The system performance in this region actually corresponds to the results in [20], [21] that large scale deployment of parallel connections does not always give better performance. Once again, if we refer to Figure 1(b), we can see that there is not much room to increase system utilization when ν is large regardless the value of L.

Even when the system is operating at the under-utilized region, utilizing more parallel connections may not guarantee better performance. First, each additional connection does not reduce the average download time by the same amount. Consider the case when $|\mathcal{S}| \gg |\mathcal{D}|L$,

and we have the following approximation

$$\rho = 1 - \left(1 - \frac{L}{|\mathcal{S}|}\right)^{|\mathcal{D}|} \approx 1 - \left(1 - L\frac{|\mathcal{D}|}{|\mathcal{S}|}\right) = L\nu.$$

Equation (18) approximately becomes $F/(A(\vec{c})L)$ and the average download time is inversely proportional to L. Second, the negative impact of temporal correlation may compromise the benefit we gain from using parallel connections. Consider the case when $\nu \approx 0.5$ in Figure 1, we may be able to reduce the average download time a little more by increasing the average number of parallel connections from 5 to 10 or more. However, increasing L implies that we will have less correlation reduction, and it was shown in [15] that different levels of temporal correlation in received capacity could increase the average download time up to 25% or more. Therefore, combining the effect of both parallel connections and temporal correlation, a large L may give the downloading peers worse performance compared with a small L. In summary, the average number of parallel connections should be limited to some small number.

4.3 Maximizing System Utilization

From the previous section, we demonstrate that changing the set of source peers dynamically over time can always achieve equality in (8) under the condition that both the average number of parallel connections L and the connection probability to each source peer $\max_{j\in\mathcal{S}}\{p_j\}$ are limited. Given that we always have (8), what is the minimum average download time? Note that the variables, $F, A(\vec{c})$, and ν in (18) are determined once the P2P network is formed. Downloading peers are not able to change these parameters. On the other hand, the system utilization ρ in (18) is partially determined by the peer selection probabilities (See (4)). Hence, in what follows, we consider how to assign selection probabilities to source peers in order to achieve optimal performance (minimal average download time).

Specifically, we assume that it is possible to measure or obtain the accurate values of $\mathbb{E}\{C_j(T)\} = c_j$ for each of the source peer j in the network. Following the notion in Section 3.2, we let each peer have an average of L parallel connections, i.e. $\mathbb{E}\{\sum_{j\in\mathcal{S}} I_{ij}(t)\} = \sum_{j\in\mathcal{S}} p_j = L < |\mathcal{S}|$. From (8), minimizing the average download time is equivalent to maximizing the system utilization ρ. From (4), we can formulate our problem as follows:

$$\max \quad \frac{\sum_{j\in\mathcal{S}}\left[1-(1-p_j)^{|\mathcal{D}|}\right]c_j}{\sum_{j\in\mathcal{S}} c_j} \tag{19}$$

$$\text{s.t} \quad \sum_{j\in\mathcal{S}} p_j = L, \quad 0 \le p_j \le 1 \tag{20}$$

Since $\left(1-(1-p_j)^{|\mathcal{D}|}\right)$ is strictly concave in p_j, the problem in (19)–(20) is a convex optimization problem. We can apply any standard convex programming technique [29] to solve the problem. First, without the loss of generality, assume that the vector $\vec{c} = [c_j, j \in \mathcal{S}]$ is sorted in a decreasing order, i.e, $c_1 \ge c_2 \ge c_3 \cdots$. Then, the optimal solution $\{p_j^*\}$ is

$$p_j^* = \begin{cases} 1 - \frac{K^*-L}{\sum_{j=1}^{K^*}\left(\frac{1}{c_j}\right)^{\frac{1}{(|\mathcal{D}|-1)}}}\left(\frac{1}{c_j}\right)^{\frac{1}{(|\mathcal{D}|-1)}}, & j \le K^* \\ 0, & j > K^* \end{cases} \tag{21}$$

Here, K^* is the maximum K such that $\frac{\mu(K)}{|\mathcal{D}|} < \frac{c_K}{C(K)}$, where $C(n) = \sum_{j=1}^{n} c_j$ and $\mu(n)$ is defined as

$$\mu(n) = \frac{|\mathcal{D}|}{C(n)}\left[\frac{n-L}{\sum_{j=1}^{n}\left(\frac{1}{c_j}\right)^{1/(|\mathcal{D}|-1)}}\right]^{(|\mathcal{D}|-1)}. \tag{22}$$

For any given $\mathcal{D}, \mathcal{S}, L$, (21)–(22) gives the optimal peer selection probabilities for any choice of $\{c_j\}, j \in \mathcal{S}$. In a typical P2P network with tens to hundreds of uploading and downloading peers, we argue by an example in the next section that $\max_{j\in\mathcal{S}}\{p_j^*\}$ is most likely to be some small number so that we in general have a significant amount of correlation reduction from Theorem 2, and we always have (18). Further, $\{p_j^*\}$ maximizes the system utilization ρ in (18), and the minimum possible average download time is achieved.

By careful investigation of the expression in (21)–(22), we can identify the optimal peer selection probabilities in some very special network settings. First, if the network is homogeneous, i.e. all source peers offer the same average service capacity ($c_j = c$ for all $j \in \mathcal{S}$), then $\mu(n)/|\mathcal{D}| = \frac{1}{n}\left(\frac{n-L}{n}\right)^{(|\mathcal{D}|-1)} < 1/n$ for any n. In this case, $K^* = |\mathcal{S}|$ and (21) becomes

$$p_j = 1 - \left(\frac{|\mathcal{S}|-L}{|\mathcal{S}|\left(\frac{1}{c}\right)^{1/(|\mathcal{D}|-1)}}\right)\left(\frac{1}{c}\right)^{1/(|\mathcal{D}|-1)} = \frac{L}{|\mathcal{S}|}.$$

Hence, uniform random selection among all possible source peers is the best strategy in a homogeneous environment. Second, recall that $|\mathcal{D}|$ is the number of competing downloading peers in the network. If $|\mathcal{D}|$ is large, i.e. $|\mathcal{D}| \to \infty$, then we have

$$\lim_{|\mathcal{D}|\to\infty}\left(\frac{1}{c_j}\right)^{1/(|\mathcal{D}|-1)} = 1$$

in (21), regardless of the value of $c_j > 0$. Thus, again, we have $p_j \approx L/|\mathcal{S}|$, i.e., uniform random selection. Note that this uniform random peer selection achieves near-optimal performance *only under very special network configurations* that we just described above, namely, either homogeneous network or a very large number of downloading peers. In all other scenarios such as intermediate number of downloading peers under heterogeneous environment, assigning connection probabilities for each source peer according to (21)–(22) clearly give far better performance in general. In what follows, we illustrate how the optimal connection probabilities change as the network setting varies.

4.4 Optimal Peer Selection Example

Suppose that we have a network with 40 source peers. The source peers are paired into 20 groups. The two source peers in each group have the same average service capacity, while the average service capacity of the group ranges from 1MB/min to 20MB/min in increments of 1MB/min. We set the average number of parallel connections $L = 3$ and vary the number of downloading peers $|\mathcal{D}|$ to calculate the optimal vector of connection probabilities using (21)–(22).

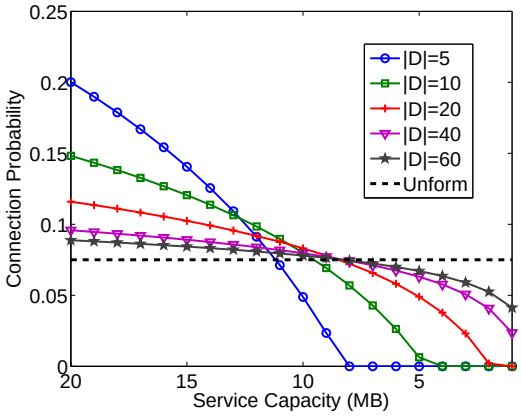


Fig. 2. The optimal connection probabilities to source peers (from (21)) offering different service capacities.

Figure 2 shows the computed optimal connection probabilities for 20 different groups, indexed with their average service capacities from 20MB/min to 1MB/min. All the lines in the figure display a common trend: assigning higher probabilities to source peers offering higher average capacities. Note that the $\max_j\{p_j\}$ is still very small even in a very heterogeneous network (we have service capacity of source peers ranging from 1MB/min to 20MB/min). Clearly, Figure 2 shows that the probability of connecting to peers offering the average capacity of 20MB/min should increase as $|\mathcal{D}|$ decreases. When there is no competition in the network, it is intuitive that a downloading peer should always choose the fastest source peers as [17], [18] suggested. However, how should the peer selection probability change if there exists competition in the network? Figure 2 also shows that the optimal connection vector tends to be "flatter" as the number of concurrent downloading peers $|\mathcal{D}|$ gets larger, and it approaches to that of uniform random selection when $|\mathcal{D}|$ goes to infinity, as we have discussed in Section 4.3. Although both the single downloading peer case and infinite downloading peer case are well studied, note that our algorithm finds the optimal vector of connection probabilities between these two extremes under stochastic and heterogeneous environments. Further performance comparison between the optimal peer selection and other selection algorithms will be presented in Section 5.

4.5 Distributed Adaptive Peer Selection Strategy

Note that the centralized peer selection algorithm discussed in the previous section requires the information about the "average" service capacity of each source peer and the number of downloading peers in the network. Such requirement suggests that we need a centralized authority to calculate the optimal connection probability for each downloading peers. Further, broadcasting the optimal solution to all downloading peers introduces much communication overhead. In a system like a P2P network, a distributed algorithm for calculating the connection probability would be more desirable because a distributed algorithm reduces the cost of building a centralized infrastructure and the cost of communication overhead. In this section, we show that we can have a distribution algorithm that achieves near optimal performance with only very little communication overhead. In other words, each downloading peer can calculate its own vector of connection probabilities using online measurements.

Our distributed algorithm is a modification of the centralized algorithm in Section 4.3. The global parameters in (21)–(22) are to be replaced by each downloading peer's own estimates. In other words, each downloading peer estimates the values of c_j and $|\mathcal{D}|$ from its own measurements during its download session. Both c_j and $|\mathcal{D}|$ can be inferred by the information about the number of downloading peers that are actively connected to a source peer.

Let $\chi_j(t) = \sum_{i\in\mathcal{D}} I_{ij}(t)$ denote the number of downloading peer connected to source peer j at time t. Suppose that each downloading peer is now able to obtain the information about $\chi_j(t)$ when it connects to source peer j. We emphasize that the source peers do not broadcast the process $\chi_j(t)$ to all downloading peers in every time slot t. Instead, the downloading peers ask for the value of $\chi_j(t)$ only when they decide to connect to source peer j at t. Such actions introduce very little (if any) communication overhead. Let $\chi_{ij}(t) = \chi_j(t) \cdot I_{ij}(t)$ be the total number of downloading peers sharing the capacity of source peer j, *seen by* the downloading peer i. A downloading peer i can then estimate the average capacity of source peer j by

$$\begin{aligned}\hat{c}_{ij} &= \frac{\sum_{t=1}^{T} R_{ij}(t)\chi_{ij}(t)}{\sum_{t=1}^{T} I_{ij}(t)} \\ &= \frac{\sum_{t=1}^{T}\left(\frac{I_{ij}(t)}{\sum_{i\in\mathcal{D}} I_{ij}(t)} C_j(t)\right)\left(\left(\sum_{i\in\mathcal{D}} I_{ij}(t)\right) I_{ij}(t)\right)}{\sum_{t=1}^{T} I_{ij}(t)} \\ &= \frac{\sum_{t=1}^{T} I_{ij}(t)C_j(t)}{\sum_{t=1}^{T} I_{ij}(t)}\end{aligned}$$

Clearly, $\hat{c}_{ij}$ is an unbiased estimator for $c_j = \mathbb{E}\{C_j(t)\}$.

We argue heuristically that $\chi_{ij}(t)$ can be used for estimating the number of downloading peers in the network as well. Recall that each downloading peer has an average of L parallel connections, hence the

average total number of connections is $|\mathcal{D}|L$. Note that $\hat{\chi}_{ij} = \frac{\sum_{t=1}^{T}\chi_{ij}(t)}{\sum_{t=1}^{T}I_{ij}(t)}$ is the estimate for $\mathbb{E}\{\sum_{i\in\mathcal{D}}I_{ij}(t)\}$, which is the average number of downloading peers connected to source peer j, from the downloading peer i's point of view. Summing up the average number of downloading peers connected to source peer j gives the total average number of connections in the network, i.e., $|\mathcal{D}|L$. Thus, each downloading peer i can use $\sum_{j\in\mathcal{S}}\hat{\chi}_{ij}$ to estimate the number of all downloading peers $|\hat{\mathcal{D}}|$ in the network. Then, the downloading peers can use their estimated system parameters to calculate the connection probability to each source peer by replacing c_j and $|\mathcal{D}|$ in (21) and (22) with $\hat{c}_{ij}$ and $|\hat{\mathcal{D}}|$.

Note that all the calculation so far is based on each peer's own observation from the network. Since no observation can be made without connection, we assume that $p_{ij} \geq \epsilon > 0$, where ϵ is a very small number to ensure that each downloading peer is able to observe and estimate the global information rather than simply ignores some source peers completely. Algorithm 1 summarizes our distributed algorithm for connection probability assignment.

Algorithm 1 Distributed Connection Prob. Assignment

1: $S_{ij} := \sum_{t=1}^{T} I_{ij}(t)$ with initial value 0
2: In each time slot T:
3: $R_{ij}(T)$:=received capacity from a source j.
4: $\chi_{ij}(T)$:=observed number of peers connected to j.
5: **while** !(file complete) **do**
6: **if** $\exists\ S_{ij} = 0$ **then**
7: Connect to L source peer from $\{j|S_{ij}=0\}$ uniformly. Let G denotes the selected set.
8: **else**
9: $|\hat{\mathcal{D}}|_i = \frac{\sum_j \hat{\chi}_{ij}}{L}$
10: Calculate p_{ij} using $|\hat{\mathcal{D}}|_i, \hat{c}_{ij}$ according to (21) and (22) with L in (20) replaced by $L' = L - |\mathcal{S}|\epsilon$
11: $p_{ij} = p_{ij} + \epsilon$
12: Connect to source peer j with probability p_{ij}. Let G denotes the set of connected source peers.
13: **end if**
14: $S_{ij} := S_{ij} + 1\ \forall j \in G$.
15: $\hat{c}_{ij} := \frac{(S_{ij}-1)\hat{c}_{ij}+R_{ij}(T)\cdot\chi_{ij}(T)}{S_{ij}}\ \forall j \in G$.
16: $\hat{\chi}_{ij} := \frac{(S_{ij}-1)\hat{\chi}_{ij}+\chi_{ij}(T)}{S_{ij}}\ \forall j \in G$.
17: **end while**

In Algorithm 1, S_{ij} holds the number of times that a downloading peer i connects to a source j. Only two variables $R_{ij}(T)$ and $\chi_{ij}(T)$ are needed from each source j in time slot T to carry out the connection probability assignment. Here, we briefly explain the key steps of Algorithm 1 in each time slot. From line 6 and 7, we first connect to L source peers that have not been connected ($S_{ij} = 0$) at a time to get a global estimates on the parameters for the network as quick as possible. When $S_{ij} > 0,\ \forall j$, a downloading peer i can then calculate the connection probability for each source peer j from lines 9 to 12 using (21) and (22) with the estimated parameters. Lines 14 – 16 are the steps to update each downloading peer i's estimated parameters, namely $\hat{c}_{ij}$ and $\hat{\chi}_{ij}$. We have to emphasize again that the estimates of both $\hat{c}_{ij}$ and $\hat{\chi}_{ij}$ cost almost no communication overhead because they are updated only when i decides to connect to j. In summary, our Algorithm 1 is simple and fully distributed. In Section 5, we show the performance of Algorithm 1 via NS-2 simulations.

4.6 Summary

Before we present our simulation result, we summarize the analytical results we have obtained so far. The service capacity of each source peer $j \in \mathcal{S}$ is denoted by a random process $C_j(t)$. Each of the downloading peer connects to source peer j with probability p_j. Table 4.6 shows the summary of our result on the average download time of the network.

File size	F
Avg. Capacity	$A(\vec{c}) = \frac{1}{\lvert\mathcal{S}\rvert}\sum_j c_j,\ c_j = \mathbb{E}\{C_j(t)\}$
Utilization	$\rho = \frac{\sum_{j\in\mathcal{S}}\left[1-(1-p_j)^{\lvert\mathcal{D}\rvert}\right]c_j}{\sum_{j\in\mathcal{S}}c_j}$
Competition Factor	$\nu = \frac{\lvert\mathcal{D}\rvert}{\lvert\mathcal{S}\rvert}$
Min. Avg. Time	$\frac{F}{A(\vec{c})}\frac{\nu}{\rho}$

TABLE 1
System parameters and performance metrics

To achieve the minimum average download time $\frac{F}{A(\vec{c})}\frac{\nu}{\rho}$, each peer has to make connection decisions in each time slot. At the beginning of each time slot, the downloading peer decides whether to connect to a source peer j by flipping a coin with probability of getting a head (making a connection) p_j. We have calculated the optimal connection probability p_j^* for each source peer j. Suppose the each source peer is indexed by its average service capacity c_j in descending order, then

$$p_j^* = \begin{cases} 1 - \frac{K^*-L}{\sum_{j=1}^{K^*}\left(\frac{1}{c_j}\right)^{\frac{1}{(|\mathcal{D}|-1)}}}\left(\frac{1}{c_j}\right)^{\frac{1}{(|\mathcal{D}|-1)}}, & j \leq K^* \\ 0, & j > K^* \end{cases}$$

The value K^* is the cut-off index and it is defined as

$$K^* = \max\left\{K \middle| \frac{\mu(K)}{|\mathcal{D}|} < \frac{c_K}{C(K)}\right\},$$

where

$$C(n) = \sum_{j=1}^{n} c_j,\ \mu(n) = \frac{|\mathcal{D}|}{C(n)}\left[\frac{n-L}{\sum_{j=1}^{n}\left(\frac{1}{c_j}\right)^{1/(|\mathcal{D}|-1)}}\right]^{(|\mathcal{D}|-1)}$$

5 Simulations

In this section, we use NS-2 simulations to compare the performance of peer selection strategies in a P2P network. A simple illustration of our network setting is

depicted in Figure 3. The internet cloud consists of many inter-connected nodes including backbone routers and edge routers. We assume that the Internet backbone has high bandwidth and does not introduce any congestion, implying that the main bottleneck is the access link of each peer. Such assumption enables us to run simulations using a star-shaped network topology in which the internet backbone is replaced by a single network node that has infinite capacity and zero delay. Although our goal is to improve the download performance, whether our algorithm is ISP-friendly is an interesting future topic. Some new work has started to consider the impact of peer selection on cross-ISP traffic [30], [31]. The access links of peers have different bandwidth limits and are connected to the "super" node representing the internet backbone. To reflect a general network setting for the access bandwidth, we configure 10% of the total 50 peers to have 10Mbps upstream (from peer node to the center node) capacity limit, 20% to have 5Mbps, 40% to have 1Mbps and the rest to have 100Kbps. These groups represent situations in typical LAN, high speed cable/DSL, low speed cable/DSL, and modem connections, respectively. We set 10Mbps as the downstream capacity limit so that the transmission bottleneck will most likely be at the source peers. The total number of network nodes is 51 (one center node and 50 peers participating in content transfer).

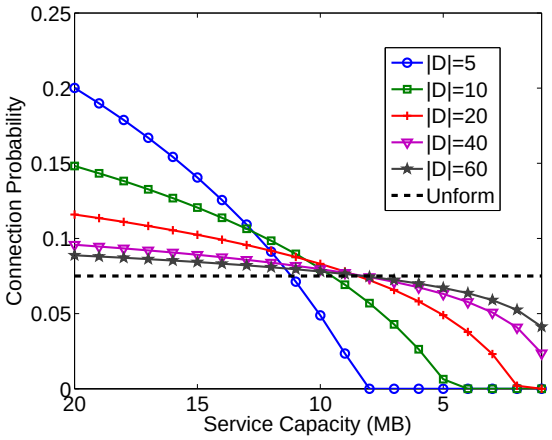


Fig. 3. The illustration of the network used in our NS-2 simulation. We assume that the Internet backbone does not introduce any congestion. The upstream access link of each peer is bandwidth limited.

Note that each node's upstream bandwidth is not only shared among other peers connected to it but also by some other network applications as in real world case. We model those non-P2P traffic by using the on/off Pareto traffic generators in NS-2. We set both the average "on" and "off" periods to be 10 minutes. When a traffic source is in the "on" state, it generates a constant bit-rate traffic at 90% of the full upstream capacity. Hence, the long term upstream average capacity for P2P traffic will be 55% of the access link bandwidth limit. The P2P file transfer traffic is carried over the TCP connection in NS-2. Note that in our NS-2 simulations, the actual capacity fluctuation in each of connections of a peer is governed by not only those asymmetric physical bandwidth limits, but also the actual TCP congestion control algorithm, the number of concurrent connections to the source peer, and the random Pareto on/off background traffic, i.e., it is stochastic in time and heterogeneous over space. As we have stated in the beginning of the paper, we are only interested in the peer selection algorithm itself rather than the problem of free-riders. All peers can serve as source peers, hence $|\mathcal{S}| = 50$. The number of downloading peers ranges from 10 to 50. Here, we set the number of average parallel connections $L = 2$.

First, we show the performance difference between the simple "connect-and-wait" strategy and dynamic peer selection with no connection preference (uniform selection). Under the "connect-and-wait" strategy, the file is divided into 2 pieces of equal size. Under dynamic peer selections, we set the basic data transfer unit (chunk) to be 16KB and each time slot is set to be 1 minute. We choose these two strategies for separate comparison purposes because "periodic uniform peer selection" alone already makes drastic improvement over the "connect-and-wait" strategy. We then compare the performance between periodic uniform peer selection and other algorithms to demonstrate the effectiveness of the optimal strategies more clearly. We first set the file size F=50MB for the ease of presenting our results. We have also run simulations using larger file sizes and observed the same result. ("connect-and-wait" strategy always performs worst)

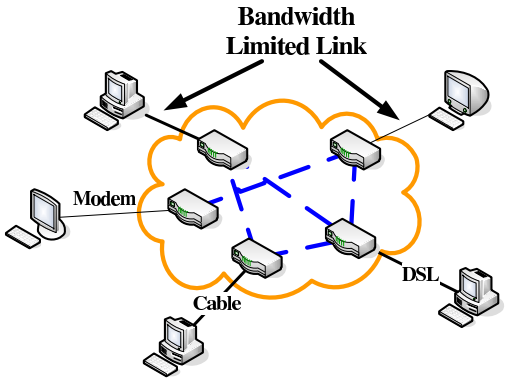


Fig. 4. The average download time for periodic uniform peer selection and "connect-and-wait" strategy. There are 50 nodes and no free-riders, hence $|\mathcal{S}| = 50$. The number of downloading peers varies from 10 to 50.

Figure 4 shows the results for our first scenario. The line marked with "uniform" is the result of the periodic uniform selection, and the line marked with "permanent" is the result for the "connect-and-wait" strategy. In all cases, we can see at least 50% reduction in average download time by using periodic uniform

peer selection. In some extreme cases, take $|\mathcal{D}| = 50$ as an example, the average download time for "connect-and-wait" strategy is almost 4 times longer than the periodic uniform selection. The result is exactly what we anticipated from our discussion in Section 3. Next, we show that under the dynamic peer selection schemes (non-uniform), the average download time can be further shortened by selecting peers using the optimal connection probabilities. The duration of each time slot is still 1 minute. We compare the performance between the uniform strategy (periodic uniform peer selection), the centralized optimal selection strategy in (21)–(22) and the distributed strategy (Algorithm 1) in Section 4.5. We change the file size to 200MB, which is typical for some video clips that contribute to long sessions so as to reflect a more general and realistic case.

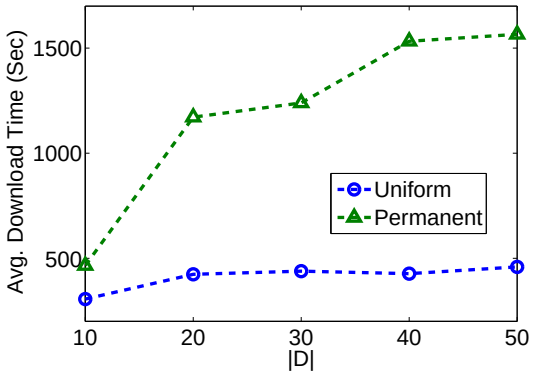


Fig. 5. The average download time for three dynamic peer selection strategies: periodic uniform peer selection (Uniform), the centralized optimal strategy in (21)–(22) (Optimal), and the distributed strategy in Algorithm 1 (Distributed).

Figure 5 shows the average download time along with its confidence interval for three different dynamic peer selection strategies. It is clear that both the centralize optimal strategy and the distributed strategy offer a performance boost over the simple uniform peer selection. (Note that this simple uniform selection is still much better than the usual connect-and-wait strategy as shown in Figure 4.) To further demonstrate the effectiveness of our strategies over uniform peer selection, we plot the reduction in the average download time of both centralized and distributed strategies over uniform peer selection in Figure 6.

It is clear that the optimal strategy can further reduce about 50% of the average download time compared with the periodic uniform selection strategy over most range of values of $|\mathcal{D}|$. We see that our simple distributed algorithm (developed from the centralized optimal strategy) can still reduce the average download time by 10% to 15%. Clearly, there is a performance gap between the distributed algorithm and the centralized optimal strategy. This is inevitable due to the estimation errors

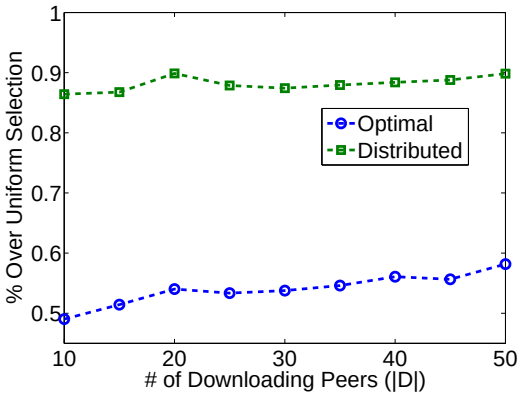


Fig. 6. The reduction in the average download time for different peer selection strategies over the uniform peer selection. The value in Y-axis is the fraction of the average download time of non-uniform selection algorithms over the average download time of uniform peer selection.

in our simple distributed algorithm. First, each peer estimates the level of competition of the entire network only through partial information and there will always be difference between 'local view' and global information. Second, the estimate $\hat{c}_{ij}$ may not be close to the real value of c_j. As we can see from Figure 5, the average download time for the distributed algorithm ranges from 1500 to 2000 seconds, which means that a downloading peer can make 25 to 35 connection switches during its session. Each peer will thus make around 60 observations of the system ($L = 2$). However, such number of observations is still not enough to obtain an accurate estimate for c_j, especially under highly stochastic and heterogeneous environment as in our setting. We point out these two factors to illustrate the possibility that the performance of the distributed algorithm can be further improved by increasing the accuracy of estimation of system variables, which can be made totally decoupled from our consideration of stochastic heterogeneous P2P networks under competition.

6 Conclusion and Future Work

In this paper, we jointly consider many factors that have impact on the average download time, which are often considered separately in the literature, and derive an optimal peer selection strategy that greatly improves content delivery performance in a stochastic heterogeneous P2P network. Specifically, we address some of the often neglected issues, namely, the competition for resource among multiple concurrent downloading peers, jointly with the spatial heterogeneity and temporal correlation in service capacities of the source peers. We derive the relationship between the average download time and system performance metrics such as system utilization and level of competition, and develop our optimal peer selection strategies based on this relationship.

We are also able to explain the fundamental impact of using parallel connections in stochastic P2P networks. In our work, we have discovered that downloading peers may not improve its performance (shortens average download time) by opening more parallel connections, sometimes it is even harmful to open many concurrent connections in a stochastic network. The system is saturated (achieves near 100% utilization) even with each downloading peer opening a very small number of parallel connections, say 3 or 5 (refer to Figure 1(b)). Our algorithm suggests that the downloading peers should not always connect the the source peer with the highest average service capacity. In fact, downloading peers should select its source peers probabilistically to distribute the system load to achieve the optimal system performance. We also show that a simple heuristic distributed algorithm derived from our centralized one can greatly improve performance over the uniform random peer selection, with almost no computational overhead.

In addition to file transfer applications, our algorithm may find application in streaming services. In most streaming services, a data buffer is required to allow smooth playback of the stream. How fast the data buffer is filled is very important. For example, the startup delay is exactly the time required to fill the buffer for the first time.

Certainly, we have to make some necessary assumptions in our model to make the analysis tractable. We did not consider many factors that also has significant impact on the system performance such as the effectiveness incentive algorithms, the efficiency of query mechanism, or the impact of network churn, just to name a few. For example, flood based query mechanism tend to give faster response to the querier as where the file is located. However, these mechanism generates a lot of query messages over the network. On the other hand, random walk based query messages reduces the over all query traffic but tends to have slower response. In file transfer or streaming applications, a session consists of both the time required for query and the actual file transfer. Of course, when considering the performance of a P2P network, both have to be considered. The reason for use to consider the actual file transfer time is because the query time is often much shorter than the actual file transfer [32]. Our result serves an lower bound for the file download time. An interesting and challenging future research topic would be to investigate how close we can get to this lower bound when we also consider other factors such as the effectiveness incentive algorithms, the efficiency of query mechanism, or the impact of network churn.

Appendix A
Proof of Proposition 1

Proof: From definition in (2), we have

$$\mathbb{E}\{R_{ij}(t)\} = \mathbb{E}\left\{\frac{I_{ij}(t)}{\sum_{i\in\mathcal{D}} I_{ij}(t)} C_j(t)\right\}$$

$$= \mathbb{E}\left\{\frac{I_{ij}(t)}{\sum_{i\in\mathcal{D}} I_{ij}(t)}\right\}\mathbb{E}\left\{C_j(t)\right\} \quad (23)$$

$$= \mathbb{E}\left\{\frac{1}{1+\sum_{k\neq i,k\in\mathcal{D}} I_{kj}(t)}\Big| I_{ij}(t)=1\right\} p_j c_j \quad (24)$$

$$= \mathbb{E}\left\{\frac{1}{1+\sum_{k\neq i,k\in\mathcal{D}} I_{kj}(t)}\right\} p_j c_j. \quad (25)$$

Note that (23) comes from (A3), i.e., $\{I_{ij}(t), \forall i\}$ and $C_j(t)$ are independent. We have (24) by using Bayes' rule and $\mathbb{E}\{C_j(t)\} = c_j$. Equation (25) comes from the previous observation that, for each j, $\{I_{ij}, i \neq k\}$ and I_{kj} are independent.

Let $X = \sum_{k\neq i,k\in\mathcal{D}} I_{kj}(t)$. Then, since $I_{ij}(t)$ are $i.i.d.$ over i, it follows that X is a binomial random variable with parameter $(|\mathcal{D}|-1, p_j)$. [1] Then, for $X \sim b(N,p)$, note that

$$\mathbb{E}\left\{\frac{1}{1+X}\right\} = \sum_{i=0}^{N}\left(\frac{1}{1+i}\right)\binom{N}{i}p^i(1-p)^{N-i}$$

$$= \frac{1}{(1+N)p}\sum_{i=0}^{N}\binom{N+1}{i+1}p^{i+1}(1-p)^{(N+1)-(i+1)}$$

$$= \frac{1}{(1+N)p}\left(1-(1-p)^{N+1}\right). \quad (26)$$

From (25) and (26), we get

$$\mathbb{E}\{R_{ij}(t)\} = \frac{1}{|\mathcal{D}|}\left[1-(1-p_j)^{|\mathcal{D}|}\right]c_j. \quad (27)$$

By substituting (27) into (2), we obtain (4). □

Appendix B
Proof of Theorem 2

Proof: Let's define a new processes $C'_j(t) = C_j(t) - \mathbb{E}\{C_j(t)\}$, then we have the following from the definition (1) that

$$\begin{cases} X(t) = \sum_{j\in\mathcal{S}} \frac{I_{ij}(0)}{\sum_{i\in\mathcal{D}} I_{ij}(0)}(C'_j(t)+c_j) \\ Y(t) = \sum_{j\in\mathcal{S}} \frac{I_{ij}(t)}{\sum_{i\in\mathcal{D}} I_{ij}(t)}(C'_j(t)+c_j) \end{cases}$$

Note that the processes $X(t)$ and $Y(t)$ are stationary and they both have the same mean and variance. We can compare $\phi_X(\tau)$ and $\phi_Y(\tau)$ by simply comparing $\mathrm{Cov}(X(t), X(t'))$ and $\mathrm{Cov}(Y(t), Y(t'))$, where $t' = t+\tau$. Let's define $\Phi_X(\tau) = \mathrm{Cov}(X(t), X(t'))$ and $\Phi_Y(\tau) = \mathrm{Cov}(Y(t), Y(t'))$ for notational simplicity.

We use the following property of the covariance of random variables:

$$\mathrm{Cov}\left(\sum_j X_j, \sum_j Y_j\right) = \sum_j\sum_k \mathrm{Cov}(X_j, Y_k). \quad (28)$$

1. For a binomial random variable X with parameter (N,p), (we write $X \sim b(N,p)$), we have

$$\mathbb{P}(X=i) = \binom{n}{i}p^i(1-p)^{N-i}, \quad i = 0,1,\ldots,N.$$

First, for any given downloading peer i, let

$$A_j(t) := \frac{I_{ij}(t)}{\sum_{i\in\mathcal{D}} I_{ij}(t)},$$

and we have the following by applying (28)

$$\begin{aligned}&\mathrm{Cov}(\sum_{j\in\mathcal{S}} A_j(t)C_j, \sum_{k\in\mathcal{S}} A_k(t')C_k(t'))\\ &= \sum_{j\in\mathcal{S}}\sum_{k\in\mathcal{S}} \mathrm{Cov}(A_j(t)C_j(t), A_k(t')C_k(t')). \end{aligned} \quad (29)$$

Note that each single term in (29) is

$$\begin{aligned}&\mathrm{Cov}(A_j(t)C_j(t), A_k(t')C_k(t'))\\ &= \mathrm{Cov}(A_j(t)(C'_j(t)+c_j), A_k(t')(C'_k(t')+c_k))\\ &= \mathbb{E}\{A_j(t)(C'_j(t)+c_j)A_k(t')(C'_k(t')+c_k)\}\\ &\quad - \mathbb{E}\{A_j(t)(C'_j(t)+c_j)\}\mathbb{E}\{A_k(t')(C'_k(t')+c_k)\}\\ &= \mathbb{E}\{A_j(t)C'_j(t)A_k(t')C'_k(t')\}\\ &\qquad - \mathbb{E}\{A_j(t)C'_j(t)\}\mathbb{E}\{A_k(t')C'_k(t')\}\\ &\quad + c_j\mathbb{E}\{C'_k(t')\}\left(\mathbb{E}\{A_j(t)A_k(t')\} - \mathbb{E}\{A_j(t)\}\mathbb{E}\{A_k(t')\}\right)\\ &\quad + c_k\mathbb{E}\{C'_j(t)\}\left(\mathbb{E}\{A_j(t)A_k(t')\} - \mathbb{E}\{A_j(t)\}\mathbb{E}\{A_k(t')\}\right)\\ &\quad + c_jc_k(\mathbb{E}\{A_j(t)A_k(t') - \mathbb{E}\{A_j(t)\}\mathbb{E}\{A_k(t')\}\})\\ &= \mathrm{Cov}(A_j(t)C'_j(t), A_k(t')C'_k(t'))\\ &\qquad + c_jc_k\mathrm{Cov}(A_j(t), A_k(t')) \end{aligned} \quad (30)$$

For the process $Y(t)$, the first term in (30) can be further simplified into

$$\begin{aligned}&\mathrm{Cov}(A_j(t)C'_j(t), A_k(t')C'_k(t'))\\ &\quad = \mathbb{E}\{A_j(t)C'_j(t)A_k(t')C'_k(t')\}\\ &\qquad - \mathbb{E}\{A_j(t)C'_j(t)\}\mathbb{E}\{A_k(t')C'_k(t')\}\\ &\quad = \mathbb{E}\{A_j(t)\}\mathbb{E}\{A_k(t')\}\mathbb{E}\{C'_j(t)C'_k(t')\},\end{aligned}$$

from the independence of $A_j(t), A_k(t')$ and $C_j(t)$. Let $\mathbb{E}\{A_j(t)\} = p_j\mathbb{E}\{1/(1+Z_j(t))\} = p_j\mathbb{E}\{W_j(t)\}$ where $Z_j(t) = \sum_{i\in\mathcal{D}} I_{ij}(t)$. From (A1), the process $Z_j(t)$ is $i.i.d.$ and so is $W_j(t)$, we drop the time index t for notational simplicity, i.e. $\mathbb{E}\{W_j\} = \mathbb{E}\{W_j(t)\}$. Recall that $A_j(t)$ and $A_j(t')$ are independent for $t' \neq t$, the second term in (30) for $Y(t)$ is zero. Further, $C_j(t)$ and $C_k(t')$ are independent for all $j \neq k$, and (29) for the random process $Y(t)$ now becomes

$$\begin{aligned}&\Phi_Y(\tau) = \mathrm{Cov}(Y(t), Y(t'))\\ &= \sum_{j\in\mathcal{S}} \mathbb{E}\{A_j(t)\}\mathbb{E}\{A_j(t')\}\mathrm{Cov}(C'_j(t), C'_j(t'))\\ &= \sum_{j\in\mathcal{S}} p_j^2(\mathbb{E}\{W_j\})^2\left(\mathbb{E}\{C'_j(t)C'_j(t')\}\right)\end{aligned} \quad (31)$$

Now, consider the process $X(t)$, after manipulating the terms, (29) becomes

$$\begin{aligned}&\Phi_X(\tau) = \mathrm{Cov}(X(t), X(t'))\\ &= \sum_{j\in\mathcal{S}} \mathrm{Cov}(A_j(0)C'_j(t), A_j(0)C'_j(t'))\\ &\qquad + \sum_{j\in\mathcal{S}} c_j^2\mathrm{Cov}(A_j(0), A_j(0))\\ &= \sum_{j\in\mathcal{S}} p_j\mathbb{E}\{W_j^2\}\mathbb{E}\{C'_j(t)C'_j(t')\} + \sum_{j\in\mathcal{S}} c_j^2\mathrm{Var}(A_j(0)),\end{aligned} \quad (32)$$

where $\mathrm{Var}(X)$ denotes the variance of a random variable X.

We can show from (32) and (31) that

$$\begin{aligned}&\frac{\phi_X(\tau)}{\phi_Y(\tau)} = \frac{\Phi_X(\tau)}{\Phi_Y(\tau)}\\ &= \frac{\sum_{j\in\mathcal{S}} p_j\mathbb{E}\{W_j^2\}\mathbb{E}\{C'_j(t)C'_j(t')\}}{\sum_{j\in\mathcal{S}} p_j^2(\mathbb{E}\{W_j\})^2\mathbb{E}\{C'_j(t)C'_j(t')\}}\\ &\qquad + \frac{\sum_{j\in\mathcal{S}} c_j^2\mathrm{Var}(A_j(0))}{\sum_{j\in\mathcal{S}} p_j^2(\mathbb{E}\{W_j\})^2\mathbb{E}\{C'_j(t)C'_j(t')\}} \quad (33)\\ &\geq \frac{1}{\max_j\{p_j\}} \quad (34)\end{aligned}$$

The inequality in (34) comes from $\sum_{j\in\mathcal{S}} p_j^2 \leq \max_j\{p_j\}\sum_{j\in\mathcal{S}} p_j$ and $\left(\mathbb{E}\{W_j^2\}\right)^2 \leq \mathbb{E}\{W_j^2\}$ in the first term of (33), and the second term in (33) is always non-negative. Thus we complete the proof. □

REFERENCES

[1] Y. M. Chiu and D. Y. Eun, "On the performance of download strategies in a p2p like network," in *IEEE Globecom*, Washington, DC, Nov 2007.

[2] B. Cohen, *BitTorrent Protocol Specification*. [Online]. Available: http://www.bittorrent.org/protocol.html

[3] *The Annotated Gnutella Protocol Specification v0.4*, The Gnutella Developer Forum. [Online]. Available: http://rfc-gnutella.sourceforge.net/developer/stable/index.html

[4] E. Sharman Networks, *Kazaa*, Kazaa. [Online]. Available: http://www.kazaa.com/us/help/new_p2p.htm

[5] X. Yang and G. de Veciana, "Service capacity of peer to peer networks," in *Proceedings of IEEE Infocom*, Mar. 2004.

[6] S. Saroiu, K. P. Gummadi, and S. D. Gribble, "A measurement study of peer-to-peer file sharing systems," in *Proceegins of ACM Multimedia Computing and Networking (MMCN)*, 2002.

[7] K. P. Gummadi, R. J. Dunn, and S. Saroiu, "Measurement, modeling, and analysis of a peer-to-peer file sharing workload," in *Proceedings of ACM Symposium on Operating Systems Principles (SOSP)*, 2003.

[8] B. Cohen, "Incentives build robustness in bittorrent," in *Proceedings of the First Workshop on the Economics of Peer-to-Peer Systems*, Berkeley, CA, Jun. 2003. [Online]. Available: http://citeseer.ist.psu.edu/cohen03incentives.html

[9] S. Jun and M. Ahamad, "Incentives in bittorrent induce free riding," in *ACM SIGCOMM workshop on Economics of peer-to-peer systems*, Philadelphia, Pennsylvania, Aug. 2005.

[10] M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy, and A. Venkataramani, "Do incentives build robustness in bittorrent?" in *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, Cambridge,MA, Apr. 2007.

[11] K. Eger and U. Killat, "Fair resource allocation in peer-to-peer networks," *Computer Communications*, vol. 30, no. 16, pp. 3046–3054, Jun. 2007.

[12] T. Locher, S. Schmid, and R. Wattenhofer, "Rescuing tit-for-tat with source coding," in *IEEE International Conference on Peer-to-Peer Computing (P2P'07)*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 3–10.

[13] K. Eger and U. Killat, "Bandwidth trading in bittorrent-like p2p networks for content distribution," *Computer Communications*, vol. 31, no. 2, pp. 201–211, 2008.
[14] D. Qiu and R. Srikant, "Modelling and performance analysis of bittorrent-like peer-to-peer networks," in *Proceedings of ACM Sigcomm*, Aug. 2004.
[15] Y. M. Chiu and D. Y. Eun, "Minimizing file download time in stochastic peer-to-peer networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 2, pp. 253–266, Apr. 2008.
[16] D. S. Bernstein, Z. Feng, and B. N. Levine, "Adaptive peer selection," in *Proceedings of International Workshop on Peer-to-Peer Systems (IPTPS)*, Berkeley, CA, Feb. 2003.
[17] M. Adler, R. Kumar, K. Ross, D. Rubenstein, D. Turner, and D. D. Yao, "Optimal peer selection in a free-market peer-resource economy," in *Workshop on Economics of Peer-to-Peer Systems*, Cambridge, MA, Jun. 2004.
[18] ——, "Optimal peer selection for p2p downloading and streaming," in *Proceedings of IEEE Infocom*, Miami, FL, Mar. 2005.
[19] "The network simulator ns-2," http://www.isi.edu/nsnam/ns/.
[20] C. Gkantsidis, M. Ammar, and E. Zegura, "On the effect of large-scale deployment of parallel downloading," in *Proceedings of IEEE Workshop on Internet Applications (WIAPP)*, Jun. 2003.
[21] S. Koo, C. Rosenberg, and D. Xu, "Analysis of parallel downloading for large file distribution," in *Proceedings of IEEE International Workshop on Future Trends in Distributed Computing Systems (FTDCS)*, May 2003.
[22] M. Lin, B. Fan, J. C. S. Lui, and D. M. Chiu, "Stochastic analysis of file-swarming systems," *Performance Evaluation*, vol. 64, no. 9-12, pp. 856–875, 2007.
[23] T. Bonald, L. Massoulié, F. Mathieu, D. Perino, and A. Twigg, "Epidemic live streaming: optimal performance trade-offs," in *Proceedings of ACM Sigmetrics*. New York, NY, USA: ACM, 2008, pp. 325–336.
[24] Y. Qiao, F. E. Bustamante, P. A. Dinda, S. Birrer, and D. Lu, "Improving peer-to-peer performance through server-side scheduling," *ACM Trans. Comput. Syst.*, vol. 26, no. 4, pp. 1–30, 2008.
[25] M. Zhang, Q. Zhang, L. Sun, and S. Yang, "Understanding the power of pull-based streaming protocol: Can we do better?" *Selected Areas in Communications, IEEE Journal on*, vol. 25, no. 9, pp. 1678–1694, December 2007.
[26] N. Hu and P. Steenkiste, "Evaluation and characterization of available bandwidth probing techniques," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 6, pp. 879–894, Aug. 2003.
[27] M. Jain and C. Dovrolis, "End-to-end estimation of the available bandwidth variation range," in *Proceedings of ACM Sigmetrics*, Jun. 2005.
[28] S. M. Ross, *Stochastic Processes*, 2nd ed. New York: John Wiley & Son, 1996.
[29] D. P. Bertsekas, *Nonlinear Programming*. Belmont, MA, USA: Athena Scientific, 1995.
[30] D. R. Choffnes and F. E. Bustamante, "Taming the torrent: a practical approach to reducing cross-isp traffic in peer-to-peer systems," in *ACM SIGCOMM*, vol. 38, no. 4. New York, NY, USA: ACM, 2008, pp. 363–374.
[31] H. Xie, Y. R. Yang, A. Krishnamurthy, and A. S. Y. Liu, "P4p: Provider portal for applications," in *ACM SIGCOMM*. New York, NY, USA: ACM, 2008.
[32] Intel, "Peer-to-peer content distribution: Using client pc resources to store and distribute content in the enterprise," Intel, Tech. Rep., September 2003. [Online]. Available: http://www.intel.com/it/digital-enterprise/peer-peer-content-distribution.pdf

Yuh-Ming Chiu received his B.S. degree from the Department of Communication Engineering, National Chiao Tung University, Taiwan, and M.S. degree from the Department of Electrical Engineering, National Tsing Hua University, Taiwan, in 1997 and 2000, respectively. He received his Ph.D. degree in Electrical and Computer Engineering from North Carolina State University, Raleigh, NC, in 2009. He is currently a senior software engineer at Yahoo! His research interests include queueing analysis, peer-to-peer networks, and distributed systems.

Do Young Eun received his B.S. and M.S. degree in Electrical Engineering from Korea Advanced Institute of Science and Technology (KAIST), Taejon, Korea, in 1995 and 1997, respectively, and Ph.D. degree from Purdue University, West Lafayette, IN, in 2003. Since August 2003, he has been with the Department of Electrical and Computer Engineering at North Carolina State University, Raleigh, NC, where he currently an associate professor. His research interests include network modeling and performance analysis, mobile ad-hoc/sensor networks, mobility modeling, congestion control, resource allocation. He is a member of Technical Program Committee of various conferences including IEEE INFOCOM, ICC, Globecom, ACM MobiHoc, ICDCS, IEEE IPCCC, and ICCCN. He received the Best Paper Awards in the IEEE ICCCN 2005 and IEEE IPCCC 2006, and the National Science Foundation CAREER Award 2006. He supervised and co-authored a paper that received the Best Student Paper Award in ACM MobiCom 2007.