IEICE TRANS. COMMUN., VOL.E91–B, NO.12 DECEMBER 2008

PAPER Special Section on Peer to Peer Networking Technology

An Analytical and Experimental Study of Super-Seeding in BitTorrent-Like P2P Networks*

Zhijia CHEN[†a)], ***Student Member***, **Chuang LIN**[†], **Yang CHEN**[†], **Vaibhav NIVARGI**[††], ***and*** **Pei CAO**[††], ***Nonmembers***

SUMMARY With the popularity of BitTorrent-like P2P applications, improving its performance has been an active research area. Super-seeding, a special upload policy for the initial seeder, improves the efficiency in producing multiple seeds and reduces the uploading bytes of content initiators, thus being highly expected as a promising solution for improving downloading performance while decreasing uploading cost. However, the overall impacts of super seeding upon BitTorrent performance remain a question and have not been analyzed so far in literature. In this paper, we present an analytical and experimental study over the performance of super-seeding scheme. We attempt to answer the following questions: whether and how much super-seeding saves uploading cost, whether the overall downloading time is decreased by super-seeding, and in which circumstances super-seeding performs worse. Based on the seeding process, our analytical study gives formulas on the new piece distribution time, average downloading time and minimum distribution time for heterogeneous P2P file distribution system with super-seeding. Robust evidence supporting the use (or not) of super-seeding is given based on our worldwide Internet experiments over wide distribution of 250 PlanetLab nodes. With a well-designed experimental scenario, we study the overall download time and upload cost of super seeding scheme under varying seed bandwidth and peer behavior. Results show that super-seeding can save an upload ratio of 20% and does help speeding up swarms in certain modes. Tentative conclusions about the effectiveness of super-seeding and its optimal working circumstances are given with inside mechanism analyzed and negative factor identified. Our work not only provides reference for the potential adoption of super-seeding in BitTorrent and other P2P applications, but also much insights for the tussle of enhancing of Quality of Experience (QoE) and saving cost for a large-scale BitTorrent-like P2P commercial application.

key words: *super-seeding, quality of experience, performance evaluation*

1. Introduction

In the popular BitTorrent-like P2P file sharing application, the seeds and seeding schemes play a significant role in file distribution performance. Peers that provide a complete file for distribution are called seeders, and particularly, the initial seeder who provides the initial copy, counts much in BitTorrent performance. How a seed upload pieces to peers, i.e. the seeding scheme, can impact the speed of creating other seeds and the whole downloading Quality of Experience (QoE) of end users. Meanwhile, when BitTorrent is adopted by content providers who have to pay for upload bandwidth by the byte, seeding schemes also directly affect the commercial cost. The success a large-scale commercial BitTorrent application would thus largely rely on the improvement of peer performance and the decrease of seed cost in an efficient seeding scheme.

To meet with those challenges in BitTorrent seeding schemes, Super-seeding, a special seeding scheme different from the default one, was first introduced in the BitTornado [4] client in mid 2003, aiming to help the initial seeder with limited bandwidth to "pump up" a large torrent (all pieces of file for distribution), thus reducing the amount of data it needs to upload to spawn new seeds. Similar feature is also implemented in μTorrent [6] and is called Initial Seeding due to its special application for content initiator.

By inducing peers into taking only the rarest data and reducing the amount of redundant data sent, super-seeding scheme serves as a promising solution to improve seeding efficiency and decrease seeder uploading cost. However, the overall benefits of super seeding for the whole P2P file distribution remain to be a question. Since the configurations of peers and their upload capacities vary widely, the overall performance may change significantly under different circumstances. Despite the benefits claimed by BitTornado, the adoption of super-seeding has not extended to most BitTorrent implementations and other P2P networks, though highly expected.

Although there has been many literature on the analytical and experimental performance evaluation of BitTorrent [1], [8]–[10], the super-seeding of BitTorrent has not been analyzed so far in the literature (Only anecdotal evidence is available). Therefore, a performance study, both analytical and experimental, that provides robust evidence to explore the use (or not) of super-seeding, would be rather necessary and valuable.

In this paper, we conducted a comprehensive study over the impact of super-seeding in exploring: 1) whether and how much super-seeding would save uploading cost; 2) whether the overall download time can be decreased by super-seeding; 3) in which schemes super-seeding works/does not work. To fully explore the problem, we utilize a performance evaluation approach based on analytical study and large-scale experiments. Also, a QoE-targeted and cost-saving analysis are presented to meet the needs of a large-scale commercial BitTorrent application. Upon our extensive measurements and trace analysis from the experiment in around 250 planet-lab nodes, our study gives

Manuscript received March 22, 2008.
Manuscript revised July 22, 2008.

[†]The authors are with Tsinghua University, China.
[††]The authors are with Stanford University, USA.

*This work is supported by 973 Program of China (No.2007CB310806) and the NSF of China (No.60873254, No.90412012, No.60473087 and No.60703052). Partial results appeared in conference paper in ICC'08.

a) E-mail: zj-chen05@mails.tsinghua.edu.cn

DOI: 10.1093/ietcom/e91–b.12.3842

out tentative conclusions about the effectiveness of super-seeding and its optimal working circumstances, and explains the mechanisms and tussles inside. In addition, during exploring this new topic, interesting finding provides us insights about (Content Distribution Networks) CDN-assisted P2P content distribution, peer share-ratio improvement, performance in heterogeneous network with selfish user behavior, etc, for promising further research.

The rest paper is organized as follows: Sect. 2 analyzes the mechanism and effects of the super-seeding through an analytical study; Sect. 3 designs and analyzes the experimental scenario; Sect. 4 illustrates experiments with varying seed bandwidth; Sect. 5 presents experiments varying peer behavior in heterogeneous network; Sect. 6 gives out the conclusion on super-seeding impact and highlights our contribution.

2. Analysis of Super-Seeding

2.1 Mechanism and Performance Analysis

Inside the super-seeding mechanism, its primary goal is to minimize the cost for seeders to upload. When a seeder, which has initial content to distribute, enters the super-seed mode, it pretends to be a normal client without any data, until some peer connects to it. Then the seeder informs the requesting peer that it has received one piece of data, which the other peer can now download. When the peer has finished downloading that piece, it is not able to download new pieces from the seed until the seeder finds that the piece it has just sent is present on at least one other peer [4]. This way, the client does not have access to any other pieces before it distributes what it has received, and therefore does not waste the seed's bandwidth to upload redundant data.

According to the key concept in super-seeding mechanism, the file distribution in super-seeding can be roughly depicted as Fig. 1 based on Petri Nets. Petri Net, as a formal tool, is well suited to describe discrete processes and can efficiently analyze the organizational structures and dynamic behaviors in a system. It consists of *places*, *transitions*, and *arcs* that connect them. *Places* can contain tokens, the current state of the modeled system. The marking is given by the number of tokens in each place. *Transitions* are active components. A transition of a Petri net may fire whenever there is a token at its input arcs, and it then consumes these tokens, and places tokens at its output arcs. They model activities which can occur, thus changing the state of the system [11].

As is shown in Fig. 1, the seed (place 1) has n tokens (which depicts n pieces of the file) to distribute. For a new piece i (one token in place 1), it will take time T_1 to distribute to the peer (Place 2). After the peer has the token, it will take time T_2 to distribute to other peers so that the token can come to place 3. As long as place 1 and place 3 both have tokens, the transition can be fired and place 2 can get another new token (new piece) from the seed (this shows the super-seeding mechanism that a peer has to distribute the piece it receives from the server before it gets another new piece).

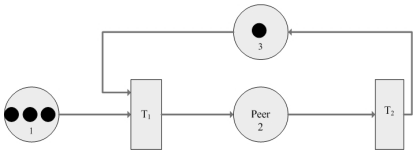


Fig. 1 Petri net model of super-seeding file distribution.

Suppose there is a new file with the size of S for distribution and it is divided into N pieces. Only one seed has the file and will distribute to n peers in super-seeding scheme. Those peers share the pieces they have between each other with the traditional "tit-for-tat" Choking/Unchoking and "rarest first replication" piece selection scheme. Other parameters are defined in Table 1.

Single Piece Distribution Time: We first consider the distribution of one new piece to all peers. As is shown by Fig. 2, for every new piece distribution, it takes time $T_1 = \frac{S}{N*d_p}$ to distribute from the seed to up to $a_1 = k = \left\lfloor \frac{U_s}{d_p} \right\rfloor$ peers simultaneously. Then those peers will take time $T_2 = \frac{S}{N*d_p}$ to share this new piece with another $m = \frac{k*u_p}{d_p}$ peers. Suppose after m times of piece distribution from peers, the number of peers a_{m+1}reaches the total number of peers: n. We have $a_{m+1} = a_m + \frac{a_m \times u_p}{d_p}$. We then can compute the geometric progression (sequence) as:

$$a_{m+1} = a_1 \times \left(1 + \frac{u_p}{d_p}\right)^m = n \tag{1}$$

The total time for one new piece distribution around the whole network is:

$$T_{piece} = T_1 + mT_2 = \frac{S}{N * d_p}\left(1 + \log_{(1+\frac{u_p}{d_p})}^{\frac{nd_p}{U_S}}\right) \tag{2}$$

In optimal situation, the seed only has to upload new pieces one time and take time T_{piece} to fully distribute to every peers. Peers will take at least time $T_2 + T_1$between it gets two new pieces. Please note that the total piece distribution will be much quicker than n independent T_{piece} time distribution; actually they are not independent, several piece downloading initiates at the same time randomly from either peers or the seed. Thus $T_{max} \geq N \times T_{piece} = \frac{S}{d_p}\left(1 + \log_{(1+\frac{u_p}{d_p})}^{\frac{nd_p}{U_S}}\right)$

Table 1 Parameter for file distribution analysis.

Parameter	Definition
S	Size of the file for distribution
N	Number of pieces of the file
n	Number of peers in the network
U_s	Upload Bandwidth of the seed
u_p	Average upload rate of peers
d_p	Average Aggregate download rate of peers
T_1	Time for one piece distribution from seed to peer
T_2	Time for one piece distribution from peer to peer

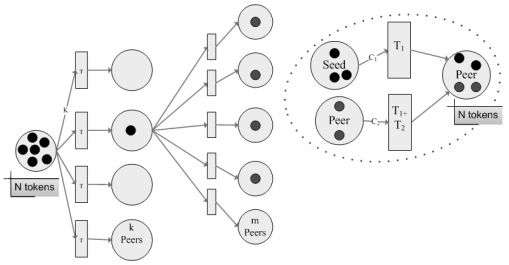


Fig. 2 Process and final state of file distribution.

could be treated as one upper bound of the downloading time (equation achieved when all new pieces download in sequence independently).

Average Downloading Time: Next, we consider the average distribution for all pieces. As is shown by the right part of Fig. 2, in the final state, the peer place has downloaded all the pieces, with probability c1 from seed and with probability c2 from other peers. Since the seed can serve up to k peers, the probability that the peer can get a new piece from the seed is: $c_1 = k/n$. The upload bandwidth of peer i is u_i and the download rate is d_i. Peers that get content from another peer would first wait content to be transmitted from the server and then between the peers. So the total time for a peer to have all pieces, also the average downloading completion time is:

$$\begin{aligned} T_{ave.} &= \sum_{i=1}^{n} \{c_1 N T_2(i) + c_2 N[T_1(i) + T_2(i)]\}/n \\ &= c_1 * N * \frac{S}{N * d_p} + (1 - c_1) * N * \left(\frac{S}{N * d_p} + \frac{S}{N * d_p}\right) \\ &= \left(2 - \frac{\left\lfloor \frac{U_s}{d_p} \right\rfloor}{n}\right) * \frac{S}{d_p} \end{aligned} \tag{3}$$

Where, d_p, the average Aggregate downloading rate of the peers (also the *maximum achievable downloading rate*) can be computed as follows:

$$d_p = \frac{U_S}{k} * \frac{k}{n} + \sum_{i=1}^{n} \left(\frac{u_i}{j} * \frac{j}{n}\right) = \frac{U_S + \sum_{i=1}^{n} u_i}{n} \tag{4}$$

Note k, j is the maximum peers the server/one peer can support and k/n is the probability that one peer can get downloading from the server (in Steady state, the total uploading bandwidth is equal to the total downloading bandwidth).

Both Formula (3) and (4) show that the most important parameter in file distribution time is the upload bandwidth of the seed, U_s, and the average aggregate downloading rate, which is limited by the uploading rate of the peers. This explains why we would vary the relationships of seed upload bandwidth and the peer upload bandwidth in experimenting with the super-seeding performance (see Sect. 3 for more details).

With Peer Heterogeneity: In heterogeneous network, peers of different uploading and downloading capacity join the system. Since the downloading rate is seldom the boundary [14], we assume peers still have the same average downloading rate, and put our focus on the uploading capacity. If we adopt two-class model[32] of tier-1 peer with upload rate $u_1 > d$ and tier-2 peer with $u_2 < d$, and they come at the probability λ_1, λ_2, where $\lambda_1 + \lambda_2 = 1$. Since $d_p = \frac{U_s + n\lambda_1 u_1 + n\lambda_2 u_2}{n}$, the more tier-1 peers are, the higher is d_p, thus the lower of the average downloading time T_ave. And the scale that could be supported by the system capacity is $N = n + \frac{U_s + n(\lambda_1 u_1 + \lambda_2 u_2 - \lambda_1 d - \lambda_2 d)}{d} = n + \frac{U_s + n\lambda_1(u_1 - d) - n\lambda_2(d - u_2)}{d}$, the higher λ_1 is, the bigger is N. With the increase of peer scale n, the system scale increases, demonstrating the high scalability of P2P systems.

Lower Bound on Downloading Time: Considering that some assumption of the model cannot completely reflect the dynamics of the P2P system, we further extend above analysis by analyzing the minimum downloading time for all peers. Based on paper [12], the total download time with super-seeding downloading is also subject to following boundaries: 1) the time it takes for the peer with lowest download rate to complete, i.e. S/d_{min}; 2) the quickest time for the seed to produce a new file, i.e. S/U_s. 3) the time it takes for total of $n * S$ bits to be distributed with the maximum system upload bandwidth of $U_s + \sum_{i=1}^{n} u_i$. So we have the minimum distribution time for the general heterogeneous P2P File distribution system as:

$$T_{min} = \frac{S}{min(d_{min}, U_s, \frac{U_s + \sum_{i=1}^{n} u_i}{n})} \tag{5}$$

If we consider the downloading time for most peers, i.e. ruling out the last 10% slow downloaders like our latter experiment in Sect. 3, the value of the lower bound is then decided by the relationships of U_s and $\frac{U_s + \sum_{i=1}^{n} u_i}{n}$. In case of $\frac{U_s}{u_p} \geq \frac{n}{n-1}$, where $u_p = \frac{(U_s + \sum_{i=1}^{n} u_i)}{n}$, we have the lower bound time $T_{min} = \frac{S}{\frac{(U_s + \sum_{i=1}^{n} u_i)}{n}}$ (Paper[21] shows this lower bound is achievable). Since $n/(n-1) \approx 1$, this explains why experiment with upload bandwidth of seed above, or equal, or below that of the peer with 1:1 ratio.

To verify the correctness of above analysis, we compare the theoretical value with our experiments running on Planetlab. The experimental data was drawn from one set of experiments in our latter sections, in the scenario that a file with size of 55.51M was distributed from one source seed with upload rate of 100 kb/s to 150 successful downloaders with an upload limit of 50 kb/s in both mode of regular seeding and super seeding. Formula 3 forms the average downloading value and formula 5 forms the lower bound. As is shown by Fig. 3, our theoretical lower bound of downloading time well serves as the bound in experiments and the average value from analysis roughly meets the varying downloading time of peers. We can also compute the upper bound of downloading time from formula 2, $N * T_{piece} = 117.6$, which is bigger than the downloading time of all peers in

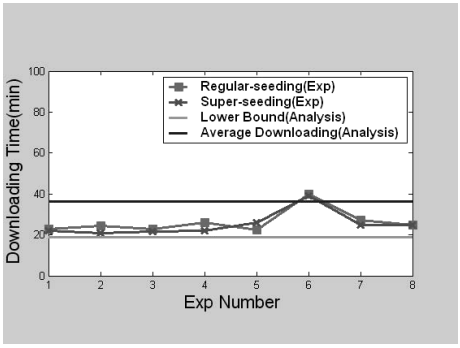


Fig. 3 Experimental result vs. analysis.

experiments.

2.2 Impact Analysis

What is clear?: The authors of BitTornado report that a standard seed might have to upload 150% to 200% of the total size of a torrent before other clients become seeds, while the super-seeding seeder may only have to upload 105% of the data [4]. Super-seeding saves uploading cost, but its actual saving ratio in varied circumstances need to be quantified. In addition, it's clear that, if super-seed mode is adopted all over the network, it will limit the selection of pieces that a client can download. This will significantly decrease the overall downloading speed, and thus is not recommended.

What is not clear?: For the overall influence over all peers, one observation is that the super-seeding scheme results in much higher seeding efficiency, by making peers into taking only the rarest data and reducing the amount of redundant data sent. The whole peers can benefit from super-seeding scheme for creating multiple seeds in a more efficient manner. With the efficient and proper distribution of data piece, once the initial seeder uploads one complete copy of the file, multiple new seeds will emerge rather quickly, thus boosting the overall uploading speed of the swarm (all peers, including seeders, sharing the torrent).

However, this does not mean that the uploading of the whole torrent will take less time. First, the time it takes the super seeder to produce the first completion of a downloader is greatly limited by the upload rate of the peers connected to the downloader. Second, with varied peer behavior and rate, it is too early to say that peers can download faster due to the seeding efficiency in super-seeding. Complicate factors decide the downloading performance, whether and when super-seeding can perform at its advantage need to be studied.

3. Experimental Scenario

Having analyzed the impacts of super seeding theoretically and identified the key parameter in influencing the downloading time with super-seeding, in this part, we move to the experimental part with the guidance of our above theoretical findings. To evaluate the performance of super-seeding scheme, we deployed both the regular Bittorent and super-seeding featured BitTornado version in PlanetLab, a world-wide platform for performing Internet measurements [5], [16]. Around 250 planetlab nodes with a wide range of sites all over the world were picked (mainly from Europe, north American and China based on current Planetlab distribution), and tested in the period from April 2007 to Sept. 2007.

Consider the dynamics of Planetlab nodes, the testing time were randomly chosen during those months and the test results were averaged in same time period, i.e. day and night. We launched our tracker and seeder in two planetlab nodes at Stanford University, from which they track and seed other nodes respectively. Same with the settings in BitTornado's specification for super-seeding, the node that functions at super seeding scheme is only the initial content provider. All peers would wait for the unique seeder to provide the initial content for sharing. Two files downloading are tested, with a file size of 55.51M and 697.89M respectively. Three mode are measured and compared, i.e., the upload bandwidth of the seeder, is ***above, or equal, or below*** that of the peers. Peers download at its own rate and behavior to reflect the real heterogeneous network.

Based on the key concern in large-scale BitTorrent application, we define two metrics for the evaluation, i.e. download time and upload cost. Considering the dynamics of the network and the heterogeneity of the peers, we mark the download time when 50%, 90% of the peers finish their downloading.

The second metric measures the seed upload cost: we define Seed Upload Ratio as, S_{ur} = (size of total uploading)/(size of content file).

Why we vary the upload bandwidth of peers and seeders?

Intuitively, super-seeding changes the behavior of the initial seeder, and thus its effectiveness critically depends on the relationships between the seeder and all other peers. Our above analytical study demonstrates the key role that upload rate of seed and peers play in deciding the downloading time (note in our experimental scenario, the only server and content provider is the initial seeder).

Why We don't Run Experiments with Concurrent Seeds?

As paper [2] formerly discussed, while we run two independent Bittorrent clients downloading at the same machine, there tends to be great difference in downloading time due to the random selection of neighbors. Similarly, we conducted several rounds of experiment to compare the hybrid mode in the same machine by concurrently making the seeder running two independent processes, one in regular seeding, the other in super-seeding, and serving the same other peers respectively.

As Fig. 5 shows, in this concurrent seeds, it seems regular seeding would compete resource with super-seeding and finally super seeding suffers and takes more time to complete. In comparison, as is shown in Fig. 6, when we implement those two modes independently, the super-seeding

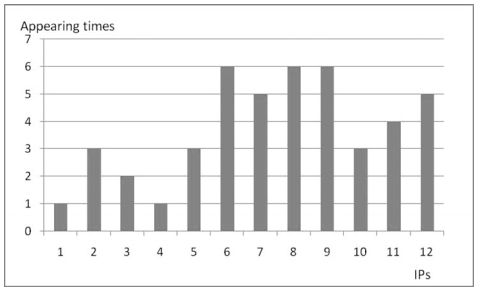


Fig. 4 Slow nodes in experiments.

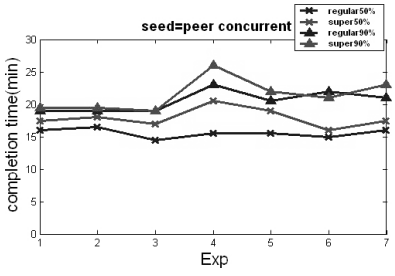


Fig. 5 Concurrent seeding.

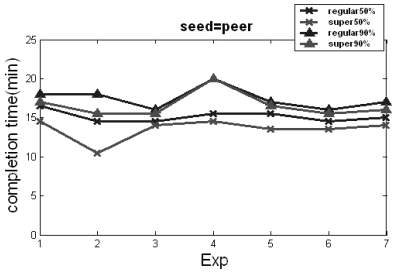


Fig. 6 Independent seeding.

scheme can download faster than regular-seeding. We will further justify that this trend is rather stable in this case in later sections. So for the fairness of the experiment and to achieve better performance of super-seeding, we run the two modes independently and avoid concurrent seeds in the same machine.

Why We Use 90% Completion Time as the Metric?

As we observed in most of our experiments, around 10% of peers cannot finish downloading in testing time. In selfish mode (see Sect. 5), this data can go as much as 50% sometimes. If normally a peer can finish downloading within t minutes, we would regard it unacceptable to complete in more than $300\% * t$ minutes, otherwise, the Quality of Experience (QoE) of end users cannot be guaranteed. This is would be a crucial problem if BitTorrent goes to a large-scale commercial application.

To study this issue, we randomly choose one experiment with around 10% unfinished nodes and marked the IP of uncompleted nodes. We then compare those IPs in list of unfinished nodes in another 6 experiments and counted their appearing frequency. As Fig. 4 shows, most nodes appear more than 3 times in our 7 experiments as unfinished nodes, and some even appear 6 out of 7 times. It should be the network heterogeneity and dynamics that affect their completion. We also notice some nodes only appeared one time as unfinished nodes, but in all other experiments, completed their downloading rather smoothly. There is little doubt that those peers will finally complete if enough time is given, but the problem is how long the user can bear with. Further experiment [2] shows that the reasons for higher turn-over in close peers for the slower client are mainly events outside its control, including peers that are unstable hosts and disconnect randomly, peers that finish its download and exit, or peers that simply do not stay in the P2P network for long.

As a solution out, a CDN (Content Distribution Network) assisted BitTorrent scheme may serve as a backup solution to ensure the QoE of every user. To guarantee user QoE and improve system scalability, based on our success in a CDN and P2P hybrid streaming system [13], we can build a CDN-assisted P2P structure for content distribution. With the assistance of CDN, strategically deployed CDN servers around the Internet enable end users to obtain the content from one of the nearby servers to reduce distribution time and overall network congestion. All those carefully-deployed servers form a Trustworthy and Controllable overlay network, through which servers are used to provide initial content, guide downloading traffic to achieve overall traffic optimization, and conduct AAA and key distribution. As this is not the main target of this paper, we do not go to more details here. Yet this would surely be a promising area to explore.

4. Experiments with Varying Seed Bandwidth

We study the super-seeding impact in modes of: upload bandwidth of seed above, or equal, or below that of the peer. Two key performance metrics are considered: downloading time and seed upload ratio.

4.1 Upload Bandwidth Seed=Peer

As the Fig. 7 shows, in the randomly selected 12 sets of experiments during the testing period, with the super-seeding scheme, the whole network would finish downloading faster than the regular mode. Further experiments show that, it is rather stable that super seeding helps speed up the whole swarm when upload rate seed=peer, no matter in day time or night time.

To be more specific, we also draw out the whole process of a full downloading in Fig. 8, which also demonstrates our observation about the advantage of download completion time of super-seeding.

Another interesting observation is that the saving of upload ratio is minimal in this scheme. It seems that the strength of saving uploading bytes of seeders is not as that obvious as the advantage of accelerating downloading when the upload rate seed=peers. We will discuss this in detail in Sect. 4.4.

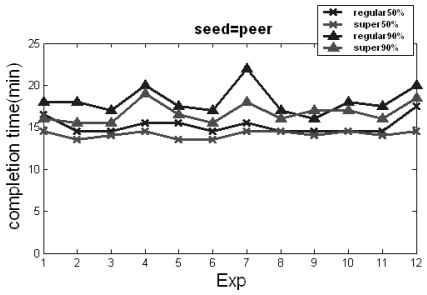


Fig. 7 Download completion time (seed=peer).

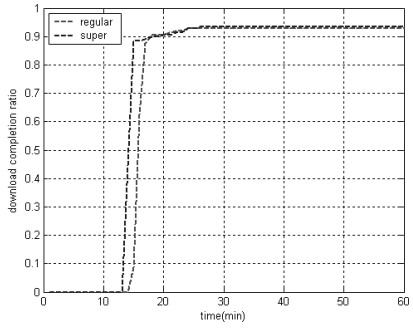


Fig. 8 Single download completion process (seed=peer), peer = seed = 75 kb/s, file: 55.5M, Seed upload ratio: regular: 124.2% super-seed: 124.1%.

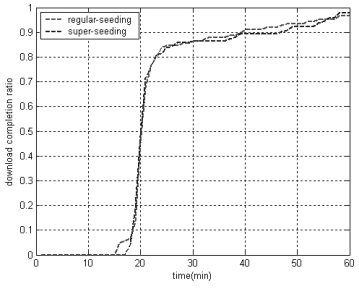


Fig. 9 Small file (S>P).

4.2 Upload Bandwidth Seed > Peer

When Seed Upload Bandwidth is above Peer Upload Bandwidth, experiments show, roughly, the super-seeding scheme would work better than the regular scheme. But the decease in downloading time is not that obvious as in the mode of upload rate seed=peer.

To get more details, we run experiments with both a file size of 55M and of 700M, expecting a longer downloading time would give some clue on what is going on in the whole process. The interesting finding is that the regular mode tends to produce first few seeds faster than super-seeding, but later on, with the more optimized distribution of seeds, the super-seeding scheme catches up and outweighs the regular mode. As is shown in Fig. 9 and Fig. 10, in the 55.5 file downloading, the late-started super seeding roughly catches up the regular seeding; with a longer downloading

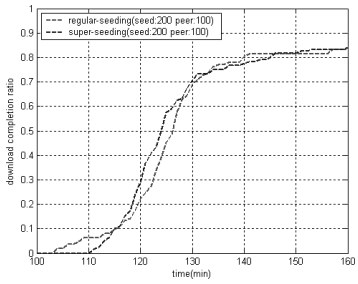


Fig. 10 Big file (S>P).

time, super-seeding in the 697.89 file downloading would quickly catches up the early-started regular-seeding, though the left 20% nodes seems to halt in process and unable to finish in our observing time, which is mainly due to slow nodes as we discussed in Sect. 3. (Note: the x-coordinate denotes the downloading completion time and the y-coordinate denotes the ratio of peers that finish their downloading).

There seems to be a tradeoff between the weakness of the super seeding in producing the first few completions of downloading (since super-seeding limits some piece download) and its advantage in the efficiency of producing multiple seeds. Under the mode of upload bandwidth seed>peer, those two influences tend to tussle with each other thus producing little speed increase.

Another observation is that for the regular-seeding, the seed upload ratio is 173.79/55.5 = 313.1%, but the super-seed upload ratio: 133.6/55.5 = 240.7%. This is rather obvious and encouraging progress in saving uploading cost. The advantage of the mode seed>peer lies most in this point. We will further discuss this in Sect. 4.4

4.3 Upload Bandwidth Seed < Peer

As Fig. 11 shows, over the 10 sets of experiments, the download completion time seems to be almost the same between super-seeding and regular-seeding. During different experiments, the performance of super-seeding varies greatly; sometimes downloads quite faster than regular-seeding but other times performs oppositely.

To be more specific, our single downloading process in Fig. 12 shows the download varies in the process and finally achieves minimal difference between super-seeding and regular seeding in mode seed<peer.

As a summary for three modes downloading time, super-seeding tends to perform better than regular-seeding when the upload bandwidth seed is equal to peer. The improvement is rather obvious and stable under this scheme. In scheme of upload bandwidth Seed (S)>Peer (P), super-seeding has some improvements but is rather limited due to its tussle between first few distribution and overall seeding efficiency. In scheme of seed<peer, the difference is minimal and unstable.

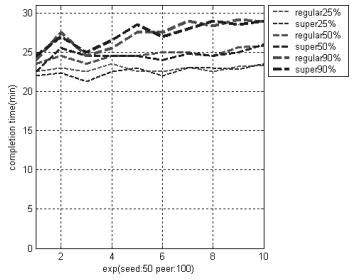


Fig. 11 Time (S<P).

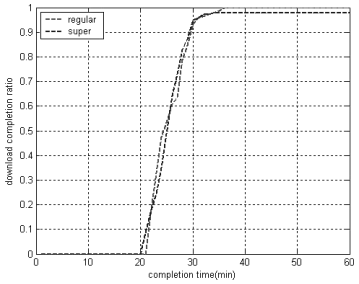


Fig. 12 Process (S<P).

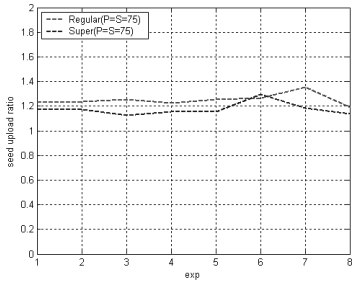


Fig. 13 Upload (S=P).

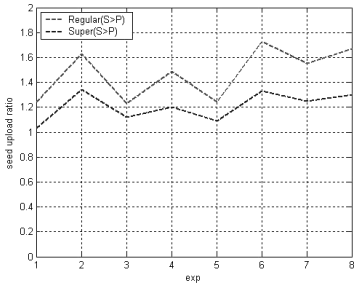


Fig. 14 Upload (S>P).

4.4 Seed Upload Cost under Three Modes

As Fig. 13 and Fig. 14 shows, there is an obvious saving of upload bytes in the mode of upload bandwidth seed>peer. Almost all experiments show that super-seeding scheme would produce over 20% saving of uploading ratio than regular-seeding. This verifies our analysis for seed uploading cost in Sect. 2.2. In mode of seed=peer, we can also observe the saving ratio but it is not as large as in mode seed>peer.

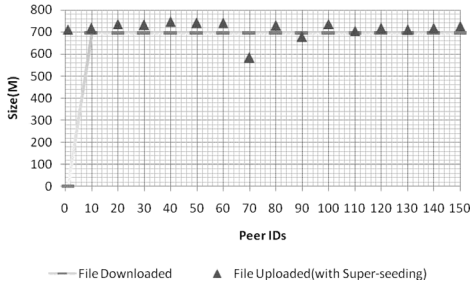


Fig. 15 File uploaded vs. file downloaded.

As a summary of several observed experiments average, we can find that the saving of upload ratio in the mode of Seed (S)>Peer (P) is most obvious (with an upload ratio of 229% vs. 180.9%), followed the mode of seed=peer (125.1% vs. 117.5%), and the influence over the mode of seed<peer is minimal (121.7% vs. 117.8%).

While the seed uploads less in super-seeding, peers tend to upload more to share with other peer, suggesting a good peer share ratio in super-seeding. As is shown in Fig. 15, those peers, uploaded almost as many as they downloaded, showing a good sharing ratio. The node with ID 1 is the seed, and the file size is 696.23M. For statistics convenience, nodes are randomly selected from those with early completions, thus tending to have similar uploading size when they finish downloading. Despite the scale appearance, they did have difference, i.e. node 130 is 710.57M and node 150 is 727.73M.

5. Experiments Varying Peer Behavior in Heterogeneous Network

In real P2P network, some peers would function in a selfish manner and leave the network immediately right after they finish the downloading. BitTorrent currently does not have any incentives to encourage users to stay in the system after they have downloaded the file, i.e., when they become seeds. Along with other free-rider behavior, the selfish behavior hinders much of the overall BitTorrent Performance [11]. In this section we will study the influence of selfish behavior of leaving seeds upon the performance of super-seeding.

We design our experiment by making peers self-kill themselves after they finish downloading, which can well simulate the overall selfish behavior. Considering the performance of super-seeding is rather stable and obvious under the mode of upload rate seed=peer, we choose it as the benchmark to compare the super-seeding performance with selfish behavior and without it.

Table 2 gives a summary about the download completion time and upload ratio in mode of upload rate seed=peer, with all peers acting in selfish manner. As we formerly discussed, in the mode of upload rate seed=peer, super seeding would obviously help downloading faster and is stable in saving upload ratio. But to our surprise, under the selfish

Table 2 Selfish behavior over super-seeding (seed=peer).

Time	R	S	R	S	R	S
25%	14.5	15.5	16	15.5	15.5	18
50%	15.5	17.5	19	18	27	>60
Ends	81%	87%	90%	50%	50%	35%
Upload	118%	120%	122%	120%	145%	133%

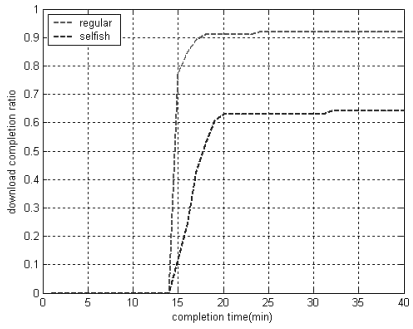


Fig. 16 Selfish manner vs. regular super-seeding.

manner, all above conclusion does not seem to be that obvious. Under selfish manner, super-seeding may even sometimes download slower than regular-seeding and do not save upload ratio. This shows that selfish behavior harms the advantage of super-seeding.

To be more specific, we draw out the whole process of two experiments in mode of super-seeding (upload bandwidth seed=peer), one in regular super-seeding and one in selfish manner. As Fig. 16 shows, in regular super-seeding (denoted as regular), after the first few downloading completion, new seeds would boom up and most peers would complete the downloading rather quickly. But in super-seeding with selfish manner (denoted as Selfish), the performance would be greatly hindered since the leaving peers after downloading would decrease the chance of other unfinished peers to complete. We can notice regular super-seeding takes around 1 minute from the first completion to 80% peer completion. But it takes more than 5 minutes for selfish manner to complete 60% after first downloading. After that, the whole network downloading tends to halt due to the negative influence of peers leaving and network dynamics.

Further, we experiment on the peer downloading time in heterogeneous network with varied peer behavior, i.e. selfish manner, and varied uploading bandwidth. The file size is 697.89M. Similar to our former analysis in two-class model, we categorize the peers according to their uploading bandwidth. The upload rate of the seed, with ID 1, is set to be 200 kb/s. IDs in domain (1,50) are slow public users, usually having an upload rate around 100 kb/s. IDs in domain (50,100) are normal users, usually having an upload rate around 200 kb/s. IDs in domain (50,150) are high-speed campus nodes, usually having an upload rate around 500 kb/s. As Fig. 17 shows, in domain (1, 50) super-seeding downloads slightly faster than regular-seeding; in domain (51,100), super-seeding steadily downloads faster

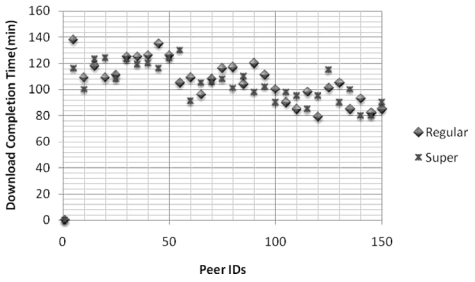


Fig. 17 Super-seeding performance in heterogeneous network.

than regular-seeding. We also note, with the negative influence of selfish behavior we analyzed above, some nodes with super-seeding, i.e. ID 20 and 55, download much slower than regular-seeding. In domain (101,150), the performance is unstable and super-seeding does not seem to have any advantage.

This results in heterogeneous network again verify our former conclusions. But we also note this one set of experiment cannot fully reflect the situation in heterogeneous network. As our future work, we have recently implemented a more large-scale experiment over real Internet and will report more on-field data in heterogeneous circumstances.

6. Conclusion

In this paper, we present an analytical and experimental study over the performance of super-seeding scheme in BitTorrent. With the data from our experiments over PlanetLab nodes in varied seed bandwidth and peer behavior, we can arrive at the following conclusion:

i) The super-seeding does help saving uploading cost of the seed. Though the needed file upload ratio by seed may not always be as low as 105% as BitTornado claims, in most cases, super-seeding can help saving an upload ratio of around 20%.

ii) The super-seeding can help deceasing the overall downloading time in certain scenarios. There seems to be a tussle of super seeder's weakness in producing first few completions and its efficiency in making multiple new seeds.

iii) The best circumstances to work: if the seed uploading cost is the prior consideration, super-seeding functions best when upload bandwidth seed>peer; if the overall downloading time is the priority, the super-seeding scheme should be adopted when upload bandwidth seed=peer. There is certain tradeoff between the peer download time and seed upload cost.

iv) The un-recommended circumstances: it is not recommended to adopt super-seeding when upload rate seed<peer. The advantage of super-seeding is not obvious and stable at this mode. Also super-seeding is not recommended for use of every peer, but must strictly limit to the initial content seeder.

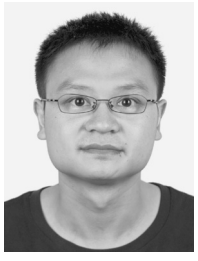
v) Factors that hinder the performance of super-seeding: selfish manner, concurrent seeds and network dynamics. To our best knowledge, this is the first comprehensive study, both analytical and experimental, of super-

seeding over BitTorrent performance.

Due to the dynamics of P2P network and the limitation of experiment runs, our conclusions may vary slightly in heterogeneous circumstances, yet our work still provides much academic and industrial insights.

References

[1] D. Qiu and R. Srikant, "Modeling and performance analysis of bittorrent-like peer-to-peer networks," Proc. ACM Sigcomm, Portland, OR, Aug. 2004.
[2] R. Bindal and P. Cao, "Can self-organizing P2P file distribution provide QoS guarantees?," Operating Systems Review, vol.40, no.3, pp.22–30, 2006.
[3] BitTorrent, www.bittorrent.com
[4] BitTornado, www.bittornado.com
[5] S. Banerjee, T.G. Griffin, and M. Pias, "The interdomain connectivity of PlanetLab nodes," Proc. 5th International Workshop on Passive and Active Network Measurement, 2004.
[6] uTorrent,www.utorrent.com
[7] M. Sirivianos, J. Han, R. Chen, and X. Yang, "Free-riding in BitTorrent networks with the large view exploit," 6th International Workshop on Peer-to-Peer Systems (IPTPS 2007), US, 2007.
[8] A.R. Bharambe, C. Herley, and V.N. Padmanabhan, "Analyzing and improving a bittorrent networks performance mechanism," IEEE Infocom, Barcelona, Spain, 2006.
[9] J. Mundinger, R.R. Weber, and G. Weiss, "Analysis of peer-to-peer file dissemination amongst users of different upload capacities," ACM SIGMETRICS Performance Evaluation Review, vol.34, no.2, pp.5–6, Sept. 2006.
[10] G. Neglia, G. Reina, H. Zhang, D. Towsley, A. Venkataramani, and J. Danaher, "Availability in BitTorrent systems," INFOCOM 2007, pp.2216–2224, May 2007.
[11] C. Girault and R. Valk, Petri Nets for Systems Engineering: A Guide to Modeling, Verification, and Applications, Springer-Verlag, 2002.
[12] R. Kumar and K.W. Ross, "Peer-assisted file distribution: The minimum distribution time," IEEE Workshop on Hot Topics in Web Systems and Technologies (HOTWEB'06), pp.1–11, Nov. 2006.
[13] H. Yin, C. Lin, Q. Zhang, Z.J. Chen, and D.P. Wu, "Truststream: A secure and scalable architecture for large-scale Internet media streaming," IEEE Trans. Circuits Syst. Video Technol., (to appear).
[14] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang, "Measurements, analysis, and modeling of BitTorrent-like systems," Proc. Internet Measurement Conference, 2005.
[15] R. Kumar, Y. Liu, and K.W. Ross, "Stochastic fluid theory for P2P streaming systems," Proc. IEEE INFOCOM, 2007.
[16] Planetlab, http://www.planet-lab.org/



Zhijia Chen is a Ph.D. student at Dept. of Computer Science and Technology, Tsinghua University, China. He was a visiting student at Department of Computer Science of Stanford University in Spring 2007 and a visiting scholar at CS Department of HKUST in 2008. He was the First Prize winner in American Mathematical Contest in Modeling 2004. His research interest is P2P streaming and BitTorrent performance evaluation. He has till now published over 10 papers in area of P2P networks and media streaming.

Chuang Lin is a chair professor and the Dean (former) at the Department of Computer Science and Technology, Tsinghua University, China. He received his Ph.D. degree in Computer Science from Tsinghua University in 1994, his M.S. from Graduate School of the Chinese Academy of Sciences, in 1981 and B.S. from in 1977. He served as the dean of computer science department at Tsinghua University during year 2004 and 2007. His research interests include computer networks, performance evaluation, logic reasoning, and Petri net theory and its applications. He is on editorial boards of Computer networks, IEEE Transactions on Vehicular Technology, etc and is the Duty Director for Internet Technical committee, Network and Data Communication Technical committee, and Petri Net Technical committee of China Computer Federation, and the Chinese Delegate in TC6 of IFIP.

Yang Chen is a Ph.D. Candidate at Department of Electronic Engineering, Tsinghua University. He has been a Visiting student in Wireless and Networking Group at Microsoft Research Asia (MSRA) since March 2006. His Current Research Interest include: Network Coordinate, P2P Overlay Network, BitTorrent, etc.

Vaibhav Nivargi is a graduate student at Computer Science department of Stanford University. Currently, MTS at Aster Data Systems (US). Interests include large scale Distributed Systems, Storage Networking, Systems Security and Machine Learning.

Pei Cao is Acting Assistant Professor of Computer Science at Stanford University (currently a consulting faculty). She received her Ph.D. from Princeton in 1996, M.S., from Princeton in 1992, and BS from Tsinghua University in 1990, all in computer science. In addition to substantial industrial engineering experience in the field of networking, Prof. Cao has published over twenty-five publications in web caching, distributed systems, and storage systems in top journals and conferences.