An Efficient Search Scheme Based on Perfect Difference Graph for P2P Networks

Chen-Wei Wang and Yaw-Chung Chen
Department of Computer Science
National Chiao Tung University
Hsinchu, Taiwan
{iverson2220@yahoo.com.tw, ycchen@cs.nctu.edu.tw}

***Abstract* - We propose an efficient search scheme for multi-layer unstructured P2P systems, and show that it is not only reliable, but also scalable. To the best of our knowledge, there are few researches focusing on the reliable and scalable search mechanism for unstructured P2P systems. The broadcasting performance of the P2P system is enhanced through the use of a Multi-hop Index Replication with Perfect Difference Graph (PDG) forwarding algorithm, which makes certain that each super-peer receives just one copy of the broadcast message. Furthermore, by using the Multi-hop Index, a super-peer has extra information to know whether the queried file is available or not. The experimental results show that our proposed scheme improves existing unstructured P2P systems in terms of a higher query success ratio, fewer query flooding messages and shorter average delays. In other words, our proposed scheme achieves high scalability, low communication overhead and improved performance of query responses.**

Keywords: Perfect Difference Graph; P2P systems; Multihop-index

I. INTRODUCTION

Structured P2P overlay networks can provide efficient and accurate query service but it needs a large overhead to maintain the distributed hash table (DHT) and suffers peer churn. Measurement studies of deployed P2P overlays [19][20] show a high rate of churn. Unstructured P2P overlays use flooding or random walks to look up data items. It is resilient to churn, but its queries may generate a large volume of traffic and cause systems un-scalable. Although queries by using random walk reduce the traffic volume, huge traffic overhead still occurs when the requested resource does not exist. Unfortunately, this situation happens often. In [7] authors observed that roughly a half of the queries cannot be matched. A solution to reduce the query traffic is setting the time-to-live (TTL) of query packets to a small value. However, this approach will search only a small portion of the peers in the system and queries are very likely failed even if the requested resource does exist. Measurement studies on actual unstructured P2P networks observed that the ratio of successful query is typically around 10% [21].

In this paper, we propose Multi-hop Index Replication that can improve search performance for rare objects. Index replication features not only much lower overhead compared with data replication, but also more effective for improving the scalability of unstructured networks [6][11][20]. Our Multi-hop Index Replication with PDG forwarding algorithm eliminates the impacts of redundant query flooding messages and reduces the traffic in searching unavailable files. We use a so-called bootstrap peer (BSP) to maintain a super-peer table, a peer joining the P2P network and wishing to become a super-peer must first send a request to the BSP. After examining the requesting peer's bandwidth conditions, the BSP may select the peer as a super-peer, and send it the corresponding connection information or register the peer as a redundant super-peer.

When the overlay topology is established, a pure PDG [8][9] forwarding algorithm can be used to transport the query messages from the originating super-peer to other super-peers in such a way that each super-peer receives just one copy of the message. In addition, each super-peer has to maintain an AVL tree-based index that is constructed with a randomly generated key. The average-case complexity of search, insert, and delete operations is O(log n), where n is the number of shared files in the overlay network.

The rest of the paper is organized as follows. In Section II we present the background related to both structured P2P and unstructured P2P systems. In Section III, we discuss the proposed hybrid P2P system with analytic models. In Section IV, we evaluate the effectiveness of our methods and discuss the performance of the system. We conclude the work and address the future issues in Section V.

II. BACKGROUND

Peer-to-peer (P2P) architecture is an alternative to the traditional client/server architecture. P2P makes it possible for users to organize themselves into ad hoc groups that can efficiently and securely handle requests, share resources, collaborate, and communicate. As P2P systems evolve, we can anticipate the emergence of a wide variety of online communities [18].

A. P2P File Sharing Applications

Many P2P systems have been proposed for different applications [1]-[6], [10]-[14]. In this paper, we focus on P2P overlays for efficient data (file) sharing among peers. The Content Addressable Network (CAN) [15] was proposed for file sharing and the entire space is partitioned to distinct zones such that each peer is in charge of one zone. Every peer maintains a routing table which holds the IP address of its neighbors in the coordinate space. The data is stored into and retrieved from the peer that owns the zone. CAN takes advantage of the ordering of the Cartesian coordinate space in the routing protocol.

Chord [10] organizes the node keys into a so–called chord ring, where each peer is assigned an ID. Peers are inserted into the ring according to the order of their IDs. Each peer has a successor and a predecessor. To accelerate the search, each peer maintains a finger table, in which each finger points to a peer with a certain distance from the current peer. Compared to CAN, Chord is simpler as the key is hashed into a one dimensional space.

Gnutella [12] is a decentralized unstructured peer-to-peer overlay, in which peers join the system based on loose rules. To look up a data item, a peer sends a flooding search request to all neighbors within certain radius. In Gnutella, flooding method consumes a lot of network bandwidth and hence the system is not scalable. Also, it is usually hard to find a rare data item because it is unlikely to flood the search request to all peers. The study in [18] improved the efficiency while looking up a rare data item.

BitTorrent[13] is a centralized unstructured P2P network. It uses a central server called tracker to keep track of peers in which files are stored. The tracker records the network location of each client either uploading or downloading the file associated with a torrent. Each file has a corresponding torrent file stored in the tracker. Upon receiving a download request, the tracker sends back a random list of peers which with the same file. BitTorrent suffers single point of failure problem at the central server.

YAPPERS [15] combines both structured and unstructured P2P overlays to provide a scalable search service over an arbitrary topology. It is designed for efficient partial search which only returns partial values of data. For a complete search, YAPPERS still needs to flood the query to all peers which are in the same color as the data. Compared to YAPPERS, our proposed multi-layer unstructured P2P system can further improve the accuracy of the lookups in a more efficient way.

In [16], the structured overlay was used to support unsuccessful flooding data search, however, its structured overlay is responsible for connecting all the unstructured overlays and transferring query requests between them.

B. *The P2P Lookup Problem*

In addition to single point of failure and poor scalability, a drawback of centralized approach is the vulnerability to malicious attacks and legality issue. These shortcomings led to the adoption of decentralized solutions, which make it difficult to ensure high performance and availability, so a high degree of redundancy is required.

III. PROPOSED APPROACHES

Neither structured P2P nor unstructured P2P networks alone can fulfill the requirements of efficiency, scalability, and reliability of services. The motivation of this study is to combine both types of P2P networks to provide a hybrid approach which can offer better efficiency and scalability.

A. *Super-Peer Overlays and Forwarding Protocols*

The super-peers overlay topology can be constructed as a graph, in which vertices represent individual super-peers while undirected edges stand for connections between super-peers. As shown in Table I, each peer in perfect difference graph (PDG) [8] has a degree $O(\sqrt{n})$, thus the topology is more flexible than $O(n)$ in the complete graph. Furthermore, the search range of a PDG-based topology is similar to that in the complete graph-based topology. Table I shows that other graph topologies have both a lower peer connection degree and a larger diameter (i.e. no. of hops along the path) than the PDG approach. Thus, the PDG-based overlay topology is a better choice for the hybrid P2P system presented in this paper.

1) *Perfect difference graph*:

According to the definition of perfect difference sets (PDSs), PDG provides the mathematical knowledge for achieving this optimum number of peers to construct the framework of perfect difference networks or PDNs as defined bellow:

Definition 1: Perfect Difference Network (PDN) — there are $n = \delta^2 + \delta + 1$ nodes, numbered 0 to n-1. Node i is connected to node $i \pm 1$ and $i \pm s_j (mod\ n)$, for $2 \le j \le \delta$, where s_j is an element of the PDS $\{s_0, s_1, \ldots, s_\delta\}$ of order δ.

Table II illustrates the number of peers, the number of elements and the order in the first ten PDSs. Fig. 1 shows a PDG overlay based on the PDS {1, 3}. Since there are two elements in the PDS, the graph has seven ($2^2 + 2 + 1 = 7$) peers. For example, peer 0 has edges connecting to peers (0 ± 1) mod 7 and (0 ± 3) mod 7, which are peer 1, 3, 4 and 6.

For example, in Fig. 1, we present a brief description of the forward edges of peer 0, which are the edges connecting peer 0 to peer 1 and 3, respectively, and the backward edges are the edges connecting peer 0 to peer 4 and 6, respectively.

TABLE I. COMPARISON OF VERTEX DEGREE AND GRAPH DIAMETER.

Order of vertex degree	Graph diameter	Example network
$O(n)$	1	Complete Graph
$O(\sqrt{n})$	2	Perfect Difference Graph
$O\left(\frac{\log n}{\log\log n}\right)$	$\theta\left(\frac{\log n}{\log\log n}\right)$	Star, Pancake
$O(\log n)$	$\log n$	Binary Tree Hypercube
$O(1)$	$n/2$	Ring

TABLE II. RELATION BETWEEN NUMBER OF SUPER-PEERS N, ORDER Δ AND PDS

N	δ	PDS $\{s_1, s_2, \ldots, s_\delta\}$
7	2	{1,3}
13	3	{1,3,9}
21	4	{1,4,14,16}
31	5	{1,3,8,12,18}
57	7	{1,3,13,32,36,54,63}
73	8	{1,3,7,15,31,36,54,63}
91	9	{1,3,9,27,49,56,61,77,81}
133	11	{1,3,12,20,34,38,81,88,94,104,109}
183	13	{1,3,16,23,28,42,76,82,86,119,137,154,175}
273	16	{1,3,7,15,63,90,116,127,136,181,194,204,233,238,255}

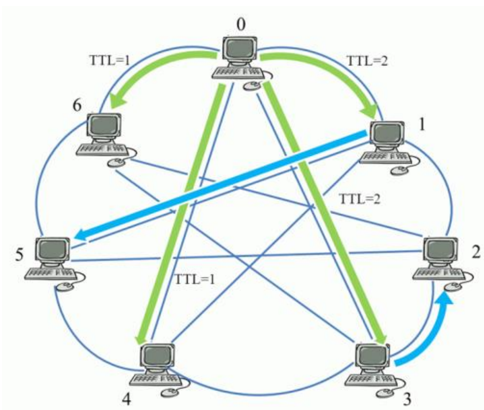

Figure 1. An example of PDG-based forwarding algorithm.

2) Broadcasting over a Super-peer Overlay

We deploy a PDG-based forwarding algorithm [9] in which the query requests are delivered to all super-peers in the overlay via the forward and backward edges of the perfect difference network. Each super-peer will send the search requests by using the PDG forwarding algorithm so that each super-peer receives only one copy of the search requests. The PDG forward query message in two steps:

- **Step 1**: Super-peer *i* sends a request message with TTL=2 to all of its forward partners and sends a request message with TTL=1 to all of its backward partners.
- **Step 2**: If an intermediate super-peer receives the request message, it duplicates the message to all of its backward partners other than the partner from which it received the original query message.

Fig. 1 illustrates the PDG-based forwarding algorithm for a super-peer overlay forming a PDG with an order of $\delta = 2$. Here super-peer 0 wants to send a query to all other super-peers. According to the above two steps, super-peer 0 sends a query request with TTL=1 and 2 by its forward and backward edges to partners {1, 3} and {4, 6}, respectively. In the case of TTL=2, the TTL value is reduced to 1, and partners {1, 3} forward a copy of the query request to all their backward peers other than the edge from which they received the query message. In the case of TTL=1, since the TTL value is reduced to zero, partners 4 and 6 take no further action.

3) Multi-hop Index Replication

Since each super-peer maintains the index replication in AVL tree, so that lookup, insertion, and deletion all take O(log *n*) time in both the average and worst cases, where *n* is the number of shared files in the overlay.

To ease the discussion, we explore two-hop index replication strategy in which each ordinary peer sends the name of shared files to all of its one-hop super-peer, which maintains the index replication in AVL tree. The index is constructed with a randomly generated key that is the name of shared files published by the ordinary peers through the SHA1-like algorithm. These hashed keys are inserted to the AVL tree-based index. Each super-peer broadcasts the available resource names by using PDG algorithm. In order to further reducing the broadcast messages, each super-peer uses 1-bit to record the status of the shared files. If the bit were set, the shared file would come from ordinary peers it controlled. Otherwise the file would be in other super-peers. By this way, only when the bit is set, does a super-peer sends the query messages to its ordinary peers. Otherwise, the super-peer will forward the lookup messages to other super-peers directly by PDG-based algorithm. Each super-peer can search the AVL tree to decide whether to broadcast messages to its ordinary peers or not. If we can't find any records in the AVL tree-based index, it means that there is no resource published by peers.

B. System Construction and Architecture

The bootstrap peer (BSP) uses a super-peer table to maintain the super-peer overlay structure. For convenience, we discuss only one bootstrap peer attached to the overlay network.

1) System Construction

In our example system, the ordinary peer can connect to two super-peers. Thus when one of the super-peers leaves or crashes, the other can still hold the records. When a new peer enters the overlay as an ordinary peer, it sends a request to the bootstrap peer. Upon receiving the joining request, BSP acknowledges the peer with a list containing the IP addresses of randomly selected super-peers. The peer then chooses a less loaded super-peer to establish a connection. Once the new peer connects to the super-peer, it becomes one of ordinary peers of that super-peer and super-peer sends it a peer list which contains the part of the ordinary peers in the same overlay. When the ordinary peer wants to leave the system, it simply sends a message to inform its super-peer, which then updates the corresponding AVL tree-based index to show that the shared files in the leaving peer no longer exist in the tree. Fig. 2 shows the overlay configuration in which a new peer is joining the P2P overlay. By multi-layer and multi-hop architecture, our system can serve as much powerful super-peer (MSP) with large storage space, computational capability and higher bandwidth to manage the whole PDN cluster, as shown in Fig. 3.

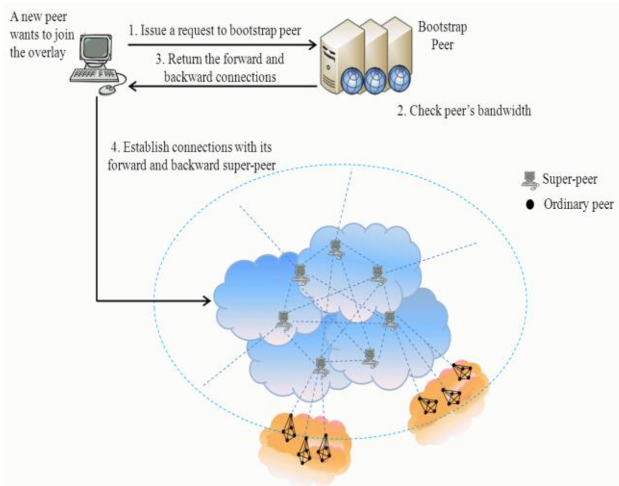

Figure 2. Configuration of a new peer joining the super-peer.

2) New Super-peer Joining

Any peer with high bandwidth entering the P2P overlay network sends a joining request with its bandwidth and IP information to the BSP. After checking the bandwidth quality, the BSP may accept the peer as a super-peer. When the number of super-peers is larger than the value ($\delta^2 + \delta + 1$), a newly joining peer that fulfills the bandwidth requirement will be marked by BSP as a redundant super-peer. When the total number of super-peers increases to a threshold, $1/2\,[(\delta^2 + \delta) + (l^2 + l)]$, the current PDS will be extended to the successor PDS order and the super-peer overlay will be extended accordingly.

In the initial set-up phase, the BSP utilizes a min-order PDS with order 2 to construct a basic super-peer overlay network for a maximum of 7 super-peers. Assume that 9 new peers fulfill the bandwidth requirements as a super-peer, since the number of new peers exceeds the number of available spaces in the overlay, the former 7 peers are assigned as super-peers, while the remaining peers are appointed as redundant peers. Later, when a newly coming peer wants to become a super-peer, it will result in the total number of super-peers, including active, new incoming, and redundant super-peer, exceeding the threshold 9(= (6+12)/2). The BSP then extends the super-peer overlay topology using a PDS with an order of 3, thus allowing for super-peers up to 13, including 10 active super peers in the newly extended configuration.

3) Super-peer Leaveing

When a super-peer is leaving the system, it sends a leave message to both the BSP and all of its ordinary children peers. The BSP selects one of the redundant super-peers to take place the leaving one. Then BSP sets the active state of the redundant peer and informs other active peers to update their partner records correspondingly.

Having received a leave message from a super-peer that tries to disconnect from the P2P overlay, each ordinary peers re-enter to the overlay by choosing one of the super-peers with the shortest response time in its super-peer list. When the number of super-peers is below the threshold ($\delta^2 + \delta + 1$), there will be no enough super-peers to take over the leaving peers in the overlay network, some of the super-peers lose their forward or backward partners. As a result, some of the super-peers may fail to receive the messages delivered by other super-peers in the overlay. To overcome this effect, when the number of super-peers decreases to the threshold, $1/2\,[(\delta^2 + \delta) + (l^2 + l)]$, the order of the current PDS will be reduced to the predecessor of the PDS, and the super-peer overlay topology will be reduced consequently. Then BSP computes and updates new forward and backward partners based on the new order δ in its super-peer table and sets the status of those redundant super-peers to 1. Finally, the BSP notifies active super-peers of the forward and backward partners and redundant super-peers.

Assuming 10 active super-peers in a super-peer overlay use a PDS with an order of 3. If one active super-peer sends a leaving message, the BSP reduces the topology because the number of super-peers equals the threshold 9(= (6+12)/2). The BSP appoints a min-order PDS (an order of 2) to reduce the current super-peer overlay, thus allowing up to 7 super-peers. It assigns new peer ID to the remaining super-peers. The super-peers with peer ID less than 7 are appointed as active super-peers in the reduced topology. The rest are appointed as redundant super-peers. As a result, 7 active super-peers and 2 redundant super-peers exist in the system.

C. Quantitive Evaluation

We evaluated the existing mechanisms in terms of the query success ratio, number of flooded lookup query messages and average delay. All the results are the average of 10 simulation runs. Let p_a be the probability of the resource recorded in the AVL tree and τ the connection degree. For a two-layer unstructured P2P system, the super-peer layer is in mesh-based structure, the location of a data item is arbitrary, and it uses flooding to do a best-effort search. With pure-PDG structure, whether the resources exist or not, it always sends the query messages,

$$p_a \times \tau + (1 - p_a) \times (1 + \tau + \tau^2) \qquad (1)$$

For super-peer layer with AVL-PDG structure, we want to further reduce the number of lookup messages between super-peers. The range of the flooding is

$$\begin{cases} 1 & \text{, if the resource can't be matched} \\ p_a \times \tau & \text{, if the resource is in the same local area} \\ (1 - p_a)(1 + \tau + \tau^2) & \text{, if the resource is in other overlay} \end{cases} \qquad (2)$$

IV. Simulation and Numerical Results

We present the simulation results to evaluate the performance of the random mesh-based, hierarchical unstructured P2P system and our proposed scheme. From various aspects of performance, we show that our proposed approach is practical and works well.

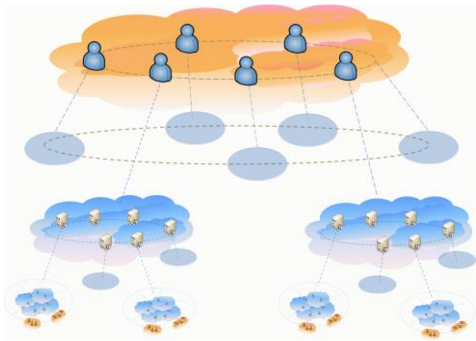

Figure 3. Multi-layer architecture for storage system.

A. Simulation Environment and Simulation Setup

We perform simulation with NS-2 (version 2.27) simulation tool [14] with GnutellaSim to evaluate our proposed method. We adopt Gnutella as our basic architecture in the simulation and add our proposed approach to the basic scheme. We validate the performance and improvement through the simulation result.

1) Simulation Environment

The topology used for simulation is presented in Fig. 2. It consists of a bootstrap peer (BSP), super-peers and ordinary peers. A new peer entering the overlay will send a request to the BSP, which then acknowledges the peer with a list containing the addresses of randomly selected super-peers. Then the joining peer starts to query the resources it needs. The general parameters are presented in Table III.

2) Simulation Setup

Each network topology is composed of 1,000 nodes, and each node is assigned as either a super-peer or an ordinary peer randomly. The ratio between the number of super-peers and the total number of peers is set to 10%. We set super-peer's download bandwidth to at least 2 Mbps and upload bandwidth at least 1.5 Mbps. The ordinary peer's download bandwidth is no more than 1.5 Mbps and connection degree is 2. We set the peers with heterogeneous link capacities such that 10% of the peers have the link capacity greater than 2Mbps, 30% of them have the link capacity less than 1.5Mbps, and 60% of them have the link capacity between these two values. Each connection's link delay is randomly set between 1 and 10 ms. We run the simulation 10 times to get its average and the duration is 1000 seconds each time.

TABLE III. EXPERIMENTAL ENVIRONMENT.

Parameter	Value
Peer Number	5,000
Simulation Time	1,000(sec)
Number of files	100,000
Query frequency	5 per sec
Super-peer Download Bandwidth	> 2 Mbps
Super-peer Upload Bandwidth	> 1.5 Mbps
Peer Max degree	2
Connection link delay	1~10(ms)

B. Numerical Results

For our simulations, we modified an implementation of Gnutella [12].

1) Aerage Traffic

Fig. 4 shows that the AVL-list with PDG overlay has the lightest broadcast overhead. In Fig. 5, the results clearly demonstrate the success rate of the AVL-list with PDG overlay significantly higher than mesh-based overlay and pure PDG overlay. The AVL-list improves about 40% and 50% of query success ratio compared to the pure-PDG and mesh-based overlay, respectively as shown in Table IV.

2) Performance of the Network Traffic

Fig. 6 shows the comparison of network traffic between mesh-based overlay and PDG-based overlay. In the former, queries for unavailable files can generate an unbounded traffic load. While the traffic load with AVL-list is bounded.

3) Performance of the Response Time

Fig. 7 shows that PDG forwarding can help resolve the queries in the super-peer overlay, which has smaller diameter than the entire P2P network. The average response time is kept between 1.5 and 2 hops with 5000 peers in the system. The simulation result shows that the average response time decreases from 4.3 hops to 1.7 hops. The overall user perceived response time can be reduced by 60%.

V. CONCLUSION AND FUTURE WORKS

We propose a novel and efficient search scheme for multi-layer P2P systems using a Multi-hop Index Replication with PDG forwarding algorithm. We show that our scheme is not only reliable but also scalable. Our work shows that unstructured P2P systems can achieve excellent scalability and reliability. The performance of the proposed scheme has been benchmarked against a super-peer overlay topology based on a mesh graph using the flooding with TTL value 7. The theoretical results showed that the Multi-hop Index Replication with PDG-based construction scheme yield a higher query success ratio, a reduced number of search messages, and a lower average hop-count delay. It would be interesting for our future work to investigate how the heterogeneity affects our proposal.

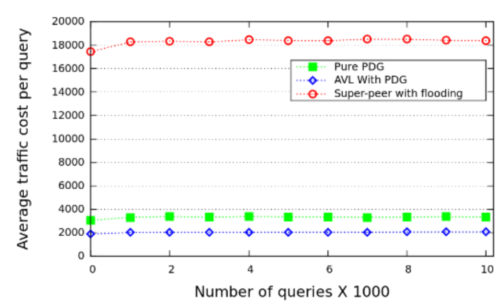

Figure 4. Comparison of number of broadcast search messages in mesh-based and PDG overlay networks

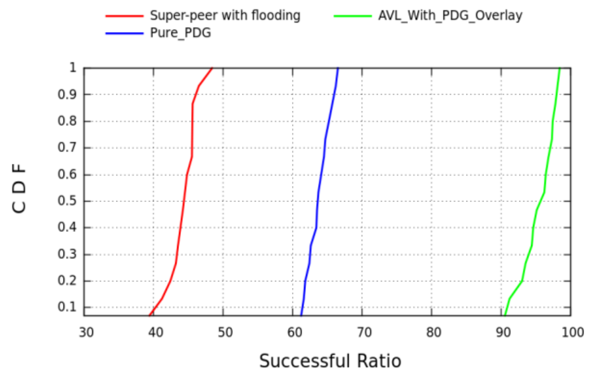

Figure 5. Comparison of query successful ratio in mesh-based and PDG overlay networks.

TABLE IV. AVERAGE VALUE COMPARISON

Scheme	Volume of Traffic	Success Ratio
Mesh-Based	100%	45.6%
Pure-PDG	18.15%	62.5%
AVL-PDG	11.09%	96.6%

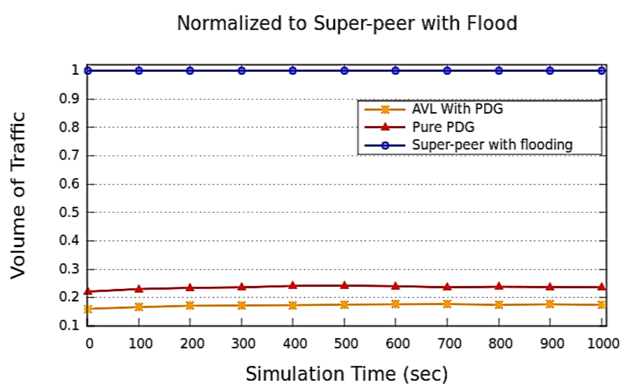

Figure 6. Normalized Network Traffic in mesh-based and PDG overlay networks.

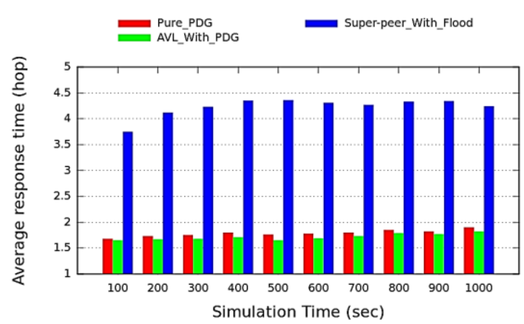

Figure 7. Average response time incurred in mesh-based and PDG overlay networks.

References

[1] S. Banerjee, C. Kommareddy, K. Kar, S. Bhattacharjee, and S. Khuller, *"Construction of an Efficient Overlay Multicast Infrastructure for Real-Time Applications,"* Proc. IEEE INFOCOM '03, pp. 1521-1531, Mar. 2003.

[2] S. Sen and J. Wang, *"Analyzing Peer-to-Peer Traffic Across Large Networks,"* Proc. Internet Measurement Workshop, Nov. 2002.

[3] Y. Chu, S. Rao, and H. Zhang, *"A Case for End System Multicast,"* Proc. ACM SIGMETRICS'00, pp. 1-12, June 2000.

[4] E. Brosh and Y. Shavitt, *"Approximation and Heuristic Algorithms for Minimum Delay Application-Layer Multicast Trees,"* Proc. IEEE INFOCOM '04, Mar. 2004.

[5] M. Freedman and R. Morris, *"Tarzan: A Peer-to-Peer Anonymizing Network Layer,"* Proc. ACM Conference on Computer and Communications Security, Nov. 2002.

[6] S. Tewari and L. Kleinrock, *"Proportional replication in peer-to-peer networks,"* Proc. IEEE INFOCOM '06, pp. 1-11, Apr. 2006.

[7] W. Acosta and S. Chandra, *"Understanding the practical limits of the Gnutella p2p system: An analysis of query terms and object name distributions,"* in MMCN, 2008.

[8] B. Parhami and M. Rakov, *"Perfect Difference Networks and Related Interconnection Structures for Parallel and Distributed Systems,"* IEEE Transactions on Parallel and Distributed Systems, VOL. 16, NO. 8, pp. 714-724, Aug. 2005.

[9] B. Parhami and M. Rakov, *"Performance, Algorithmic, and Robustness Attributes of Perfect Difference Networks,"* IEEE Trans. Parallel and Distributed Systems, vol. 16, no. 8, pp. 725-736, Aug. 2005.

[10] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, and H. Balakrishnan, *"Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications,"* IEEE/ACM Transactions on Networking (TON), VOL. 11, NO. 1, pp. 17-32, Feb. 2003.

[11] C. Gkantisdis, M. Mihail, and A. Saberi, *"Random walks in peer-to-peer networks,"* Proc. IEEE INFOCOM '04, pp. 7-11, Mar. 2004.

[12] The Gnutella v0.6 Protocol, http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html, May 2014

[13] BitTorrent, http://www.bittorrent.com/, May 2014.

[14] The Network Simulator - ns-2, http://www.isi.edu/nsnam/ns/, May 2014

[15] P. Ganesan, Q. Sun, and H. Garcia-Molina, *"YAPPERS: A Peer-to-Peer Lookup Service over Arbitrary Topology,"* Proc. IEEE INFOCOM '03, pp. 1250-1260, Apr. 2003.

[16] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, *"A Scalable Content Addressable Network,"* Proc. ACM SIGCOMM '01, pp. 161-172, Oct. 2001.

[17] B.T. Loo, R. Huebsch, I. Stoica, and J.M. Hellerstein, *"The Case for a Hybrid p2p Search Infrastructure,"* Proc. Workshop Peer-to-Peer Systems (IPTPS '04), pp. 141-150, Feb. 2004.

[18] http://Iwayan.info/Research/PeerToPeer/Papers_Research/P2P_Architecture/it02012.pdf, May 2014

[19] F. Wang, Y. Xiongand, and J. Liu, *"mtreebone: A hybrid tree/mesh overlay for application-layer live video multicast."* In The 27th IEEE International Conference on Distributed Computing Systems (ICDCS'07), Toronto, Canada, June 2007.

[20] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker, *"Making Gnutella-like P2P systems scalable,"* Proc. ACM SIGCOMM '03, pp. 407-418, Jan. 2003.

[21] S. Ioannidis, and P. Marbach, *"Absence of Evidence as Evidence of Absence: A Simple Mechanism for Scalable P2P Search,"* Proc. IEEE INFOCOM '09, pp. 576-584, Apr. 2009.

 ISBN: 978-1-61208-349-0