2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing

Improving BitTorrent Network's Performance via Deploying Helpers

Ke Xu[1], Yahui Yang[2], Tao Chen[1]
1 Department of Computer Science and Technology, Tsinghua University, Beijing, China
2 School of Software and Microelectronics, Peking University, Beijing, China
Email: xuke@mail.tsinghua.edu.cn, yhyang@ss.pku.edu.cn, chentao@csnet1.cs.tsinghua.edu.cn

Abstract

This paper presents a study on how to increase the BitTorrent Network's performance via deploying Helpers by modeling, simulating and analyzing. We first define some high-bandwidth, high-connection and controllable super nodes as Helpers, and then try to find the best way to deploy them. Unlike other researchers, we focus on combining the advantages of both Multi-Server Systems and BitTorrent Systems. Our main findings includes: (1) Deploying Helpers into BitTorrent Systems can distinctly enhance the system performance until the overall uploading bandwidth is no longer the constraint condition. (2) After deploying Helpers, the system can present resistance against selfish peers. (3) We can dynamically change the content of Helpers to serve the ever-changing hot torrents and make it only serve charged peers or local peers from economic angle.

1. Introduction

The peer-to-peer (p2p) network is used to share content files containing audio, video, and data in a scalable way. It no longer has the notion of "clients" or "servers", but only the notion of peers as both "clients" and "servers" to other nodes in the network. BitTorrent [2] is now becoming the most popular peer-to-peer communication protocol. The success of system has many reasons, such as: ease of use, cutting file into pieces, rarest first, tit-for-tat, and so on. Ease of use has greatly contributed to the fast adoption of BitTorrent, and may be the most important factor for the success of the system. Cutting a file into pieces of fixed size can make peer keep track of what it has and start uploading to other peers as soon as they have downloaded even the first piece. Rarest first ensures peers download the pieces whose neighbors have fewest and reduce the probability of certain peer having nothing of interest. And the tit-for-tat (TFT) policy makes peers prefer to upload to other peers which it can download fast.

Although the BitTorrent System is very successful and effective, and recently has been proved by many measurement and analytical research [4] [5] [6], it leaves us some unsolved problems. For Example:

(1)D.Qiu et al [6] have proved that BitTorrent scales very well, but it is not all worthy of trust. For example, if we try to download a file which is not very popular, it will cost us a lot of time to download or even cannot be finished. It is even worse in the case that the publishers want to publish certain files, for example, Microsoft wants to release a patch, but cannot assure that the process of publishing is fast and steady. So how can we solve this problem?

(2)Although the BitTorrent system performance is very well, the overall upload bandwidth is still the bottleneck of the system, and the system performance is limited. How can we further improve the system performance?

All answers to these questions lead to a simple solution: introducing some servers into the BitTorrent Systems.

The Multi-Servers system is very steady. If there are enough servers in the system, the system performance will be very good, Even if there are only a few servers, we can still ensure some downloader can get the file; at least the system will not die. The system can also ensure a certain downloading speed. But if the arrival rate of new requests is too high, the system performance will turn bad quickly.

The BitTorrent system scales very well, but if the seed leaving speed is very high or the uploading bandwidth is too low, the system will die quickly.

So to a certain extent, they are complementary to each other. We define some high-bandwidth, high-connection and controllable super nodes as Helpers,

978-0-7695-3492-3/08 $25.00 © 2008 IEEE
DOI 10.1109/EUC.2008.96

and use them to combine the advantages of both systems.

In this paper we use a simulator with a real BitTorrent trace to study the performance of the new mixed system. By using simulator, we can change some parameters or even add or remove some components of the system, which is very difficult to be implemented in the true experimental environment. Through real trace, we can improve the creditability of our experiments. We also spread the fluid model by D.Qiu et al [6], and use simulator results to validate our model. We believe that our research is complementary to previous BitTorrent researches and establishes a new perspective.

The rest of paper is organized as follows: Section 2 lists some related works. Section 3 describes our fluid model with Helpers. Section 4 gives details of our simulator. Section 5 lists a series of simulation results and validates the model. At last, section 6 concludes.

2. Related work

There have already been many researches on BitTorrent, including analytical studies, measurement studies, and simulator studies.

At the analytical side, D.Qiu and R.Srikant [6] built a simple fluid model and got the relationship between average download time and node joining, as well as leaving rate and node bandwidth. They also studied the scalability, performance and efficiency of such a file-sharing mechanism, then considered the built-in incentive mechanism of BitTorrent and studied its effect on network performance. Their main finding is that the BitTorrent scales very well and the file sharing are very effective. L.Guo et al[9] found that due to the exponentially decreasing peer-arrival-rate in reality, service availability in such systems becomes poor quickly, after which it is difficult for the file to be located and downloaded. Shirshanka Das et al [7] tried to bring together traditional client/distributed-server paradigms and pure BitTorrent based peer-to-peer delivery models. The content-delivery-provider benefits as it can provide guarantees with expected download times while deploying a much lower number of servers, and the end user benefits by getting a much shorter download time compared to a pure BitTorrent download. We expand this idea and do a series of more in-depth research and experiments. Yao Yue et al [10] also expanded the simple fluid model with multi-group.

At the measurement side, M.Izal et al [4] studied BitTorrent based on the "tracker log" of a pop file (Red hat 9 Distributions). The main findings are: 1）the leechers will stay for another 6.5 hours after they have finished downloading. The reasons include: the downloading process often finishes at night; the downloader is lazy to stop; and the downloading process is legal. 2) The average download speed is more than 500kb. 3) TFT mechanism works well 4) LRF mechanism works well 5) the seeds upload is 2 times of the leechers upload. Another study of long time (8 months) and more files (more than 60000files) by J.A.pouwelse et al [5] got some different results. The main findings include: 1) the average download bandwidth is only 240kb 2) only 17% of the leechers stay more than 1 hour after they have finished downloading. And the most important finding is: *a few highly reliable seeds are more important than much larger number of short-lived seeds*, which is the main motivation of our research.

At the simulator side, for us, the most related work is done by Ashwin R.Bharambe et al [1]. They presented a simulation-based study of BitTorrent to deconstruct the system and evaluate the impact of its core mechanisms. They focused on peer link utilization, file download time and fairness amongst peers in terms of volume of content served. And their simulation methods greatly help us in designing experiments. Weishuai Yang et al [8] implemented a BitTorrent simulator called General Peer-to-Peer Simulator, which is the first BT simulator that models the full behavior of the protocol. GPS[8] is a message level discrete event simulator with a built-in protocol implementation of BitTorrent. It allows simulation of both structured and unstructured overlays. While it is a message level simulator, it also partially models the underlying network topology using the GT-ITM model with Transit-Stub topology . It does not model each packet, but provides a number of different flow level models. And both of their simulator-design methods are very helpful.

Our research is different from previous research, because we always focus on the impact of deploying Helpers in the following aspects: How do they change the BitTorrent fluid model? How do they improve the system performance? And how should we deploy them to get the best performance?

3. Analysis

3.1. Definition of helper

We define Helper like this:

1) Essentially, Helper is a seed in BitTorrent, so it can be involved into the system.

2) Its bandwidth and number of connections is much higher than the normal seed.

3) It will never leave the system, while normal seed leave the system.

4) We can manually change the number and the content of the Helpers.

3.2. Deploying Helpers into BitTorrent System

Then we build a model based on D.Qiu et al [6]'s simple fluid model, introduce the concept of Helpers, and use the results of previous researchers to further simplify the model.

D.Qiu et al [6] assumed the arrival rate of new requests according to a Poisson process. L.Guo et al. [9] found that the peer arrival rate at a torrent is actually given by an exponential decreasing function of the time.

D.Qiu et al [6] also discovered that the effectiveness of file sharing $\eta = 1-(\log(p)/p)^k$, while p is the number of pieces, and k is the maximal connections. So η is very close to 1.

The parameter θ means that the rate leechers abort the download. It is probably because the downloading speed is too slow or the leechers are no longer interested in the torrent. L.Guo et al. [9] found that if the system has some altruistic seeds, which always stay until the last downloader finishes, the failure ratio of the torrents θ is probably equal to 0. In our systems, we have these altruistic seeds (at least one server), so we assume θ is 0.

Then we find that the upload bandwidth of Helpers M is actually defined by m (the maximal number of peers it can upload to) and u (the uploading bandwidth for each peer), and n (the number of Helpers). We assume:

$$M = n*m*u \quad (1)$$

Symbol	Meanings
$x(t)$	number of leechers in the system at time t.
$y(t)$	number of seeds in the system at time t.
λ	the arrival rate of new requests
μ	the uploading bandwidth of a given peer
M	the uploading bandwidth of the Helpers, we assume $M = n*m*u$
n	the number of Helpers
m	the maximal number of peers it can upload to
u	the Helper's uploading bandwidth for each peer
c	the downloading bandwidth of a given peer
γ	the rate at which seeds leave the system
θ	the rate leechers abort download, we assume it is 0
η	the effectiveness of file sharing, we assume it is 1
T	Average downloading time
F	File size, we assume it is 1

Table 1: the parameter of fluid model

We define parameters in Table 1, and try to describe the evolution of x and y based on the above model.

$$\begin{aligned}\frac{dx}{dt} &= \lambda - \min\{cx(t), \mu(x(t)+y(t))+M\} \\ \frac{dy}{dt} &= \min\{cx(t), \mu(x(t)+y(t))+M\} - \gamma y(t)\end{aligned} \quad (2)$$

To study the steady state, we let

$$\frac{dx(t)}{dt} = \frac{dy(t)}{dt} = 0$$

And we can obtain:

$$\begin{aligned}0 &= \lambda - \min\{cx(t), \mu(x(t)+y(t))+M\} \\ 0 &= \min\{cx(t), \mu(x(t)+y(t))+M\} - \gamma y(t)\end{aligned} \quad (3)$$

After solving (3), we can get

$\bar{x} = \lambda / c$ (Downloading bandwidth constraint)

$\bar{x} = \frac{\lambda - M}{\mu} - \frac{\lambda}{\gamma}$ (Uploading bandwidth constraint)

To calculate the average downloading time for a peer in steady state, we use Little's law as follows:

$\bar{x} = \lambda * T$

And we can obtain:

$T = \frac{1}{c}.$ (Downloading bandwidth constraint)

$T = \frac{1}{\mu} - \frac{1}{\gamma} - \frac{M}{\mu\lambda}$ (Uploading bandwidth constraint)

If let M=0, we can get the result of pure BitTorrent case.

$T = \frac{1}{c}.$ (Downloading bandwidth constraint)

$T = \frac{1}{\mu} - \frac{1}{\gamma}$ (Uploading bandwidth constraint)

Then we summarize the 2 cases in Table 2.

	BitTorrent	Mixed
T(download bandwidth constraint)	1/c	1/c
T(upload bandwidth constraint)	$\frac{1}{\mu} - \frac{1}{\gamma}$	$\frac{1}{\mu} - \frac{1}{\gamma} - \frac{nmu}{\mu\lambda}$

Table 2: The average download time of the 3 cases

Then we can conclude: *The BitTorrent System with Helpers will get better performance than pure BitTorrent System.*

We will further validate the fluid model through

the simulation result in next Section.

4. Methodology

4.1. Simulator detail

If we want to study the performance of new mixed system, we must change the system parameters and add some components into the system. It is very difficult to be implemented in the true experimental environment. So we use a simulation-based approach to understand and deconstruct the new system.

Our simulator is basically an expansion of Ashwin R.Bharambe et al [1]'s, and we also refer the WeiShuaiYang et al [8]'s implementation.

Our discrete-event simulator models peer and server's activity (joining, leaving, and block exchanging) as well as many of the associated BitTorrent mechanisms (local rarest first, tit-for-tat, etc.) in detail. Through comparing the two existing simulators, we find some most important design principles.

1) We don't model the detail of the protocol, just like sub-piece; due to our simulator is not packet level, the sub-piece is totally useless.

2) We don't model the bottleneck in the middle of network, because we have controlled the bandwidth in the edge of the network. There's no need to do this work twice.

The maximum scale of our experiments is up to 2000 nodes.

4.2. Simulator Parameters

We set the common simulator parameter in Table 3.

Parameter	Meanings
Trace	Redhat 9
File size	100MB
Leechers download/uplink bandwidth	1536Kb/512Kb
Neighbors	40
Block Size	256KB
Max Uploads	5
Simulation Time	5Hour
Nodes Limit	2000
Helper Bandwidth	50Mb

Table3: the parameters of simulator

To derive realistic arrival patterns, we use the tracker log for the Redhat 9 distribution torrent [4].

We choose the File Size 100MB because it is just the fit size for our simulation. If it is too small, we can't get the real situation. But if it is too large, we may not have enough simulation time to wait for the system entering the steady state.

Then we choose the downloader's bandwidth 1536Kb/512Kb in all of our experiment except the Hetero-peers experiment.

Fig.1 shows that the average downloading time of system with simulation time. It shows that the system gradually turns into steady state after 5000 second.

We will do a series of experiments to validate the fluid model and observe the different aspects of system performance increase via deploying Helpers in different ways.

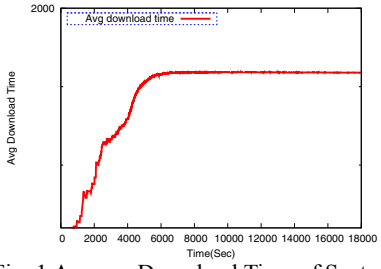


Fig. 1 Average Download Time of System

5. Simulation result

5.1. The mixed system is better

In BitTorrent system, system performance is usually constraint by the overall uploading bandwidth. So through the fluid model, we conclude that the average downloading time of mixed system $\frac{1}{\mu}-\frac{1}{\gamma}-\frac{M}{\mu\lambda}$ is less than the pure BitTorrent System $\frac{1}{\mu}-\frac{1}{\gamma}$.

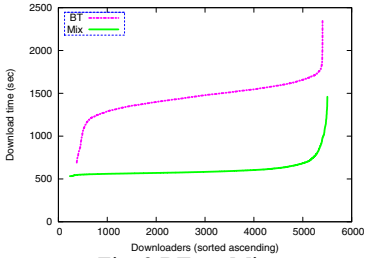


Fig. 2 BT vs. Mix

Fig.2 shows that the BitTorrent System with Helpers has the better performance.

5.2. Touch the optimal performance

In BitTorrent System, the overall downloading bandwidth is usually much higher than the overall uploading bandwidth. So as the model shows that the average downloading time is usually $\frac{1}{\mu}-\frac{1}{\gamma}$, but not

1/c.

Fig.3 shows that the uploading utilization is nearly 100%, but the downloading utilization is only less than 40%. So the overall uploading bandwidth is the constraint conditions, and the system performance is restricted.

Fig.4 shows that after adding 5 Helpers, the downloading utilization has a significant improvement. If we continually add servers, we can finally balance the difference between the overall uploading and downloading bandwidth.

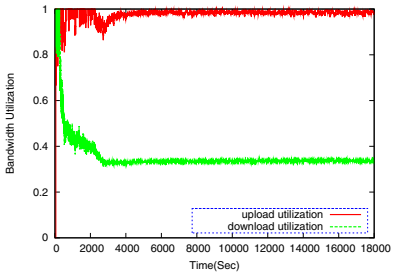


Fig. 3 Bandwidth Utilization in pure BitTorrent System

From the fluid model we can calculate the theoretical minimum of average download time.

$$T_{\min} = F * 1/c$$

In our simulation the file size F is 100MB, and the downloading bandwidth is 1536kb. We can get:

$$T_{\min} = F * 1/c = 100MB/1535kb = 533\sec$$

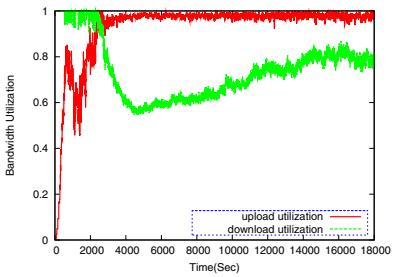


Fig. 4 Bandwidth Utilization in pure BitTorrent System

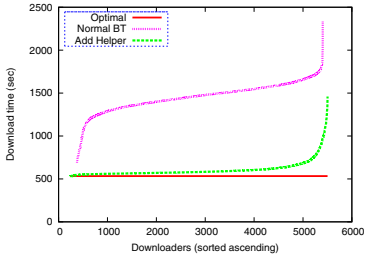
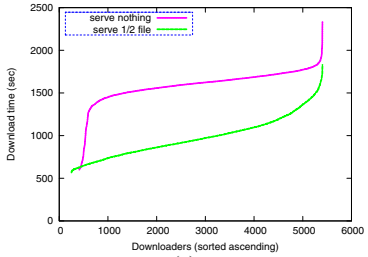


Fig. 5: Adding enough Helpers to get the best performance

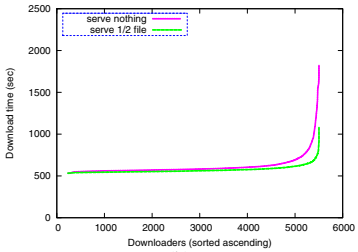
Fig.5 shows after adding enough Helpers, the system performance will be very close to the theoretically optimal state.

5.3. Presents resistance against selfish peers

In our fluid model γ is the rate at which seeds leave the system. In pure BitTorrent system, the average downloading time is $\frac{1}{\mu}-\frac{1}{\gamma}$, if $\gamma=\infty$ (also means the leechers leave as soon as they finish downloading, serving nothing), the average downloading time turns to $\frac{1}{\mu}$ .The system performance decreases quickly.



(a)



(b)

Fig.6 (a) Downloading time affected by γ in pure BitTorrent
(b) Downloading time affected by γ in BitTorrent with Helpers.

Fig.6 (a) shows that in pure BitTorrent network, this behavior will do great damage to the system performance.

But in the mixed system, the impact of $\gamma=\infty$ is much less than before. We can conclude from the fluid model even if $\gamma=\infty$, the average downloading time only turns to $\frac{1}{\mu}-\frac{nmu}{\mu\lambda}$, the parameter γ is no longer the decisive factors.

So the system presents the resistance against selfish peers.

5.4. Dynamically changing contents of Helpers

The real BitTorrent System has more than one torrent. For simplicity's sake, of all of our above experiment, we consider only one torrent.

We consider the case when the system has multi-torrents in this experiment, assuming one torrent is much popular than the other. Then we dynamically change the composition of Helpers to get the best performance.

Fig.7 shows that the whole system's performance is generally decided by the popular torrent. So we must deploy more Helpers to assist the popular torrent, and we also need to leave a small quantity of

Helpers to other torrent.

For example, if we have 5 Helpers totally, one of the optimal cases is that we put 4 Helpers to assist the popular torrent, and 1 Helper to help other torrents.

While the popular torrent is changed, we can also change the files in our Helpers to get the whole system's best performance.

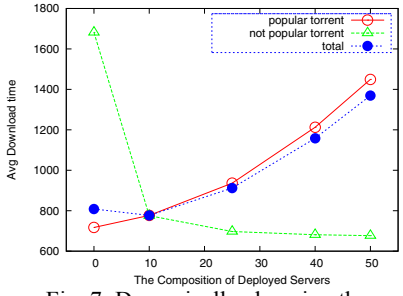


Fig. 7: Dynamically changing the composition of deployed Helpers

5.5. Serving only chosen Peers

In the real system, deploying Helpers costs a lot. So the incentive of deploying Helpers is very important.

For example, we can make Helper only serve charged peers or local peers, and the chosen peers can have a very great performance improvement.

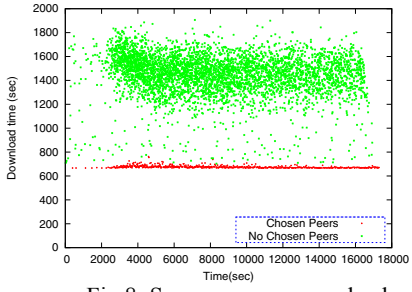


Fig.8: Servers can serve only chosen IP

Fig.8 shows that the chosen peers have a very distinct performance increase.

For example, we can deploy many Helpers in the system, and a Helper serves only one local area. This policy is very useful when a university want to provide some convenience for its students.

6. Conclusion

In this paper, we have described a fluid model and a series of experiments, aiming at understanding the performance increase after deploying Helpers. And we also try to find the best way to deploy them.

Our main findings are summarized as follows: (1) Deploying Helpers into BitTorrent Systems can distinctly enhance the system performance until the overall uploading bandwidth is no longer the constraint condition. (2) After deploying Helpers, the system can present resistance against selfish peers. (3) We can dynamically change the content of Helpers to serve the ever-changing hot torrents and make it only serve charged peers or local peers from economic angle.

Our simulation result shows that the fluid model matches the BitTorrent System with Helpers very well. And the Helpers are very helpful on the system performance.

As for future works, we intend to consider the topology of network in our simulator. We would like also to deploy some servers in real BitTorrent Network to do some real experiments.

7. Acknowledgement

Supported by the National High-Tech Re-search & Development Program of China (863 Program) (Grant Nos. 2008AA01A326)

8. References

[1] Ashwin R.Bharambe, Cormac Herley, Venkata N.Padmanabhan Analyzing and Improving BitTorrent Performance. INFOCOM 2006.

[2] BitTorrent. http://bittorrent.com

[3] B. Cohen. Incentives build robustness in BitTorrent. In Proc. Of Workshop on Economics of Peer-to-Peer Systems, May 2003.

[4] M. Izal, G. Urvoy-Keller, E. Biersack, P. Felber, A. A. Hamra, and L. Garc'es-Erice. Dissecting BitTorrent: Five months in a torrent's lifetime. In Proc. of Passive & Active Measurement Workshop, Apr. 2004.

[5] J. Pouwelse, P. Garbacki, D. Epema, and H. Sips. The BitTorrent P2P file-sharing system: Measurements and analysis. In Proc. of International Workshop on Peer-to-Peer Systems, Feb. 2005.

[6] D. Qiu and R. Srikant,. Modeling and performance analysis of BitTorrent-like peer-to-peer networks. In Proc. of ACM SIGCOMM, Aug. 2004.

[7] Shirshanka Das, Saurabh Tewari, Leonard Kleinrock The Case for Servers in a Peer-to-Peer World. IEEE ICC 2006.

[8] Weishuai Yang and Nael Abu-Ghazaleh. GPS: A General Peer-to-Peer Simulator and its Use for Modeling BitTorrent, in Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2005.

[9] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang. Measurements, analysis, and modeling of BitTorrent-like systems. In ACM SIG-COMM/USENIX IMC 2005, pages 19–21, Oct. 2005.

[10] Yao Yue , Chuang Lin, Zhangxi Tan Analyzing the Performance and Fairness of BitTorrent-like Networks Using a General Fluid Model, In IEEE GLOBECOM 2006.