Neighbour selection strategies for P2P systems using tit-for-tat exchange algorithm

L.G. Alex Sung
University of Waterloo
lgasung@uwaterloo.ca

Herman H.Y. Li
University of Waterloo
hyh2li@uwaterloo.ca

ABSTRACT
BitTorrent (BT) has recently received tremendous attention due to its superior efficiency in download large files. Each client must decide on a set of peers to download from. Thanks to the Tit-for-Tat (TFT) exchange strategy, incentive for utilizing peers' full uploading capacity is enhanced and free riding is discouraged. As the assignment of neighbours is totally random in the original BT implementation, capacity-limited peers may be assigned to high-capacity peers. Under the TFT exchange policy, the capacity-limited peer may not be able to download efficiently. Moreover, in the overlay network, neighbours may be physically far away from each other, resulting in a waste of network resources and routing time. For Peer-to-Peer(P2P) systems using TFT strategy to exchange data pieces, we propose two organized neighbor-selection strategies based on link capacity and node locality to increase the content distribution efficiency. Our work, based on experiments done on the PlanetLab nodes, shows that these schemes do improve overall system throughput by reducing the average download time for around 18%.

Categories and Subject Descriptors
C.2.4 [**Computer-communication Networks**]: Distributed Systems—*Distributed applications, Distributed databases*; D.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms
Experimentation, Measurement, Performance

Keywords
Peer-to-peer, P2P, Neighbour Selection

1. INTRODUCTION
Since the launch of Napster in 1999, more people are equipped with cheaper broadband Internet services, higher bandwidth networks and larger storage capacities. Peer-to-Peer (P2P) systems have changed their landscapes from sharing small files such as mp3 files for early systems to sharing larger files such as application programs, movies and even DVD images. Although P2P systems today have better networking capabilities, the availability of data is largely determined by the users' incentive to share their files and their uploading bandwidth. BitTorrent (BT) is the first popular P2P system tailored for sharing large files. It is famous for its downloading efficiency over other P2P systems. Thanks to the Tit-for-Tat (TFT) exchange strategy, incentive for utilizing peers' full uploading capacity is enhanced and free riding is discouraged. The optimal strategy for users is maximizing their uploading speed [6].

BT is a hybrid P2P system with central tracker servers. Each shared file is announced to at least one tracker server. When users want to download a file, they need to get the tracker file (with .torrent file extension) which contains the metadata of the file they want to download. The metadata includes the fileID and the IP address of the tracker servers. The tracker file may be downloaded from some web servers, web forum or newsgroups. The BT client program installed on the user's machine then contacts the tracker server and requests for a set of random peers. The steps to join the BT network is illustrated in Figure 1. Files are divided into pieces of 256kB in size. Studies such as [9] show that dividing large files into small chunks for download improves efficiency. The client program establishes connections with the set of peers and find out what pieces reside in each of the peers. Then, it simultaneously downloads and uploads different pieces from and to different peers. If the number of peers can be connected is less than 20, it contacts the tracker server for another set of peers. There are two types of peers: i) Seeds, which hold a full copy of the file and simply upload; and ii) Leechers, which have incomplete files and are both downloading and uploading.

If the assignment of neighbours is totally random, capacity-limited peers may be assigned to high-capacity peers. Under the TFT exchange policy, the capacity-limited peers may not be able to download efficiently because they are unable to upload efficiently in return. Moreover, in the overlay network, neighbours may be physically far away from each other, resulting in a waste of network resources and routing time. In this paper, we have studied the effect of assigning neighbours in more organized fashions and show that our proposed strategies reduces the average download time for all peers.

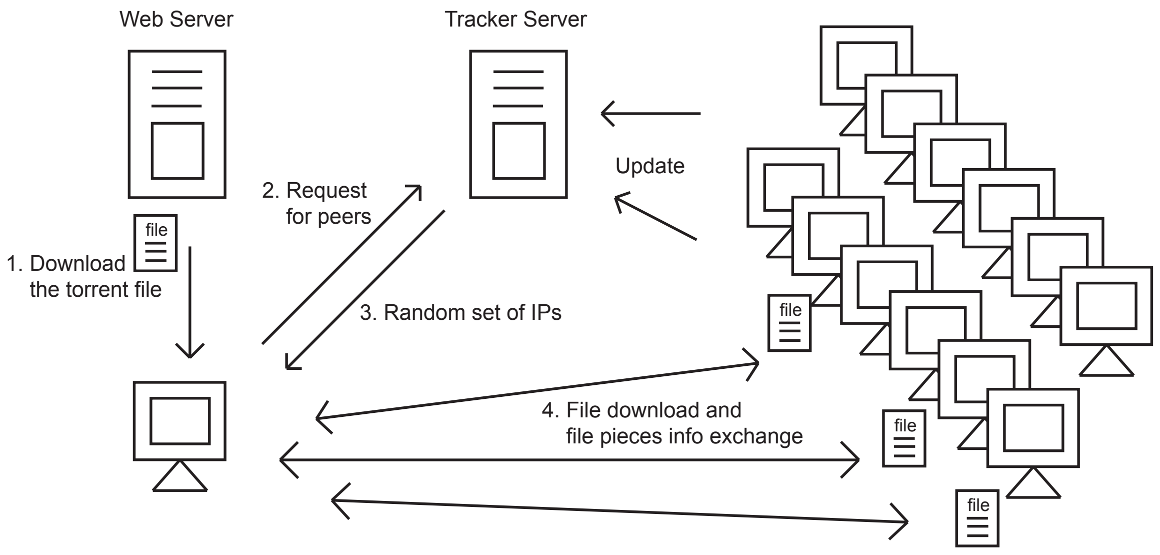

Figure 1: Steps to join the BT network

1.1 Contribution

For Peer-to-Peer systems using TFT strategy to exchange data pieces, we propose two organized neighbor-selection strategies to increase the content distribution efficiency. In particular, we have chosen the widely known BT protocol to carry out our tests. We believe that the TFT strategy will exist in future P2P systems aiming at sharing large files. Our contribution includes:

- Our results show that selecting neighbour peers based on peer capacity and locality does decrease the average download time for all peers and hence, increasing the overall system throughput.
- Results are collected from real implementation of the strategies deployed on hundreds of nodes around the world.
- Our proposed strategies are applicable to systems other than BT and not restricted to the hybrid architecture. No matter whether a central server exists or not, peers still need to obtain a peer set in order to exchange data pieces.
- Client program and the BT protocol remain unchanged. Strategies can be easily deployed to existing BT trackers.

1.2 Organization

Section 2 introduces BitTorrent which our implementation is based on. Section 3 and 4 introduces our proposed strategies. Section 5 explains our experiments and discusses about the results. Section 6 talks about the related work. Section 7 concludes and discusses potential future work.

2. BACKGROUND

The original implementation of BT distributes the peer set from the tracker in a random fashion in order to avoid the power-law distribution [4]. We are exploring the effectiveness of two relatively more organized neighbor-selection strategies. Effectiveness for this kind of data replication or content distribution can be treated as the total system throughput. Higher effectiveness means shorter average download time for all peers. While avoiding the power-law distribution, we distribute peers according to different peer groups classified by their network capacity and locality. For network capacity, peers are classified into different groups according to their upload or download ability. For locality, peers are classified according to their physical location determined by their IPs. Randomness is preserved by including some peers of other capacity or locality groups in the peer set. Peers from each capacity or locality group are chosen randomly.

The sustainability of the BT system depends on the file pieces availability. As a data replication system, if one or more pieces are missing from the whole system, the system fails as no peer can get a full copy until at least one peer with those pieces joins the system again. Preserving certain degrees of randomness not only avoids the power-law distribution for pieces, but also ensures that pieces can pass onto peers who join the system later. Therefore, it improves the system sustainability by promoting higher pieces availability. Strict group matching may result in some pieces only appearing in one classification group but not in others. Together with the local rarest first strategy to exchange pieces, once a rare piece reaches a classification group (from a peer of another classification group), it can be spread efficiently and sustain the system.

3. GROUP BY CAPACITY

Matching peers according to capacity similarities improves efficiency due to the rate-based TFT exchange strategy. By default, a peer selects four peers, which offer the best downloading rate, to upload to. Moreover, the peer randomly selects some peers to "optimistic unchoke" (upload) in order to test whether they can offer a better downloading rate.

Table 1: Peer grouping according to sorted network capacity

Group	Sorted capacity
Unknown	Unknown
High	0% to 33.33%
Medium	33.33% to 66.66%
Low	66.66% to 100%

When low ability peers are connected to high ability peers, it is very likely that they can only get pieces when they are being optimistic unchoked. Furthermore, they will get choked again very quickly as they cannot offer a good exchanging rate. On the other hand, if they are grouped among peers with a similar ability, they are able to exchange with each other continuously; in contrast to just wait for being optimistic unchoked.

In our implementation, peers are divided into 4 groups by sorting their downloading or uploading capacity as shown in Table 1. In the BT protocol, peers must report the number of bytes downloaded and uploaded to the tracker in less than 30 minutes interval. Once the peer has become a seed, it will request for peers regularly every 5 minutes and report their statistics. The capacity is calculated from the difference in the reported number of bytes at different time. New joining peers are classified into the unknown capacity group. Finding the optimal number of groups to divide the heterogeneous peers would conjecturably be a very difficult task and it will remain as a future work. After capacity information of the peer is available, the 50 peers are selected by having certain fraction of peers belonging to the target matching group, with the remaining peers from the other groups. The fraction of same-group peers is a variable.

One may think of modifying the BT protocol to let clients report their own capacity. Let alone the significant amount of peers in other P2P systems misreporting their own bandwidth [7] due to the lack of incentive or networking knowledge, even truly reported bandwidths may not be accurate enough for the neighbour matching. Real world users may download data in parallel or start several BT downloading sessions at the same time. In this case, the network capacity is not solely available for a single BT session. Our solution of calculating the bandwidth based on the reported number of bytes downloaded or uploaded takes care of the current capacity available. Moreover, peers have incentive to honestly report their statistics as it enables them to download faster.

Matching peers with the same upload capacity or same download capacity: In these strategies, peers with similar download / upload capacity are grouped respectively. The target matching group will be peers with the same download / upload capacity as the neighbour-requesting peer. For example, if the fraction of same-group peers required is 0.75 and the peer belongs to the Medium group, 37 out of the 50 peers are selected randomly from the Medium group. The remaining 13 peers are selected randomly from other groups. If there are less than 37 peers in the Medium group peers, the remaining peers will be substituted by peers from the other groups. If the total number of peers is less than 50, all peers are returned.

Matching peers by upload capacity to download capacity: In this scheme, peers in the High upload capacity group are matched with peers in the High download capacity group; Medium upload capacity peer to Medium download capacity; and Low upload capacity peer to Low download capacity. Peers in a certain upload capacity group may not belong to the same download capacity group. For example, peers belonging to the Low upload capacity group may belong to the High download capacity group.

4. GROUP BY LOCALITY

Matching peers by locality let them benefit from the low network latency and get a higher speed in their P2P connections. The topology of the overlay network can better match the underlying network. In the case that the uploading capacity was not previously fully utilized, this scheme can maximize the uploading speed by exchanging with peers that are physically closer.

In this paper, we assume there is an accurate way accessible by the tracker server to derive the latitude and longitude values from the IP addresses of peers. In our experiments, the IP addresses of each Planet-lab node are pre-submitted to NetGeo [1], an online IP to physical location database. NetGeo determines locality by the hostname in the WHOIS record. Although the accuracy is limited, it is enough for us to partition the peers. The limitations of using the WHOIS database to determine locality is discussed in Section 5.4. For PlanetLab nodes, their real physical address are known. The locality generated are checked against the real location manually. Obvious errors, such as all China nodes are mapped to Australia, are fixed. Since the errors are consistent and fine-grained locations are unnecessary, close peers can still be grouped. For example, all European and Middle East nodes are consistently reported by NetGeo as located Netherlands where ripe.net acts as the Regional Internet Registry in the region.

When a peer requests the tracker for a peer set of 50 peers, we first obtain a sorted list of peers according to the difference in physical distance to the requested peer. The distance is calculated from their coordinates. The top 33.33% of the peers are considered physically close and belong to the same group as the requesting peer. Again, the problem of how to best determine the percentage of close peers remains open. The 50 peers are selected by having certain fraction of peers belonging to the same group, and remaining peers from the last 66.66% peers in the sorted list. For example, if the fraction of same-group peers required is 0.75, 37 out of the 50 peers are selected among the top 33.33% peers in the sorted list. If there are more than 37 same-group peers, 37 will be selected randomly from close peers. The remaining 13 will be selected randomly from the last 66.66% of peers. If the total number of peers is less than 50, all peers are returned. If the number of same-group peers is not enough, the remaining peers will be substituted by peers from the other group.

5. EXPERIMENTS

The experiments were conducted on the inter-continental hosts provided by PlanetLab. The PlanetLab environment

Table 2: Locations of PlanetLab nodes used

Location	Percentage
US	52.91%
Asia	18.46%
Europe	17.50%

is designed as "an open platform for developing, deploying, and accessing planetary-scale services." [2] It gives access to nodes that are geographically distributed and provide a way to deploy BT clients over a vast number of nodes. An additional advantage is that nodes in the PlanetLab experience the same network latency, various delays and bandwidth constraints as nodes used by real users. The distribution of localities of the PlanetLab nodes used is shown in Table 2. This distribution comparable to the BT peers downloading the Redhat 9 distribution [5]. [1]

5.1 Environment

As of this writing (April 2005), the PlanetLab system consists of 564 nodes over 266 geographic locations. In our experiments, we found that approximately 250 nodes were accessible at a given point in time. Due to firewall policies, about 210 nodes were actually able to carry BT traffic. A 255.44MB file was used for the file transfer. The size of the file was chosen as such because the average download speed of the PlanetLab nodes were about 5 Mbps. Theoretically this file could be downloaded in 6.81 minutes with a 5 Mbps link and 17.0 minutes with a slower 2 Mbps link. Therefore, we could conduct one test in 30 minutes. We selected one node (planetlab1.cs.ubc.ca) to be the tracker and the initial seed. The rest of the nodes acted as leechers and downloaded the file from the initial seed and from one another. One leecher was started each second. It took approximately 3.5 minutes for all leechers to start.

5.2 Design

The official BitTorrent 3.4.2 client and tracker was used in our experiments. The tracker was modified to use one of the two proposed peer selection schemes. System throughput in terms of average download time was collected while varying the degree of randomness in the peer selection scheme. Four variations of trackers paramenters were used for each scheme. These were the original tracker, and different fraction of same-group peers at 0.5, 0.75 and 1.0 as explained in Section 3 and 4. One side of the spectrum was random selection, which was what the original BT tracker did; the other side was totally structured selection. We wanted to study the tradeoff between sustainability and system throughput. The BT clients did not need modifications at all. In other words, our optimization scheme was transparent to all BT users.

5.3 Results and Discussion

Figures 2, 3, 4 and 5 show the percentage of completed peers over time elapsed in the proposed grouping schemes since the first peer-join message is received by the tracker. In our tests, when peers have finished downloading, they will remain as seeds in the system and continue to serve their upload bandwidth to other peers. Seeds periodically report their status to the tracker every five minutes, and the tracker returns 50 neighbours to them. Even if the leechers do not request for more peers, their number of neighbours can still increase when the seeds actively connect them.

[1]Most Australia nodes (17.9% of all nodes in [5]) actually originates from Asia due to the error mapping of NetGeo.

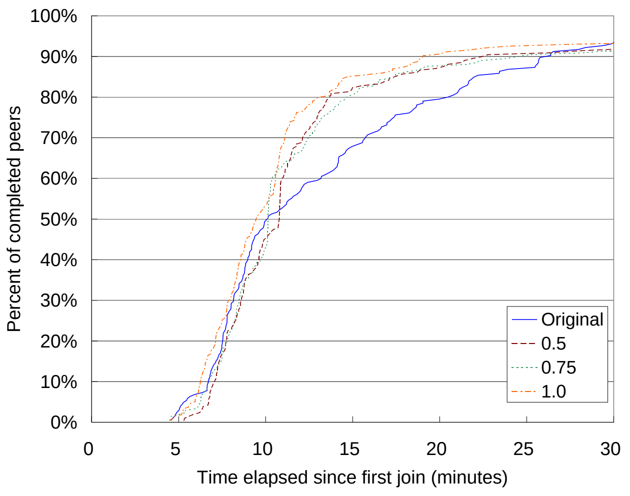

Figure 2: Peer selection by matching download capacity - Percent of completed peers over time elapsed

5.3.1 *Peer Selection by Capacity*

At the beginning, the percentage of completed peers are roughly the same for all schemes when comparing to the original tracker. From the tracker log, the first capacity of peers is usually calculated when the peer have completed the download. In other words, peer capacity is known only when they have become a seed and update their status to the tracker.

When a peer contacts the tracker, 50 neighbours are given to it every time. Therefore, the new seeds are also given 50 matched peers. From Figures 2, 3 and 4, we can see that the later 50% of completed peers gain from having seeds of matched capacity connecting to them and hence, decreasing the download time. In our experiments, most of the gain in system throughput are resulted from better matching seeds with leechers.

In the real situation, users tend to disconnect after they become seeds, when all the pieces are downloaded. If the high capacity peers complete and disconnect, most peers in the system are the low capacity peers. Some data pieces which are rare in low capacity groups may disappear and never return, due to the leaving of high capacity seeds. In the graphs, we have shown that the performance of the proposed schemes at different randomness levels are similar. We suggest using 0.5 or 0.75 as fraction of same-group peers to enhance system sustainability.

5.3.2 *Peer Selection by Locality*

In contrast to matching by capacity, the locality of peers are known when they contact the tracker at the first time.

Table 3: Peer selection by matching download capacity - Average download time for various peer selection schemes

Selection scheme	Average download time (min)
Original	10.22
0.5	8.92
0.75	8.72
1.0	8.12

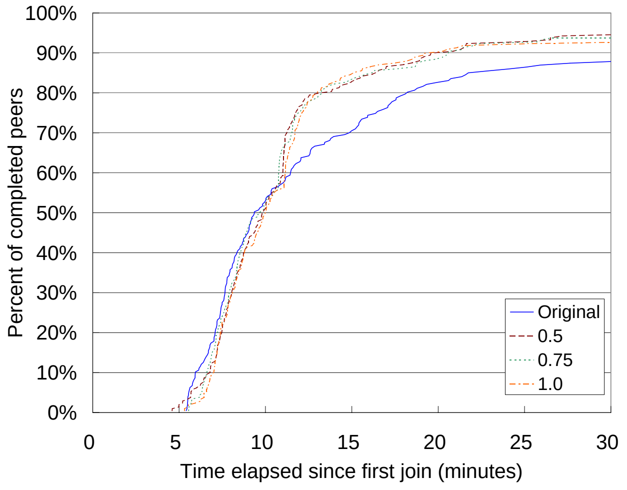

Figure 3: Peer selection by matching upload capacity - Percent of completed peers over time elapsed

Table 4: Peer selection by matching upload capacity - Average download time for various peer selection schemes

Selection scheme	Average download time (min)
Original	9.06
0.5	9.01
0.75	8.26
1.0	8.58

Table 5: Peer selection by matching upload capacity to download capacity - Average download time for various peer selection schemes

Selection scheme	Average download time (min)
Original	10.21
0.5	8.22
0.75	8.41
1.0	8.39

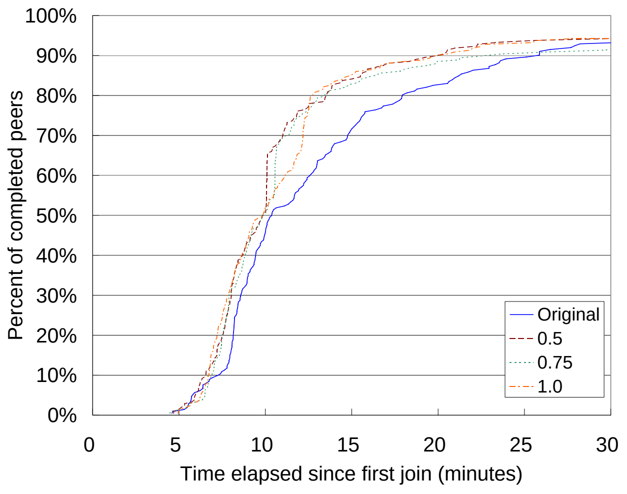

Figure 4: Peer selection by matching upload capacity to download capacity - Percent of completed peers over time elapsed

Table 6: Peer selection by matching locality - Average download time for various peer selection schemes

Selection scheme	Average download time (min)
Original	11.64
0.5	10.30
0.75	9.89
1.0	10.06

If the total number of peers is less than 50, which is the number of peers requested, all peers are returned without matching. When the number of peers is just slightly more than 50, obviously the effect of matching is not significant. It is evident in the similarity of results at the beginning 10 minutes.

5.3.3 Transient Period

The decrease in average download time for the matching schemes are shown in Tables 3, 4, 5 and 6. The increase in efficiencies are conservative since they include the transient period, where no improvement is made. In the transient period, the number of peers in the system is limited. Consequently, the neighbours given to the requesting peer are not well matched. As a result, the downloading time of those leechers are unimproved.

5.3.4 Results Comparison

Table 7 shows the percentage of speedup of different schemes when comparing to the unmodified tracker. Matching upload capacity to download capacity performs the best. In our tests, at the time matched neighbours are given out to the requesting peers, those requesting peers are already seeds. In the BT protocol, seeds selectively upload to leechers which can download the fastest from the seed. As a result, matching upload capacity to download capacity can better utilize seeds' uploading bandwidth and lowering the

Table 7: Speedup of various peer selection schemes under different random levels compared to the original implementation

Peer selection scheme	Speedup			
		Fraction of same-group peers		
	Original	*0.5*	*0.75*	*1.0*
Capacity (download)	0.00%	12.75%	14.71%	20.52%
Capacity (upload)	0.00%	0.60%	8.82%	5.27%
Capacity (download / upload)	0.00%	19.49%	17.64%	17.79%
Locality	0.00%	11.49%	15.04%	13.59%

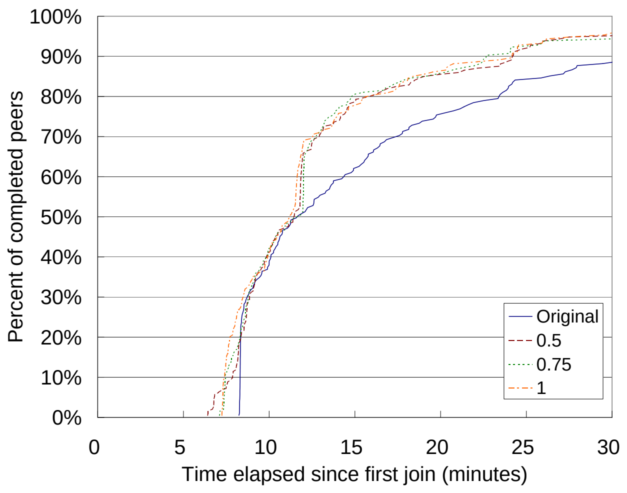

Figure 5: Peer selection by matching locality - Percent of completed peers over time elapsed

average download time.

5.4 Lessons Learned

The NetGeo program we used to get the latitude and longitude of nodes used the WHOIS record of a hostname to determine location. The WHOIS record yields the location of the administrative office for the organization or person that owns that domain. For small ISPs, the WHOIS record usually provides an accurate location. However, for large ISPs, which has routers all over a large country, the WHOIS record is necessarily sometimes inaccurate.

After using the PlanetLab nodes for a short while, we quickly noticed that it did not have robust support for deploying and running programs. After the slivers[2] were created on the PlanetLab nodes, it was up to the user to manage file transfer and invoking commands over hundreds of nodes. Even though some prior users contributed utilities such as parallel ssh and parallel scp [3], they were not flexible enough for our needs. For example, they do not provide a retry mechanism for failed nodes. There is also a maximum number of parallel ssh sessions allowed (around 100) that is too small for us. It took us a great deal of time experimenting with shell scripts and manual data manipulation to invoke commands and collect data.

[2]A Linux VServer on a PlanetLab node that is under the control of a PlanetLab user.

Another hurdle we had to cope with was that the PlanetLab nodes and the connections to these nodes were not very stable. It was very possible that one node that could be connected successfully just a few seconds ago became unresponsive for a few minutes. This created huge synchronization problems. For example, we could not assume that a file synchronization command (via ssh) indeed distributed the latest scripts to all nodes intended. Therefore, a subsequent call to these scripts may fail unexpectedly. Our solution was to instruct the nodes to download the latest script from a web server (via wget) and execute it. Instead of the unreliable push approach, this reliable pull approach ensured that all responsive nodes have up-to-date files. We were then confident that the binaries were fresh; at the expense of higher communication overhead.

6. RELATED WORK

While neighbour selection on P2P systems are widely studied, very few of them focus on P2P systems using the TFT exchange strategy. We only present studies on neighbour selection aiming at the P2P systems with TFT policy here.

Simon G. M. Koo *et al.* propose a neighbor selection method using a genetic algorithm [8]. Rather than randomly assigning peers, they distribute peer sets based on content availability estimated from the reported number of bytes downloaded. Content availability refers to the presence of data pieces among different peers. They assume the uplink throughput of a peer in BT is limited only by content availability and network capacity. It is believed that having neighbors with the most mutually disjoint collection of content pieces can assure maximum content availability and improve the overall throughput of the system. The goal of their algorithm is to maximize the number of content pieces each peer can contribute to its neighbors. They have divided the peers into 3 groups by network capacity: Class 1, 10 Mbps bi-directional link; Class 2, 128 kbps uplink / 1.5 Mbps downlink; and Class 3, 56 kbps bi-directional link. The simulation results of their algorithm, GA100, are shown in Table 8.

7. CONCLUSIONS AND FUTURE WORK

To further investigate the effects the proposed schemes have on sustainability, BT clients need to leave the system after being active for a while. This simulates the transient behaviour of peers. Realistically, many peers leave the system shortly after the file transfers are completed. This decreases the number of seeds and number of available pieces accessible by a peer. However, in order to simulate this behaviour

Table 8: Average download time (in minutes) of peers of different capacity. Variances are shown in parenthesis

Peer capacity	Random	GA100
Class 1	43.78 (0.13)	44.70 (0.16)
Class 2	145.89 (0.27)	132.58 (0.27)
Class 3	3890.18 (77.81)	2909.47 (66.94)

on PlanetLab nodes, a distributed job control framework that is robust enough to deal with the unstable nature of PlanetLab nodes is needed. Remote job creation and termination must be controlled reliably by the framework. When this level of control is available, we can observe whether power law plays a role in sustainability under our optimization schemes as mentioned in Section 2.

It would be more interesting to rate-limit BT clients for the experiments instead of using the PlanetLab nodes' native download / upload rate. This would enable us to produce more consistent results because the capacity would fluctuate less due to sharing of bandwidth among PlanetLab users. However, BitTorrent 3.4.2 only allows limiting the upload rate. Having all peers with download rate higher than upload rate is not useful because this does not reflect real-life scenarios. In the future, it may be possible to use BitTorrent 4.x.x to limit both the download rate and upload rate. This newer major version of BitTorrent has been rewritten to include many new features. BitTorrent 4.0.1 offers the ability to control both the download rate and upload rate. Unfortunately, it contains a bug that prevented us from using it in our experiments.

Each set of tests spanned over two hours. To maintain consistency, we generally performed tests between the hours of 10PM and 2AM EDT. This was due to the fact that half of the PlanetLab nodes used were from North America. Due to time constraints, we did not have the luxury to perform tests repeatedly. However, by repeating the same tests many times and averaging results, we mitigate the effects of occasional network anomalies that may produce results that do not represent the general case. This is particularly an issue in the PlanetLab environment since resources are shared among many users and their behaviours are very unpredictable under congested situations. Furthermore, standard deviations and confidence intervals can be calculated with repeated measurements to produce results that are more statistically sound.

In this paper, we have demonstrated our attempt to improve overall system throughput by modifying the official BT tracker. The original BT implementation uses a random peer selection algorithm. Our proposed strategies use capacity and locality information to aid the peer-selection process. These optimizations are applicable to P2P systems using TFT exchange algorithm. Our work showed that system throughput can indeed be improved by using the more organized neighbour selection schemes. Furthermore, these optimizations affect only the trackers and are totally transparent to BT clients. Experiments were done on the inter-continental nodes provided by PlanetLab. This setup enabled us to mimic network topology and latency very similar to real-world scenarios. From the comparison in performance results, we found that matching the neighbour-requesting peer's upload capacity to a peer set of corresponding download capacity performed the best when the peer set were mostly seeds. It significantly reduces average download time by around 18%. For matching by capacity, the average download time under different degrees of randomness were similar. Using a higher degree of randomness can increase both the system sustainability and system throughput.

8. REFERENCES

[1] Netgeo - the internet geographic database. `http://www.caida.org/tools/utilities/netgeo/`.

[2] Planetlab: Home `http://www.planet-lab.org/`.

[3] pssh `http://www.theether.org/pssh/`.

[4] B. Cohen. Incentives build robustness in bittorrent. In *Proceedings of the 1st Workshop on Economics of Peer-to-Peer Systems*, Berkeley, CA, USA, June 2003.

[5] M. Izal, G. Urvoy-Keller, E. W. Biersack, P. A. Felber, A. Al Hamra, and L. Garces-Erice. Dissecting BitTorrent: five months in a torrent's lifetime. In *PAM'2004, 5th annual Passive & Active Measurement Workshop, April 19-20, 2004, Antibes Juan-les-Pins, France / Also Published in Lecture Notes in Computer Science (LNCS), Volume 3015, Barakat, Chadi; Pratt, Ian (Eds.) 2004, XI, 300p - ISBN: 3-540-21492-5*, Apr 2004.

[6] D. Qiu and R. Srikant. Modeling and performance analysis of bittorrent-like peer-to-peer networks. In *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 367–378, New York, NY, USA, 2004. ACM Press.

[7] S. Saroiu, P. K. Gummadi, and S. D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proceedings of Multimedia Computing and Networking, 2002*, San Jose, CA, USA, January 2002.

[8] K. Tamilmani, V. Pai, and A. E. Mohr. "swift": A system with incentives for trading. In *Proceedings of the 2nd Workshop on the Economics of Peer-to-peer Systems*, Cambridge, MA, USA, June 2004.

[9] X. Yang and G. de Veciana. Service capacity of peer to peer networks. In *Proceedings of INFOCOM, 2004*, Hong Kong, China, March 2004.