A New Stable Peer-to-Peer Protocol with Non-Persistent Peers

Omer Bilgen and Aaron B. Wagner
School of Electrical and Computer Engineering, Cornell University

I. ABSTRACT

Recent studies have suggested that the stability of peer-to-peer networks may rely on persistent peers, who dwell on the network after they obtain the entire file. In the absence of such peers, one piece becomes extremely rare in the network, which leads to instability. Technological developments, however, are poised to reduce the incidence of persistent peers, giving rise to a need for a protocol that guarantees stability with non-persistent peers. We propose a novel peer-to-peer protocol, the group suppression protocol, to ensure the stability of peer-to-peer networks under the scenario that all the peers adopt non-persistent behavior. Using a suitable Lyapunov potential function, the group suppression protocol is proven to be stable when the file is broken into two pieces, and detailed experiments demonstrate the stability of the protocol for arbitrary number of pieces. Subsequent incorporations of the group suppression protocol into BitTorrent while retaining most of BitTorrent’s core mechanisms is also presented. Subsequent simulations show that under certain assumptions, BitTorrent with the official protocol cannot escape from the missing piece syndrome, but BitTorrent with group suppression does.

II. INTRODUCTION

In a peer-to-peer network, a file is divided into a number of pieces, and each peer uploads the pieces obtained so far to other peers while it continues to download the remaining pieces. This results in high utilization of the bandwidth of all peers leading to scalability, which marks a huge improvement over traditional server-client structure. Such systems have been deployed widely, as evident from their number of users and the fraction of overall internet traffic they command [1], and they are poised to become even more prominent with the rise of fog networking [2], [3]. A high level p2p protocol can be understood to be a set of rules that governs how a peer contacts another peer and chooses the piece to upload/download to/from that peer, depending on whether a “push” or “pull” policy is employed, respectively. Network and system will be used interchangeably to refer to the collection of all peers throughout the paper. Also we define a peer to be an incomplete peer if it does not have all the pieces of the file and a seed if it does.

Peer-to-peer networks with random peer and piece selection policies have been shown to struggle with what Hajek and Zhu [4], [5] call the missing piece syndrome in which very few peers possess a missing piece while most of the peers, referred to as the one club in [4], have all the pieces but the missing piece. A qualitative argument why such an imbalanced state persists in the network is as follows: Assume that the one club currently dominates the network, uniformly random peer and piece selection policies are in effect, the peers are non-persistent in the sense that the peers depart as soon as they have the entire file, and the new peers arrive into the network with no pieces. As a result of the uniformly random contact policy, any new peer will contact peers mostly from the one club and thereby become a member of the one club before it can obtain the missing piece. Even when the new peers obtain the missing piece before they are pulled to the one club, they would download the rest of the file quickly thanks to the large upload capacity of the one club. Therefore, they would depart without disseminating the missing piece to other peers long enough to significantly decimate the one club. Hence, the number of peers in the system diverges to infinity once the system is overwhelmed by the one club. A more thorough discussion of the dynamics of the missing piece syndrome can be found in [4]. In the same paper, it has been proven that if the upload rate of the fixed seed is greater than the arrival rate of new peers, who carry no pieces under the model, the system is stable. On the other hand, if the arrival rate of peers exceeds the upload rate of the fixed seed, then the system experiences the missing piece syndrome and is unstable.

With an ideal p2p protocol the system should remain stable irrespective of the relationship between the arrival rate and the upload rate of the seed, as the arrival rate cannot be internally controlled by the protocol. Zhu and Hajek’s protocol [6] establishes that if peers linger in the system on average long enough to upload one piece, then the system is stable regardless of the arrival rate. This highlights the significance of having persistent peers from stability point of view. Based on this study, one could reasonably posit that persistent peers are the main reason why many experimental works evaluating actual trace logs of large p2p networks find their performance to be satisfactory.

As per capita media consumption increases [7] and moves to wireless devices [1], however, one expects the incidence of persistent peers to decline, for three reasons. First, data consumption on wireless devices is generally priced differently than for wired terminals. In particular, most wireless providers in the U.S. charge users or lower users’ speed when their data usage exceeds a certain threshold, whereas most wired data plans allow for unlimited data transfers, constrained only by the rate. In the wireless case, uploading counts toward the overall data usage as much as downloading, and this
constitutes a substantial incentive for a mobile p2p user to behave non-persistently. Second, power consumption is a significant concern for mobile users, and hence it is less likely for a mobile p2p user to dwell in the system after the download is complete than it is for a user of a wired system. Finally, increases in per-user data consumption [7], in the absence of attending increases in network capacity, will strain users’ access and discourage them from being persistent peers. One should note that it is difficult for a protocol to penalize peers for being non-persistent since, by definition, non-persistent peers have left the p2p network and, therefore, lie outside its province.

There have been noteworthy attempts to devise stable protocols with non-persistent peers and arbitrary arrival rates. Reittu [8] suggests that all the peers sample three random peers and download a piece chosen uniformly at random from the set of pieces that are found in exactly one of these three peers, and they skip if the set is empty. Experimental results appear to demonstrate the stability of the protocol when the system with one fixed seed begins with the extreme case of all other peers lacking only one common piece, although no proof was provided for stability. In a follow-up work, Norros et al. [9] showed the global stability of this protocol under large deterministic system limits when the file is composed of two pieces. Later, global stability of the underlying stochastic system for arbitrary number of pieces was proved by Oğuz et al. in [10]. In the same work, Oğuz et al. [10] also provide a new provably stable scheme in which only new peers that arrive with no piece apply the rule in [8] and the peers lacking only one piece contact m peers and download the last piece only if every piece they have shows up at least twice in aggregate piece profile of these m peers. The protocol requires all other peers to contact a randomly chosen peer and download a piece chosen uniformly at random. The implementation of such protocol burdens the one club peers in the sense that they are not only prevented from downloading the last piece and consequently departing, but also they are required to keep uploading to the network while waiting to obtain the last piece. Thus, concerns for power consumption of mobile devices and internet data usage may make this protocol unappealing to some p2p users. The same concern can be raised for the protocol in [8].

We propose the group suppression protocol, a stable protocol with non-persistent peers and arbitrary arrival rates. Recognizing that the formation and persistence of the one club lies at the heart of instability as discussed in [4], in order to achieve stability one should intuitively aim to stop the one club from recruiting new members. This goal is accomplished by the group suppression protocol in the following way: First define a group as a collection of peers that share the same piece profile. And a group of peers whose population is strictly greater than any other group of peers in the network is called the largest club. The group suppression protocol dictates that a peer from the largest club to upload only to peers who hold greater number of pieces than it does and to refuse the upload to all the other peers when it contacts them. An upload from a peer in the largest club to any peer holding more pieces is allowed because such an upload does not draw new members into the largest club. To compare the group suppression protocol to the one suggested by Oğuz et al. in [10], consider a scenario in which the largest club consists of peers with all the pieces but one common piece, which is also the one club. In the group suppression protocol, the largest club members download the last piece and leave as soon as they encounter a peer that has the last piece. Furthermore, they upload nothing since there are no peers that are eligible to receive a piece from them. In the latter protocol, when the largest club peers encounter the last piece, they may be prohibited from downloading it based on the piece profile of their m contacts, and they continue uploading while waiting for another contact. Also, when the largest club is very crowded, its peers’ waiting times are consequently prolonged. The group suppression protocol compensates the largest club for their extended wait time in the network by allowing them to reduce their upload rate, which constitutes another desirable feature of the protocol.

In a similar spirit to the group suppression protocol, throttling the upload capacity of peers or even servers has been proposed to increase the performance of p2p systems. Zhang et al. [11] considered hybrid p2p system with a file consisting of a single piece and established via a fluid model that throttling the server capacity under certain conditions minimizes the number of peers compared with utilizing the entire capacity of the server. Their model, however, does not capture the missing piece syndrome. de Souza e Silva et al. [12] propose a protocol in which all the peers lacking only one piece, not necessarily the rarest piece, reduce their upload rate. Although this change in their closed system analysis where a departure causes an arrival leads to substantial gain in the maximum achievable throughput, the scheme requires fine-grained tuning.

Other related work is presented in II-A. A detailed system model, the stability theorem for two pieces, the stability conjecture for arbitrary number of pieces, and a discussion of the proof technique are given in Section III. Section IV offers simulation results confirming the stability theorem and the stability conjecture, and Section V provides the description and the simulations of BitTorrent modified to include the group suppression protocol.

A. Other Related Work

Massoulié and Vojnovic [13] proved global stability in a model with non-persistent peers if all new peers arrive into the system with one uniformly random piece of the file uploaded by the fixed seed. The fixed seed’s upload rate, however, may not be large enough to allow this in practice.

One of the earliest p2p models suitable to analytic analysis appeared in [14]. The file consists of only one piece, which leads to a system that distinguishes peers based on whether they are seeds or not, rather than based on what fraction of the file they have. Furthermore, all incomplete peers are assigned the same effectiveness parameter and the incomplete peers exit the network according to a certain rate after becoming a
seed, which means some of the peers are persistent. Under this model, they proved local stability of the associated deterministic system. Qiu and Sang [15] later established global stability under the same model. The model in [14] was extended by [16] and [17].

Considerable attention has been paid to improving the stability properties of p2p networks by connecting different swarms where swarm refers to the set of all peers that want to download the same particular file [18], [19], [20]. Bundling a number of files has been shown to increase the content availability and this benefit overcomes the burden of downloading unnecessary content for peers interested in unpopular files, which can clearly be seen in reduced download time in [20]. Lacking any seeds, a network with non-persistent peers who aim to obtain some distinct files of size one was considered by Zhou et al. in [19]. Peers with caches large enough to hold all the files enter the network with a file that some of the other peers are interested in. One notable result of the paper is that at most one swarm out of all swarms in the network is unstable. Zhu et al. [18] introduce a model in which, unlike [19], there exists a fixed seed, and non-persistent peers join the network empty, and the sets of pieces corresponding to distinct swarms may or may not be disjoint [18]. In case of disjoint swarms, they determined that if the maximum of the arrival rates of all swarms are less than the upload rate of seed, then the overall system is stable and if it exceeds the upload rate of seed then the overall system is unstable. Zhang et al. [21] propose forming coalitions between peers that paves the way for different peer and piece selection policies that may surpass the performance of BitTorrent’s peer and piece selection policies.

Other useful references include [22], [23], and [24].

III. Problem Setup and Main Theorem

We first describe the protocol used in Hajek and Zhu [4]. We then define the group suppression protocol by describing how it deviates from the Hajek and Zhu protocol.

In Hajek and Zhu’s model [4], which will be referred to as the unstructured p2p protocol, the file to be downloaded is divided into k pieces. Arrivals follow a Poisson process of rate λ. The new peers do not carry any pieces. The fixed seed remains in the system permanently. The incomplete peers are assumed to be non-persistent. The peer that initiates the contact uploads a piece to the other peer. In other words, the model employs a push policy instead of a pull policy. The uploading peer applies the random useful piece selection policy to determine the piece to upload, which amounts to selecting a piece among all useful pieces uniformly at random. If a contact occurs and the uploading peer has at least one piece that the other peer does not possess, we assume the upload of the randomly chosen piece takes place instantaneously. The time at which an incomplete peer contacts another peer is determined by Poisson processes with rate μ independent from an incomplete peer to incomplete peer. On the other hand, the seed contacts other peers according to a Poisson process of rate Uₙ. When it is time for a peer to contact another peer, it uniformly picks a peer from the set of all peers, which is called random peer selection policy. We allow all peers to select themselves to simplify calculations.

We now build the group suppression protocol upon the unstructured p2p protocol by employing two modifications to it. The first and arguably less significant modification requires the fixed seed to select a peer uniformly at random among the peers with the least number of pieces rather than uniformly at random among all peers and to upload a random useful piece. This seed policy is usually called the most deprived policy. It is sometimes combined with another rule that the seed upload the rarest piece rather than a random piece, which implies the seed has access to every peer’s piece profile in the network [12], [25]. The group suppression protocol allows the seed to upload a random useful piece.

To understand the second modification, recall from Section II that two peers belong to the same group if their piece profiles exactly match each other and the strictly most populous group is designated as the largest club. We assume for now that peers have access to this central knowledge. Notice that the network may not necessarily have a largest club at all times. The backbone of the group suppression protocol relies on preventing a group of peers whose members lack only one common piece from dominating the network. In order to accomplish that, the second deviation from the unstructured p2p protocol, which earns the group suppression protocol its name, requires the peers in the largest club to deny the upload to any peer with a smaller or equal number of pieces whenever they initiate a contact with such a peer.

The group suppression protocol contributes to stability by curbing the growth of the largest club and by concentrating the seed’s upload on the peers with the least number of pieces. Since the largest club rapidly recruits other peers once it starts to dominate the network, stopping the largest club requires eradication of its source of growth, which is uploads from the largest club to peers who hold a subset of the largest club’s pieces, or more generally, all the peers except the ones holding greater portion of the file. The peers with greater number of pieces than the largest club peers cannot be absorbed into the largest club. Therefore, the largest club is allowed to utilize its upload capacity for these peers but not for others. The protocol’s rule concerning the seed originates from the intuition that if the seed uploads a rare piece to a peer, it is preferable that the peer be one who will remain in the system for as long as possible. A peer with the least number of pieces is presumably more likely to stay longer in the network than other peers.

The group suppression protocol might appear to require a certain level of centralization because the seed must find the peers with smallest number of pieces in the system, and every peer needs to determine if they belong to the largest club when they contact another peer. However, both elements of centralization in the protocol can be relaxed for practical implementations of p2p systems. First, finding the peers with the smallest number of pieces can be guaranteed by giving some of the newly arriving peers’ id’s to the seed upon their
arrival. It is likely that the peers with the least amount of the file are also recent arrivals, because of random contact policy. Second, the peers can estimate whether they belong to the largest club based on their recent contacts. For example, they may store the last \( l \) contacts’ piece profiles. If the fraction of those \( l \) piece profiles that exactly match the piece of profile of the peer exceeds a certain threshold, then it declares itself a member of the largest club and uploads accordingly. Apart from the small probability event that the last \( l \) contacts presents an inaccurate view of the network, another concern could be that the \( l - 1 \) contacts’ piece profiles may have changed since they were contacted by the peer. However, since the missing piece syndrome follows the formation of the one-club and the club’s piece profile cannot change due to the scarcity of the missing piece over time, accumulating the piece profiles of peers is expected to perform almost as well as instantaneously-sampled \( l \) contacts from stability standpoint.

We state the theorem regarding the stability of the group suppression protocol for \( k = 2 \) and also conjecture the stability of the protocol for arbitrary \( k \).

**Main Theorem:** If the group suppression protocol is employed and \( k = 2 \), then the continuous-time Markov chain is positive recurrent for any \( (\lambda > 0, U_s > 0, \mu > 0) \). That is, the p2p network with the group suppression protocol and \( k = 2 \) is stable for any \( (\lambda > 0, U_s > 0, \mu > 0) \).

**Conjecture 1:** Under the group suppression protocol for arbitrary \( k \), the continuous-time Markov chain is positive recurrent for any \( (\lambda > 0, U_s > 0, \mu > 0) \). That is, the p2p network with the group suppression protocol and arbitrary \( k \) is stable for any \( (\lambda > 0, U_s > 0, \mu > 0) \).

**Sketch of the proof of Main Theorem:**
We employ the Foster-Lyapunov proposition to prove positive recurrence of the underlying Markov chain. To state the proposition, we need to define the drift of a potential function 

\[
DV(s) = \sum_{y \neq s} q(s, y)[V(y) - V(s)],
\]

(1)

where \( q(s, y) \) is the transition rate from state \( s \) to \( y \)

**Foster-Lyapunov Proposition:** Let \( \phi \) be a time homogeneous, irreducible and continuous time Markov process and \( S \) be its state space. If there exists a finite set of states \( C \subset S \), a potential function \( V(s) : S \to (0, \infty) \) and some constants \( b > 0, \epsilon > 0 \), such that:

\[
\{ s : V(s) < K \} \text{ is finite } \forall K
\]

(2)

\[
DV(s) \leq -\epsilon + b1_{C}(s) \forall s \in S
\]

(3)

then \( \phi \) is positive recurrent. [26]

First note that the underlying Markov process for p2p network at hand is time homogeneous, irreducible and continuous time. To describe the particular potential function that satisfies Foster-Lyapunov proposition, we introduce the following notation for the system with \( k = 2 \) pieces:

- \( n_0 \): the number of peers with no piece
- \( n_1 \): the number of peers with piece 1 but not piece 2
- \( n_2 \): the number of peers with piece 2 but not piece 1
- \( s \): the number of peers in the system; note that \( s = n_0 + n_1 + n_2 \)

We choose the potential function as follows:

\[
V = (a^+)^2 + (b^+)^2 + d^2,
\]

(4)

where

\[
a = n_0 + \min(n_0, n_1) + c_1(n_1 - n_0)^+ - c_2n_2
\]

\[
b = n_0 + \min(n_0, n_2) + c_1(n_2 - n_0)^+ - c_2n_1
\]

\[
d = c_3n_0 + c_4n_1 + c_4n_2.
\]

Due to the length limitation, we omit the rest of the proof of the Main Theorem. The omitted part amounts to specifying a finite set of states \( C \) and a set of conditions on constants \( c_1, c_2, c_3, c_4, p \) for every \((\lambda, U_s, \mu)\), and then showing that under the group suppression protocol, the potential function \( V(s) \) in (4) and the specified \( C \) satisfies the Foster-Lyapunov Proposition for any \((\lambda > 0, U_s > 0, \mu > 0)\).

We now discuss the choice of the potential function \( V(s) \) in (4). As the complete reasoning behind the construction of the potential function is rather involved, consider a simple scenario where \( c_1 = 1, c_2 = 1, c_3 = 1, c_4 = 1 \), although this choice of constants violates the conditions imposed on these constants in the omitted part of the proof. Therefore, the elements of the potential function become:

\[
a = n_0 + n_1 - n_2
\]

\[
b = n_0 + n_2 - n_1
\]

\[
d = n_0 + n_1 + n_2.
\]

Then \( a \) is the *excess demand* for piece 2, i.e., the number of peers that demand piece 2 minus the number of peers that can supply it. Likewise, \( b \) is the excess demand for piece 1. The \( a \) and \( b \) terms in the Lyapunov function therefore penalize lopsided network states in which there is a dominant largest club. The \( d \) term evidently penalizes the overall network size.

**IV. Simulations for the Main Theorem and Conjecture 1**

This section simulates the unstructured p2p protocol and the group suppression protocol. The former will serve to show the existence of instability whereas the latter will serve to validate the Main Theorem and Conjecture 1.

The following rates and initial conditions apply to both protocols: The seed contacts the other peers according to a Poisson process of rate 2. Every incomplete peer’s contact process is according to an independent Poisson process of rate 1. The seed’s upload rate is chosen to be 2 instead of 1 because the seed does not have to download anything and, therefore, it is likely to allocate more bandwidth to the upload than the incomplete peers. Moreover, increasing the seed’s
Figure 1: The total number of peers in the network with the unstructured p2p protocol when $k = 2$ and $\lambda$ is varying.

Figure 2: The total number of peers in the network with the group suppression protocol when $k = 2$ and $\lambda$ is varying.

Figure 3: Comparison of the total number of peers to population of the largest club peers in the network with the unstructured p2p protocol when $k = 2$ and $\lambda = 4$.

Figure 4: Comparison of the total number of peers to population of the largest club peers in the network with the group suppression protocol when $k = 2$ and $\lambda = 24$.

upload rate makes it more challenging for the largest club to pull the network towards instability, which nevertheless will happen when the unstructured p2p protocol is simulated. The simulations start with 500 peers at $t = 0$ and all of them except the seed possess all the pieces of the file but the one common piece. Finally, peers are non-persistent.

While Fig. 1-3 and 5-7 represent the average of five individual realizations, Fig. 4 exhibits a single realization. Fig. 1-4 are intended to verify the Main Theorem and therefore the file is divided into only two pieces. Instability is seen in Fig. 1 where the unstructured p2p protocol is applied and $\lambda$ is varied. Since the arrival rate $\lambda$ is greater than the seed’s upload rate $U_s$, instability is expected and the result is consistent with the main theorem of Hajek and Zhu [4]. In Fig. 1, the networks roughly grow with rate $\lambda - U_s$, which is consistent with the understanding that almost no peer except the seed can help a peer exit the system by supplying the missing piece. In fact, Fig. 3 bears more direct evidence for the claim as the line that represents the number of the largest club peers closely trails the line that represents the total number of peers in the system.

The total number of peers is shown in Fig. 2 for varying $\lambda$ values when the group suppression protocol is applied. The networks for varying arrival rates, which are larger than the upload rate of the seed, recover from the severe piece imbalance and achieve stability. Larger arrival rates translate into more peers in steady state. Fig. 4 shows both the total number of peers and the population of the largest club peers when $\lambda = 24$. The increases in the population of the largest club peers apparently give rise to the sudden jumps in total number of peers. In the same vein, the downward trend in the population of the largest club is followed by the same trend in total number of peers. Between the growth and the decline of the largest club, the group suppression protocol cuts off the supply of pieces from the largest club to any peer who does not hold greater portion of the file, giving those peers more time to obtain and distribute the rarest piece without being absorbed into the largest club. Conjecture 1 in Section III states that any p2p network with the group suppression protocol is stable regardless of the number of the pieces the file is divided into. Fig. 5, where the unstructured p2p protocol is in effect,
provides experimental evidence that when $\lambda$ is greater than $U_s$, the missing piece syndrome phenomenon develops irrespective of the number of pieces $k$. Furthermore, the number of pieces in the file appears not to affect the slope of the total number of peers in the network because the total number of peers roughly grows with $\lambda - U_s$ per time unit for both $k = 48$ and $k = 6$ in Fig. 5. The group suppression protocol is shown in Fig. 6 and 7 for various $(\lambda, k)$ pairs. The stability follows what one could call a transient period and prevails permanently in all of different $(\lambda, k)$ pairs. Thus, Fig. 6-7 provide evidence that the stability conjecture holds true. After stability, the arguably most important measure of p2p system performance is the mean sojourn time. We compare different protocol’s sojourn time performances to that of the group suppression protocol in Table 1. One of the other protocols is due to Zhu and Hajek [6], which is referred to as the waiting protocol in Table 1. This protocol assumes the peers stay in the network on average long enough to upload one piece after they become seeds in our implementation of the waiting protocol. The second protocol in the comparison is Oğuz and Anantharam’s common chunk protocol in [10], which is simulated for both when the number of the contacts $m$ that one club peers make is three and five. The final protocol simulated is Reittu’s forced Friedman protocol [8], which requires every peer to sample three other peers and allows the peer to download a random piece from the set of pieces that show up exactly once in the aggregate piece profile of these three sampled peers. If the set turns out to be empty, the peer skips without downloading any piece. We obtained 100 individual realizations for each protocol where the simulations started with 499 peers holding all pieces but
Table I: Mean sojourn times for different protocols when $\lambda$ and $k$ are varied

| Protocol Name                                      | Mean sojourn time for $k = 25$ and $\lambda = 4$ | Mean sojourn time for $k = 50$ and $\lambda = 4$ | Mean sojourn time for $k = 25$ and $\lambda = 12$ | Mean sojourn time for $k = 50$ and $\lambda = 12$ |
|---------------------------------------------------|--------------------------------------------------|--------------------------------------------------|--------------------------------------------------|--------------------------------------------------|
| The group suppression protocol                     | 29.29                                            | 54.69                                            | 28.72                                            | 54.39                                            |
| The waiting protocol                               | 29.32                                            | 55.07                                            | 29.01                                            | 54.81                                            |
| The forced Friedman protocol                       | 30.69                                            | 55.86                                            | 30.46                                            | 55.71                                            |
| The common chunk protocol for $m=5$                | 30.49                                            | 57.71                                            | 30.29                                            | 57.52                                            |
| The common chunk protocol for $m=3$                | 36.88                                            | 67.37                                            | 36.86                                            | 67.21                                            |

The group suppression protocol outperforms all the other protocols for varying $k$ and $\lambda$ values. The group suppression protocol, the forced Friedman protocol and the waiting protocol seem to fare slightly better than the common chunk protocols as $k$ is increasing.

V. Models of the Modified BitTorrent Protocols and Their Simulations

We assume throughout the section that the reader is familiar with BitTorrent’s official protocol and its two fundamental algorithms, namely the rarest first and the unchoking algorithms [27]. A simplified yet rather loyal implementation of the official BitTorrent protocol, which lends itself to simulations and which we call the BitTorrent-like protocol, is introduced. We then construct the BitTorrent-like protocol with group suppression by incorporating the features of the group suppression protocol into the BitTorrent-like protocol. The section is concluded with the simulation results of the BitTorrent-like protocol and the BitTorrent-like protocol with group suppression.

We now define the BitTorrent-like protocol. The original rarest first and the unchoking algorithms of the official BitTorrent protocol are preserved in the BitTorrent-like protocol although a number of simplifications were made in order to render large scale simulations practical. First, all the upload times, arrival times and departure times are discretized and synchronous. A fixed number of peers arrives into the system every 10 seconds rather than having random individual arrivals. Also, the upload of any piece takes every incomplete peer exactly 10 seconds and they have four upload slots, meaning that they can upload at most four pieces simultaneously. Every incomplete peer’s download and upload speeds are upperbounded by 40 pieces and 4 pieces per 10 seconds, respectively, in order to mimic the incomplete peer’s tendency to allow much more bandwidth for the download than for upload. The seed, too, commands 4 upload slots but it finishes uploading a piece in exactly 2 seconds, which makes the seed’s maximum upload speed five times an incomplete peer’s maximum upload speed. In other words, the seed’s maximum upload speed is 2 pieces per second and an incomplete peer’s maximum upload speed is 0.4 pieces per second.

By the following modifications to some features of the original unchoking algorithm, we construct the BitTorrent-like protocol with group suppression: First, the seed ranks all interested neighbor peers in an ascending order of the number of pieces they hold. In case of a tie, the one with the larger upload ratio to the whole network is favored by the seed. It should be noted that this policy change does not add any complexity to the network as the seed already knows the piece profile of every neighbor peer. An incomplete peer first determines whether it belongs to the largest club or not based on the collection of its neighbors’ piece profiles. If it does not turn out to be a member of the largest club, then the unchoking algorithm in original BitTorrent protocol proceeds. If it indeed belongs to the largest club, the incomplete peer creates the list according to the original unchoking algorithm of BitTorrent and then updates the list by removing peers that do not hold greater portion of the file than this peer. It then unchokes based on the updated list. Again, one should notice that for an incomplete peer to determine the largest club does not require any additional knowledge. Thus, the group suppression fits more naturally into BitTorrent than it does into an unstructured p2p network.

Before we discuss the simulation results, note that while Fig. 8-10 demonstrate the average of five individual realizations, Fig. 11-12 represent single realizations. The initial state of the simulation for the BitTorrent-like protocol includes one fixed seed, 494 peers holding all the pieces but the last piece and 5 peers possessing only the last piece; so there are 500 peers at $t = 0$. Note that these 5 peers act as an impediment to the largest club’s gravitational effect on the new peers not only by supplying the last piece to the largest club and making them depart, but also by giving the last piece to new peers, which guarantees that they will not be absorbed into the largest club. They, thus, have a stabilizing effect. To show the occurrence of the missing piece syndrome for the BitTorrent-like protocol in simulations, the constant number of arrivals per 10 second blocks is chosen from 30, 40, 60 and 80, which is equivalent to saying $\lambda$ takes on values of 3, 4, 6 and 8 arrivals per second. Note that all values of $\lambda$ are greater than the seed’s maximum upload speed of 2 pieces per second. With this initial state, Fig. 8 shows that the BitTorrent-like protocol falls victim to the missing piece syndrome for all four $\lambda$ values when $k$ is fixed at both 12 and 48. The total number of peers in Fig. 8 roughly grows with rate $\lambda - U_s$ where $U_s$ is naturally interpreted as the
maximum upload speed of the seed per second. In conclusion, the BitTorrent-like protocol can be unstable when the arrival rate of new peers overwhelms the maximum upload speed of the seed, which in turn suggests that the original BitTorrent protocol may suffer from the missing piece syndrome if all peers choose to be non-persistent. We suspect that the real-life BitTorrent networks enjoys stability only because of the presence of persistent peers in those networks.

Aside from the occurrence of instability, there are two more observations worth mentioning regarding the BitTorrent-like protocol although we do not include the corresponding figures. First, increasing the number of pieces $k$ in the file seems to improve the stability for fixed upload rate of incomplete peers. But, if the upload rate of incomplete peers is made suitably large, then the increased upload capacity of the largest club would overcome the stability enhancing effect of increasing the number of pieces in the file. Second, unlike the unstructured p2p networks, increasing the arrival rate of peers beyond some point paradoxically tips the balance of the BitTorrent-like protocol into stability. This is more likely to be a specific result of discretized nature of arrivals and, therefore, this counter-intuitive observation of the BitTorrent-like network may not hold true for the continuous time arrivals that actually take place in real life BitTorrent.

For various $(\lambda, k)$ pairs, we provide the simulation results on the BitTorrent-like protocol with group suppression through Fig. 9-12. We modify the initial state by starting with 499 peers that hold all the pieces but the last piece and one fixed seed. Note that this tends to decrease the stability of the system. The stable trajectories of the total number of peers exhibited in Fig. 9-12 provides experimental support for the idea that the BitTorrent-like protocol with group suppression, as opposed
to the BitTorrent-like protocol, is effective against the missing piece syndrome. Fig. 9-10 reveals the following patterns: First, increasing \( k \) with fixed \( \lambda \) results in more peers in steady-state. Similarly increasing \( \lambda \) with fixed \( k \) ends up with more peers in steady-state. The spikes in the total number of peers manifest themselves more intensely when \( k \) is smaller while they are insensitive to \( \lambda \) especially when \( k \) is large. Since \( k \) tends to be large in actual BitTorrent networks, the spikes should not be of any concern when it comes to stability. Yet, better understanding of why such spikes are observed could shed light on how the group suppression protocol achieves stability. It can be seen from examining Fig. 11-12 that a buildup of the largest club in the network triggers a spike of considerable magnitude in the total number of peers. Then it can be seen in these figures that the growth of the largest club, thanks to the group suppression protocol, cannot gain further momentum and soon the largest club starts to fade away, paving the way for a steep decline in the total number of peers. Thus, the simulations indicate that the BitTorrent-like protocol with group suppression is stable even when the number of pieces is greater than two.

**ACKNOWLEDGMENT**

The authors wish to thank Ebad Ahmed for his contributions to the early stages of this project.

**REFERENCES**

[1] “Cisco visual networking index: Forecast and methodology, 2015–2020.” Cisco Systems, June 2016. [Online]. Available: http://www.cisco.com/c/dam/en/us/solutions/collateral/service-provider/visual-networking-index-vni-complete-white-paper-c11-481360.pdf

[2] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, “Fog computing and its role in the internet of things,” in Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, ser. MCC ’12. New York, NY, USA: ACM, 2012, pp. 13–16. [Online]. Available: http://doi.acm.org/10.1145/2342509.2342513

[3] L. M. Vaquero and L. Rodero-Merino, “Finding your way in the fog: Towards a comprehensive definition of fog computing,” SIGCOMM Comput. Commun., vol. 44, no. 5, pp. 27–32, Oct. 2014. [Online]. Available: http://doi.acm.org/10.1145/2677046.2677052

[4] B. Hajek and J. Zhu, “The missing piece syndrome in peer-to-peer communication,” in 2010 IEEE International Symposium on Information Theory, June 2010, pp. 1748–1752.

[5] F. Mathieu and J. Reynier, “Missing piece issue and upload strategies in flashcrawls and p2p-assisted filesharing,” in Advanced Int’l Conference on Telecommunications and Int’l Conference on Internet and Web Applications and Services (ACT/ICW’06), Feb 2006, pp. 112–112.

[6] J. Zhu and B. Hajek, “Stability of a peer-to-peer communication system,” IEEE Transactions on Information Theory, vol. 58, no. 7, pp. 4693–4713, July 2012.

[7] R. E. Bohn and J. E. Short, “How much information? 2009 report on American consumers,” Global Information Industry Center, University of California, San Diego, Dec. 2009. [Online]. Available: http://hmi.ucsd.edu/pdf/HMI_2009_ConsumerReport_Dec9_2009.pdf

[8] H. Reitt, “A stable random-contract algorithm for peer-to-peer file sharing,” in Proceedings of the 4th IFIP TC 6 International Workshop on Self-Organizing Systems, ser. IWSOS ’09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 185–192. [Online]. Available: http://doi.acm.org/10.1145/1007978-3-642-10865-5_16

[9] I. Norros, H. Reittu, and T. Eroa, “On the stability of two-chunk file-sharing systems,” Queueing Syst. Theory Appl., vol. 67, no. 3, pp. 183–206, Mar. 2011. [Online]. Available: http://dx.doi.org/10.1007/s11134-011-9209-2

[10] B. Olijz, V. Anantharam, and I. Norros, “Stable distributed p2p protocols based on random peer sampling,” IEEE/ACM Transactions on Networking, vol. 23, no. 5, pp. 1444–1456, Oct 2015.

[11] B. Zhang, S. C. Borst, and M. I. Reiman, “Optimal server scheduling in hybrid p2p networks,” Perform. Eval., vol. 67, no. 11, pp. 1259–1272, Nov. 2010. [Online]. Available: http://dx.doi.org/10.1016/j.peva.2010.08.017

[12] E. de Souza e Silva, R. M. M. Leão, D. S. Menasché, and D. Towsley, “Scalability issues in P2P systems,” CoRR, vol. abs/1405.6228, 2014. [Online]. Available: http://arxiv.org/abs/1405.6228

[13] L. Massoulié and M. Vojnović, “Coupon replication systems,” IEEE/ACM Trans. Netw., vol. 16, no. 3, pp. 603–616, Jun. 2008. [Online]. Available: http://dx.doi.org/10.1109/TNET.2007.903992

[14] D. Qiu and R. Srikannt, “Modeling and performance analysis of bittorrent-like peer-to-peer networks,” in Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, ser. SIGCOMM ’04. New York, NY, USA: ACM, 2004, pp. 367–378. [Online]. Available: http://doi.acm.org/10.1145/1015467.1015508

[15] D. Qiu and W. Sun, “Global stability of peer-to-peer file sharing systems,” Comput. Commun., vol. 31, no. 2, pp. 212–219, Feb. 2008. [Online]. Available: http://doi.acm.org/10.1016/j.comcom.2007.08.012

[16] Y. Tian, D. Wu, and K. W. Ng, “Modeling, analysis and improvement for bittorrent-like file sharing networks,” in Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications, April 2006, pp. 1–11.

[17] B. Fan, D. m. Chiu, and J. C. s. Lui, “Stochastic analysis and file availability enhancement for bt-like file sharing systems,” in 200614th IEEE International Workshop on Quality of Service, June 2006, pp. 30–39.

[18] J. Zhu, S. Ioannidis, N. Hegde, and L. Massoulié, “Stable and scalable universal swarms,” in Proceedings of the 2013 ACM Symposium on Principles of Distributed Computing, ser. PODC ’13. New York, NY, USA: ACM, 2013, pp. 260–269. [Online]. Available: http://doi.acm.org/10.1145/2484239.2484272

[19] X. Zhou, S. Ioannidis, and L. Massoulié, “On the stability and optimality of universal swarms,” in Proceedings of the ACM SIGMETRICS Joint International Conference on Measurement and Modeling of Computer Systems, ser. SIGMETRICS ’11. New York, NY, USA: ACM, 2011, pp. 341–352. [Online]. Available: http://doi.acm.org/10.1145/1993744.1993779

[20] D. S. Menasché, A. A. de A. Rocha, B. Li, D. Towsley, and A. Venkataramani, “Content availability and bundling in swarming systems,” IEEE/ACM Transactions on Networking, vol. 21, no. 2, pp. 580–593, April 2013.

[21] H. Zhang, S. Vasudevan, R. Li, and D. Towsley, “Coalitions improve performance in data swarming systems,” IEEE/ACM Transactions on Networking, vol. 23, no. 6, pp. 1790–1804, Dec 2015.

[22] A. Ferragut and F. Paganini, “Fluid models of population and download progress in p2p networks,” IEEE Transactions on Control of Network Systems, vol. 3, no. 1, pp. 34–45, March 2016.

[23] R. L. Xia and J. K. Muppala, “A survey of bittorrent performance,” IEEE Communications Surveys Tutorials, vol. 12, no. 2, pp. 140–158, Second 2010.

[24] C. Westphal, “A stable fountain code mechanism for peer-to-peer content distribution,” in IEEE INFOCOM 2014 - IEEE Conference on Computer Communications, April 2014, pp. 2571–2579.

[25] E. de Souza e Silva, R. M. M. Leão, D. Sadoc Menasché, and A. A. de A. Rocha, “On the interplay between content popularity and performance in p2p systems,” in Proceedings of the 10th International Conference on Quantitative Evaluation of Systems, ser. QEST’13. Berlin, Heidelberg: Springer-Verlag, 2013, pp. 3–21. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-40196-1_2

[26] B. Hajek, Random Processes for Engineers. Cambridge, UK: Cambridge University Press, 2015, ch. 6, p. 194.

[27] B. Cohen, “Incentives build robustness in bittorrent,” in Workshop on Economics of Peer-to-Peer systems, vol. 6, 2003, pp. 68–72.