# The Effects of Underlying Physical Network Topologies on Peer-to-peer Application Performances

Shie-Yuan Wang
Department of Computer Science
National Chiao Tung University
Hsinchu, Taiwan
Email: shieyuan@cs.nctu.edu.tw

Chih-Che Lin
Department of Computer Science
National Chiao Tung University
Hsinchu, Taiwan
Email: linjc@cs.nctu.edu.tw

Chao-Chan Huang
Department of Computer Science
National Chiao Tung University
Hsinchu, Taiwan
Email: cchuang@cs.nctu.edu.tw

***Abstract*—In this decade, peer-to-peer (P2P) applications have emerged as popular tools for sharing data and they have gained much attention. Such P2P applications build overlay networks on top of existing infrastructure networks and disseminate data in a peer-to-peer manner. How well the P2P overlay network maps onto the underlying physical network topology can significantly impact P2P application performances.**

**In the literature, little work has quantitatively studied the effects of underlying physical network topologies on P2P application performance. In this paper, we used simulations to evaluate the effects of underlying physical network topologies on the performance of the real-life BitTorrent protocol and application. Our results show that underlying physical network topologies significantly affect P2P application performances.**

## I. Introduction

In this decade, peer-to-peer (P2P) applications have emerged as popular tools for sharing data and they have gained much attention. Such P2P applications build overlay networks on top of existing wired and wireless infrastructure networks and disseminate and share data in a peer-to-peer manner. Due to the complicated behaviors of P2P applications, simulations are frequently used to study the performances of P2P applications. In general, P2P simulations can be classified into two categories.

The first category is packet-level network simulation, which simulates P2P networks with the details of underlying network protocols (e.g., the behaviors of the transport layer, the network layer, the MAC layer, etc.) and the physical network topology. This simulation approach simulates the communication network at a fine-grain level; thus, the simulation speed and simulation scalability of a packet-level P2P network simulator are limited. Representative packet-level P2P network simulation tools include ns-2 [1], Parallel/Distributed NS (PDNS) [2], and NCTUns [3].

The second category is message-level P2P network simulation, which virtualizes all the communication and network details between peer nodes and generates simulation results mainly based on messages generated by the top-level P2P protocols. Many message-level P2P network simulators have been proposed, such as P2PRealm [4], NeuroGrid [5], P2PSim [6], PeerSim [7], QueryCycle [8], 3LS [9], GPS [10], Oversim [11], etc.

The main objective of the message-level P2P network simulators is to conveniently observe and evaluate the behaviors and performances of P2P protocols in a large-scale network, rather than observe P2P protocol performances over realistic network topologies, conditions, and settings. For this reason, in these simulators a physical path (which may traverse many physical links) between two P2P peer nodes is usually abstracted as a physical link with a dedicated bandwidth, and the details of packet transmission/reception on such a physical path are usually simulated by moving a message from one peer node to another on the abstracted link.

However, in a real-life network the available bandwidth between two peer nodes is not fixed and usually cannot be easily estimated. For example, multiple P2P hosts may be located in the same campus, which connects to the Internet via a physical link. In such a condition, the connections set up between them and the P2P peer nodes on the Internet will need to share the bandwidth of the physical link that connects the campus to the Internet. The available bandwidth that each such connection can achieve is varied with the number of such connections and even time-varied with the amount of dynamic traffic on such connections. The assumption that each such connection has a fixed and exclusive bandwidth, which is made in many message-level P2P network simulations, is incorrect.

From the above explanations, one knows that the mismatch between the logical network topology constructed by P2P applications and its underlying physical network topology can have significant impacts on P2P application performance [13][14]. In the literature, little work has quantitatively studied the effects of underlying physical network topologies on P2P application performance. In this paper, we evaluated such effects on the downloading performance of the BitTorrent (BT) protocol using the NCTUns network simulator [3] and a real-life BitTorrent application [18]. Our simulation results show that underlying physical network topology significantly affects P2P application performances. This finding indicates that the performance results obtained from a message-level

978-1-4244-8016-6/10/$26.00 ©2010 IEEE

P2P network simulation can be too ideal and overestimated, when verified with those obtained from a packet-level P2P network simulation.

The remainder of this paper is organized as follows. In Section II, we survey related work. In Section III, we briefly explain the operation of the BT protocol, which is used in our simulation experiments. In Section IV, we present the results of our simulation experiments. Finally, the conclusion is drawn in Section V.

## II. Related Work

In [14], Ripeanu et al. measured and analyzed the logic networks constructed by the Gnutella application. They also estimated how well the clusters of the logic network constructed by the Gnutella application match the clusters of the underlying physical network (i.e., the autonomous systems constructed by ISPs). The authors first defined the entropy of a set C that contains $|C|$ hosts, each labeled with a domain name out of $n$ different domains, as follows:

$$E(C) = \sum_{i=1}^{n}(-p_i log(p_i) - (1-p_i)log(1-p_i)), \qquad (1)$$

where $p_i$ is the probability of randomly picking a host with a domain name $i$. They then defined the entropy of a clustering of a graph $G$ that is of size $|C|$ and divided into $k$ clusters as follows:

$$E(C_1, C_2, ...C_k) = \sum_{i=1}^{k} \frac{|Ci|}{|C_1| + |C_2| + ...|C_k|} * E(C_i), \qquad (2)$$

where $C_i$ denotes the $i$-th cluster of $G$. The clustering algorithm for merging hosts in the logic network constructed by Gnutella can be found in [14]. According to the real measurements made in [14], the entropy of the clustering of the logic network constructed by Gnutella is similar to that of the clustering constructed in a random manner (i.e., the $E(C)$ computed by Equation 1).

Such results suggest that the logic network constructed by the Gnutella application is unrelated to the Internet structure. Inspired by this finding, several recent works (e.g., [15], [16]) have designed new schemes to reduce network traffic across different ISPs because such traffic is considered expensive.

Although previous works have noticed the mismatch between the logic network and the physical network [13][14], so far, to the best of the authors' knowledge, little work has studied the effects of physical network topologies on P2P application performance. For this reason, in this paper we studied the performance of a real-life BitTorrent application on different physical network topologies. Quantitative simulation results are presented in this paper.

## III. Brief Introduction to the BitTorrent Protocol

In this paper, we used the real-life BitTorrent application [18] to evaluate the effects of underlying physical network topologies on P2P application performances. The operation of the BitTorrent protocol is briefly explained here. More details of this well-known P2P protocol can be found in [13].

In the BitTorrent protocol, nodes are divided into three types: 1) tracker server, 2) seeder, and 3) client. The tracker server maintains the list of active clients that are uploading and downloading files to be shared. The seeder is a client that possesses complete copies of files to be shared. The seeder that owns the first copy of the shared file is called the initial seeder. The last type is the client, which denotes the nodes that do not possess the shared file and intend to download that file.

For a client that intends to download a shared file $F$, it should first obtain the torrent file of the shared file $F$ (e.g., by downloading the torrent file from the web site where the initial seeder published the torrent file). The torrent file contains the meta-data describing the file $F$ and the IP address of the BitTorrent tracker server that maintains the active peer list of the file. After obtaining the torrent file, the client transmits a control message to the tracker server specified in the torrent file to request the active peer list of $F$.

On receiving such a request, the tracker server usually chooses a subset of the peer nodes that are downloading/uploading the contents of $F$ in a random manner and replies the client with a list composed of the chosen nodes. By using the information in the active peer list, the client then tries to connect to other active clients that are uploading/downloading the same shared file. (The client usually chooses peers to download $F$ in an arbitrary manner [15].) After establishing connections with those clients, the client can obtain the blocks of $F$ from the connected clients in a peer-to-peer manner. At the same time, the client can upload the blocks of $F$ that it possesses to other peer nodes that have not possessed them. Such a block sharing process repeats until the client has possessed all of the blocks of $F$.

## IV. Performance Evaluation

### A. Simulation Setting

In this section, we evaluate the performance of the BitTorrent protocol using NCTUns. NCTUns is a powerful packet-level network simulator, which has two unique features. One is that it uses the real-life TCP/IP protocol stack in the Linux kernel to conduct simulations and the other is that it allows real-life application programs to be run on it during simulation. By exploiting these two features, we ran the real-life BitTorrent software [18] over different simulated physical network topologies to evaluate the effects of these topologies on BitTorrent performances.

The simulations were conducted on a desktop computer equipped with the Intel Core2 Duo 3-GHz processor and 3 GB main memory. The main parameter setting for each simulation case is described here: the bandwidth of each simulated link is set to 1 Mbps and the propagation delay of each simulated link is set to 10 ms. It is assumed that no bit errors occur on each simulated link. The routes of a simulated network is pre-determined by using the Dijkstra's shortest path algorithm.

In our simulation experiments, each BitTorrent client program randomly selects $N$ of the peers indicated by the tracker

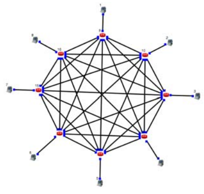

Fig. 1. Fully-connected 8-host topology

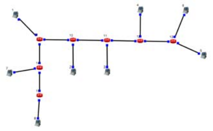

Fig. 2. Example 8-host topology

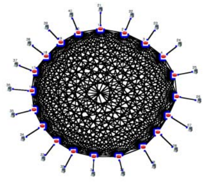

Fig. 3. Fully-connected topology

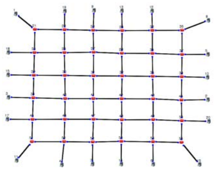

Fig. 4. The mesh topology

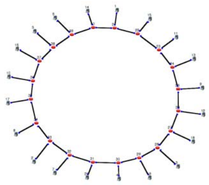

Fig. 5. The ring topology

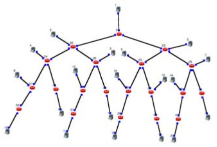

Fig. 6. The tree topology

program to establish $N$ connections, where $N$ denotes the maximum number of connections that a BitTorrent program is allowed to establish. In a simulation run, the set of all BitTorrent client programs and their established connections is referred to as the logic network constructed by the BitTorrent protocol. Such a logic network will be mapped onto a underlying physical network topology in a simulation case.

Six underlying physical network topologies are used to study their effects on the performances of the BitTorrent protocol. The first one is the fully-connected 8-host topology, as shown in Fig. 1. This topology comprises 8 hosts, each connecting with a router via a link. Each router is connected with all other routers via dedicated non-sharing links. The second physical topology is a small network comprising 8 hosts, which is shown in Fig. 2. The third physical network topology is the fully-connected topology, as shown in Fig. 3. In this topology, there are 20 hosts, each connecting with a router via a link. Each router is connected with all other routers via dedicated non-sharing links. The fourth physical network topology is the mesh topology, which is shown in Fig. 4. In this topology, 36 routers form a core grid network and each of the 20 edge routers connects with a host via a dedicated link. The fifth physical network topology is the ring topology. As shown in Fig. 5, 20 routers together form a ring network and each of them connects with a host via a dedicated link. The last physical network topology is the tree topology, which is shown in Fig. 6. In this topology, 20 routers form a binary tree structure. Each of these routers connects with a host via a dedicated link.

In a simulation case (regardless of the underlying physical network topology), each host runs a BitTorrent client program on it. Host 1 also runs a BitTorrent tracker program on it and possesses the complete file that is to be shared. The maximum queue size of each output queue associated with a physical link is set to 50 packets. For each simulation case, it is run ten times with different random number seeds. The performance reported for a simulation case in this paper is the average performance of its ten different runs.

### B. Simulation Results

Fig. 7 shows the average download time of all BitTorrent clients with different amounts of background traffic in the topologies shown in Fig. 1 and Fig. 2. In each of these two topologies, a background UDP flow generates traffic at a specified constant bit rate. The source node of the background traffic flow is randomly chosen from the nodes at the bottom left part of the network and its destination node is randomly chosen from the nodes at the top right part of the network. The download time results in Fig. 7 shows that the network topology greatly affects the behaviors of P2P applications (not only the absolute performance values but the performance trend).

In the fully-connected 8-host topology, each host has a non-shared link to each other host; thus, the background traffic does not affect the bandwidth available for each BT connection and the performances of the BT connections remain about the same regardless of the amount of the background traffic. However, in the example 8-host topology, the hosts share a common backbone link in the core network. In such a network, each BT connection has to compete for the bandwidth of the shared link with other connections and background traffic. As a result, the downloading time greatly increases as the bandwidth consumption of the background traffic increases.

These results reveal that the simple fully-connected graph, which is commonly used in the research of P2P applications, may not be adequate to study the performances of P2P applications in real networks where background and cross-ISP traffic usually exists and consumes the bandwidth of each ISP's core network. If the ideal fully-connected graph is used to study P2P application performance, one should be very careful to avoid obtaining misleading performance results and trends.

Fig. 8 shows the average time required for all BitTorrent clients to finish downloading the shared file versus the number of connections that they are allowed to establish. One obvious finding is that, when the number of allowed connections increases, the time for all BitTorrent clients to finish downloading the shared file significantly decreases. This result is expected because when the number of allowed connections increases, the number of connected peer nodes of a BitTorrent client increases. In this condition, a BitTorrent client can obtain more different blocks of the shared file from different peer nodes at the same time. Thus, the time required for a BitTorrent client to obtain all blocks of the shared file can be reduced.

A more important finding is that when the underlying physical network topology is changed, the time required for all BitTorrent clients to finish their downloads differs greatly. One first sees that the BitTorrent clients in the simulation cases that use the fully-connected graph as the underlying physical network topology on average need the least time to finish their downloads, as compared with those in simulation cases using other physical network topologies. The reason is that when the fully-connected graph is used as the underlying physical network topology, each link of the logic network constructed by BitTorrent clients is mapped onto a dedicated non-sharing physical link. In this condition, each logic connection between two peer nodes need not contend for link bandwidth with other connections. The fully-connected topology represents the ideal (but unrealistic) case for the underlying physical network topology. For simplicity, however, most message-level P2P network simulators use a fully-connected topology as the underlying physical network topology.

In contrast, when other common network topologies (such as the mesh, ring, and tree topologies) are used as the underlying physical network topology, many logic connections may need to be mapped onto the same physical link, which may result in traffic congestion on that physical link. In this condition, the quality (bandwidth, delay, loss) of a logic connection experienced by a BitTorrent client can vary greatly over time and is affected by the dynamic behaviors of other clients (e.g., packet transmission, peer node selection, and block selection). The effects of these different physical network topologies are compared and studied below.

Tab. I shows the ratios of the required download time in the three common physical network topologies to that in the fully-connected physical network topology. There are two findings worth noting. First, even in the cases where each BitTorrent client is allowed to establish connections with all other peers (i.e., the number of allowed connections is set to 20), the required download time in the three studied physical network topologies are at least 27% larger than that in the ideal fully-connected physical network topology. These results show that ignoring/simplifying the underlying physical network topologies, which is usually done by a message-level P2P network simulator, may overestimate the performances of P2P applications.

Another finding is that the underlying physical network topologies greatly affect the performance of the BitTorrent protocol. For example, when the number of connections that a BitTorrent client is allowed to establish is set to 10, the required download time in the mesh topology is on average 1.33 times larger than that in the fully-connected topology, while the required download time in the ring and tree topologies are on average 1.60 and 2.11 times larger than that in the fully-connected topology.

The reason why the BitTorrent clients on the three underlying physical network topologies achieve different performances is explained here. The BitTorrent clients in the mesh topology achieve the best performance because the mesh topology provides the most links in the core network among the three studied topologies. Suppose that each edge router is connected with only one end host and $n$ is the total number of the end hosts in the network. Then the mesh core network will have $(\frac{n+4}{4})^2$ routers and provide $(\frac{n+4}{4} - 1) * (\frac{n+4}{4}) * 2$ physical links for forwarding packets. In contrast, when the ring topology or the tree topology is used, the core network has only $n$ routers and provides $n$ physical links. As a result, the mesh topology provides more network bandwidth and flexibility to distribute the traffic load of a P2P logic network over different physical links. This reduces the possibility of traffic congestion on physical links.

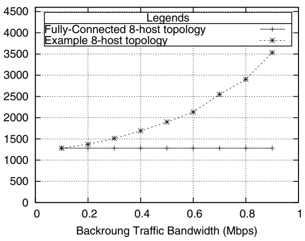

Fig. 7. The average simulated time required for all clients to finish downloading the shared file

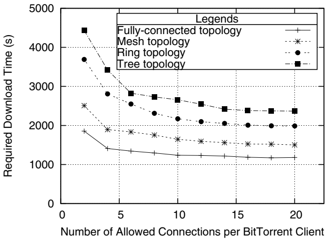

Fig. 8. The average simulated time required for all clients to finish downloading the shared file

The above explanations can be confirmed by Fig. 9 and Fig. 10, which show the probability distribution functions (PDF) of the number of logic connections passing through the same physical link, when the number of connections that a BitTorrent client is allowed to establish is set to 2 and 10, respectively. The corresponding cumulative distribution function (CDF) results are presented in the Appendix for readers' reference. The PDF and CDF results obtained when the number of connections that a BitTorrent client is allowed to establish is set to 20 (i.e., unlimited) are very similar to those when this value is set to 10. Therefore, we do not present them here for brevity.

As one sees, when the number of logic connections that a client can establish is set to 2, over 40% of the physical links in the mesh topology have 5 and less logic connections passing through them and no physical link in the mesh topology has more than 45 logic connections passing through it. In contrast, most of the physical links in the ring topology accommodate 40-55 logic connections. In the tree topology, on average 25.64 % of the physical links accommodate more than 50 logic connections and 6.13 % of the physical links accommodate 95 or more logic connections. These results evidence that the logic connections in the mesh topology can be distributed over more physical links, which reduces traffic congestion on each physical link.

For the ring topology, due to its circular and symmetric structure, the numbers of logic connections mapped onto the physical links in the ring topology are roughly the same. In the tree topology, because there is only one path between any pair of peer clients, the physical links connecting with the routers that are near the tree root will need to accommodate more

TABLE I
THE RATIOS OF THE REQUIRED DOWNLOAD TIME IN THE MESH/TREE/RING TOPOLOGY TO THE REQUIRED DOWNLOAD TIME IN THE FULLY-CONNECTED TOPOLOGY

| | Number of allowed connections | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 | 18 | 20 |
| Mesh | 1.35 | 1.34 | 1.37 | 1.36 | 1.33 | 1.29 | 1.28 | 1.29 | 1.30 | 1.27 |
| Ring | 1.99 | 1.99 | 1.87 | 1.79 | 1.60 | 1.53 | 1.40 | 1.37 | 1.42 | 1.35 |
| Tree | 2.39 | 2.28 | 2.10 | 2.11 | 2.11 | 2.13 | 1.92 | 1.90 | 1.92 | 1.86 |

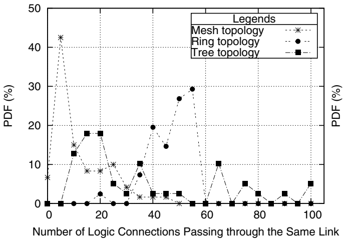

Fig. 9. The PDF of the number of logic connections passing through a physical link (the number of connections that a client can establish is set to 2)

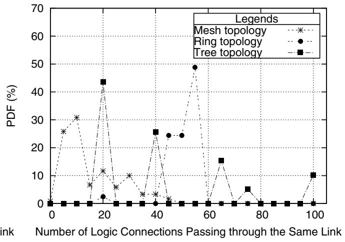

Fig. 10. The PDF of the number of logic connections passing through a physical link (the number of connections that a client can establish is set to 10)

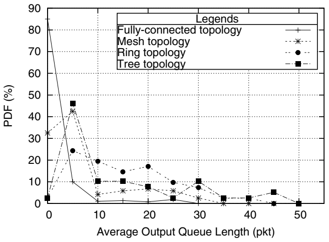

Fig. 11. The PDF of the average link output queue length (the number of connections that a client can establish is set to 2)

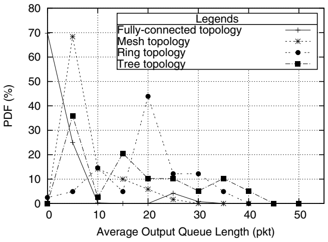

Fig. 12. The PDF of the average link output queue length (the number of connections that a client can establish is set to 10)

logic connections. As a consequence, traffic congestion may frequently occur on these physical links. This explains why in the tree topology several physical links have a large number of logic connections passing through them.

To observe how traffic load is distributed over the underlying physical network topology, we plotted the PDF of the average output queue lengths of physical links in Fig. 11 and Fig. 12. The average queue length results confirm our explanations for the impacts of the three physical topologies on P2P application performances. One can see that the BitTorrent clients in the fully-connected topology generate the least buffer occupancy, while those in the tree topology result in several bottleneck links (with very high buffer occupancy). As a result, BitTorrent clients in the fully-connected topology achieve the best download performance and those in the tree topology achieve the worst download performance. The BitTorrent clients in the mesh and ring topologies generate moderate average buffer occupancy; thus, the BitTorrent clients in these two physical topologies generate moderate download performances, as compared with those in the fully-connected and tree topologies.

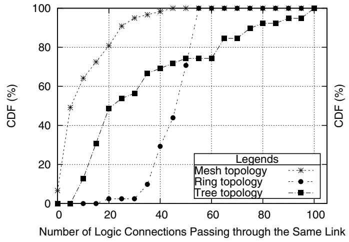

Fig. 13. The CDF of the number of logic connections passing through a physical link (the number of connections that a client can establish is set to 2)

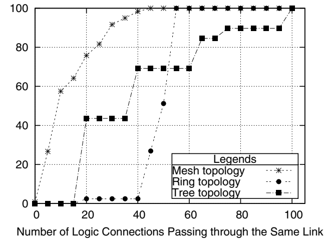

Fig. 14. The CDF of the number of logic connections passing through a physical link (the number of connections that a client can establish is set to 10)

## V. CONCLUSION

In this paper, we studied the effects of different underlying physical network topologies on the performance of the BitTorrent protocol using the NCTUns network simulator and a real-life BitTorrent application. Quantitative performance results of the BitTorrent application on different underlying physical network topologies are presented. Our simulation results show that underlying physical network topologies significantly affect P2P application performances. These results suggest that P2P performance results obtained on an abstract physical network topology (which is usually used by a message-level P2P network simulator) can be too ideal and overestimated than those obtained on a more realistic network topology (which is normally used by a packet-level network simulator).

## APPENDIX

For readers' reference, the CDFs of the number of logic connections passing through the same physical link, when the number of connections that a BitTorrent client is allowed to establish is set to 2 and 10, are plotted in Fig. 13 and Fig. 14, respectively.

## REFERENCES

[1] The official website of the ns-2 network simulator *"http://www.isi.edu/nsnam/ns/"*.

[2] The official website of the Parallel/Distributed NS (PDNS) *"http://www.cc.gatech.edu/computing/compass/pdns/index.html"*.

[3] S.Y. Wang, C.L. Chou, C.H. Huang, C.C. Hwang, Z.M. Yang, C.C. Chiou, and C.C. Lin, *The Design and Implementation of the NCTUns 1.0 Network Simulator,* Computer Networks, Vol. 42, Issue 2, June 2003, pp. 175-197.

[4] N. Kotilainen, M. Vapa, T. Keltanen, A. Auvinen, J. Vuori, *P2PRealm — peer-to-peer network simulator,* Proceedings of the 11th Intenational Workshop on Computer-Aided Modeling, Analysis and Design of Communication Links and Networks, 2006, (CAMAD 2006), p.p. 93-99, June 8-9, Trento, Italy.

[5] Sam Joseph, *An Extensible Open Source P2P Simulator,* P2P Journal, 2003.

[6] The official website of the P2PSim network simulator *"http://pdos.csail.mit.edu/p2psim/index.html"*.

[7] The official website of the PeerSim network simulator *"http://peersim.sourceforge.net/"*.

[8] Mario T. Schlosser, Tyson E. Condie, Ar D. Kamvar, *Simulating a P2P file-sharing network,* Proceedings of the first Workshop on Semantics in P2P and Grid Computing, 2002.

[9] N.S. Ting, R. Deters, *3LS — a peer-to-peer network simulator,* Proceedings of the 3rd International Conference on Peer-to-Peer Computing, 2003. (P2P 2003), p.p. 212-213, 1-3 Sept, 2003, Linkping, Sweden.

[10] W Yang, N Abu-Ghazaleh, *GPS: a general peer-to-peer simulator and its use for modeling BitTorrent,* Proceedings of the 13th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2005, (MASCOTS 2005), pp. 425-432, Sept. 27-29, Atlanta, Georgia, USA.

[11] The official website of the OverSim network simulator *"http://www.oversim.org/"*.

[12] Q. He, M. Ammar, G. Riley, H. Raj, R. Fujimoto, *Mapping peer behavior to packet-level details: a framework for packet-level simulation of peer-to-peer systems,* Proceedings of the 11th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer Telecommunications Systems, 2003 (MASCOTS 2003), p.p. 71-78, Oct. 12-15, 2003, Orlando, Florida, USA.

[13] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, *A Survey and Comparison of Peer-to-Peer Overlay Network Schemes,* IEEE Communications Surveys and Tutorials, Vol. 7, Issue 2, pp. 72-93, 2005.

[14] M. Ripeanu, I. Foster, and A. Iamnitchi, *Mapping the Gnutella Network: Properties of Large-Scale Peer-to-Peer Systems and Implications for System Design,* in IEEE Internet Computing Journal, vol. 6(1) 2002.

[15] D. R. Choffnes and F. E. Bustamante, *Taming the torrent: a practical approach to reducing cross-isp traffic in peer-to-peer systems,* in Proceedings of the ACM SIGCOMM 2008 conference on Data communication, Seattle, WA, Aug 2008.

[16] H. Xie, Yang, R., Krishnamurthy, A., Liu, Y., and Silberschatz, A. *"P4P: Provider portal for applications,"* in Proceedings of the ACM SIGCOMM 2008 conference on Data communication, Seattle, WA, Aug 2008.

[17] S. Sen and J. Wang, *Analyzing peer-to-peer traffic across large networks,* in Proc. of the 2nd ACM SIGCOMM Workshop on Internet measurment, p.p. 137-150, Marseille, France, 2002.

[18] BitTorrent, the official site: *"http://www.bittorrent.com/."*