www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 2 Issue 11 November, 2013 Page No. 3151-3157

A new routing approach in P2P Networks

Samira kalantary

Department of Computer Engineering, Arak Branch ,Islamic Azad University, Arak, Iran
s-kalantari@iau-arak.ac.ir

Abstract: ***Peer to Peer (P2P) systems are typically decentralized, distributed and anonymous systems. Some examples of P2P systems are Napster, KaZaA, SETI@HOME, Gnutella and MojoNation. One common protocol for file-sharing P2P applications is Gnutella The Gnutella protocol requires peers to broadcast messages to their neighbors when they search files. The message passing generates a lot of traffic in the network, which degrades the quality of service. We propose the new method to optimize the speed of search and to improve the quality of service in a Gnutella based peer-to-peer environment with using semantic routing and priority of nodes. Once peers generate their "friends lists", they use these lists to route queries in the network.In our approch peers can show interest in different categories This helps to reduce the search time and to decrease the network traffic by minimizing the number of messages circulating in the system as compared to standard Gnutella.***

Keywords: Gnutella,routing method,semantic routing**.**

1. Introduction

Peer to Peer (P2P) systems are typically decentralized, distributed and anonymous systems. Current P2P systems are used to share resources like storage space, CPU power and data files in domains such as music, academic/research purposes and computation systems. Some examples of P2P systems are Napster, KaZaA, SETI@HOME, Gnutella and MojoNation. One common protocol for file-sharing P2P applications is Gnutella, which broadcasts messages to all the peers in the path of the query [4][13].

Systems using Gnutella (versions 0.4 and 0.6) have performance problems, for example, they generate huge network traffic, slow response and congestion. A study [19] showed that the traffic in Gnutella systems is mainly due to messages for establishing initial connections and for queries. Ripeanu [11] reported that the traffic generated in Gnutella consists of approximately 92% Query messages, 8% Ping messages and hence the other messages constitute less than 1% of the traffic. Thus reducing the number of query messages would help in reducing traffic. Studies have also found that Gnutella systems suffer from bandwidth, congestion and latency problems [2][12]. Gnutella does not exploit the fact that people with similar interests are likely to store files that would be useful for other people sharing those interests. People with similar interests form communities that allow them to exchange resources more efficiently. We propose a query routing approach to alleviate the performance problems caused by the flooding algorithm in Gnutella systems. This approach introduces the concept of learning from experience and keeping a friends list (a list of peers that have shown to be useful) in different semantic areas (we call these "categories") so that queries can be routed semantically to these peers. Using friends lists potentially helps in reducing search time and decreasing traffic by minimizing the number of messages circulating in the system as compared to standard Gnutella. We want to show that by learning the another peers's interests, building and exploiting their friends lists, peers can get more relevant resources faster and with less generated traffic, i.e. the performance of the Gnutella system can be improved. While we believe that this approach can be applied successfully in large network, the experiments described in the paper are for small P2P communities. The main reason for this is the complexity of the simulation. An example of such system is Comtella [17][18] where the users share academic papers or learning materials in the context of a department or a class. The categories used in Comtella can be based on a subject index of the discipline. For our approach to work, it is necessary, that the peers share at least partially these categories. The paper is organized as follows. Section 2 gives an overview of the areas of peer-to-peer networks.The conceptual design of our model is explained in section 3 and the design of the experimental model is described in chapter 4. Section 5 shows the results of the experiment. Section 6 concludes the paper .

2. Previous Work

P2P systems are usually defined as distributed systems where peers or entities share computer resources and services by direct interaction among themselves [8]. The resources that the peers share in the P2P systems can be files, CPU power, and disk space. Efficient searching of files is an important problem in P2P file sharing systems.

2.1 Improving Search in P2P Systems

Lv et al. [5], introduce two approaches to improve search in Gnutella. The first approach is an "expanding ring" where the TTL (time to live) value is increased gradually to find the resource. There is duplication of messages to the same peers and peers do not learn from past experience to bypass previously forwarded peers. This approach still floods the network with messages.

The second approach, "random walk", sends a query to only one peer at a time at each hop and sequentially searches

through the system to get the results. Few messages are generated but the search time increases, as the search is sequential rather than parallel.

Another approach, called "Directed Breadth First Search" (DBFS) [24] selectively forwards queries to peers that have returned "good results" for previous queries. This approach still generates a lot of messages. If the number of results is taken to be an indication of "good results" instead of the quality of the results, then a peer that returns irrelevant but many results would be considered a good candidate.

In the "Buddy Web" [22] approach routing is done based on similarity of interest. The drawback of this system is the calculation of similarity. Keywords may not be indicative of the content of the document and keywords found by highlighting important words in the document may not necessarily reflect the interest of the peer as words have different meanings depending on the context. BestPeer [22], mentioned above, is a P2P network prototype, implemented in a university setting. The peers that return the maximum number of results are kept in a list. The approach takes into consideration implicitly the similarity of interests among peers, but it is not defined explicitly what the interests are and how they are modeled.

Sripanidkulchai [14] proposed a solution where when replies are returned, the querying peer chooses randomly a peer among all the peers that replied and adds it to a "shortcut list". The drawback here is that the shortcut list contains peers who have returned responses regardless of the relevance the reply and the semantic of queries.

A peer in [3] maintains a profile of all other peers who answered its requests in a local repository by keeping a pair list of (query, peer). If peers change their search interests often then older queries or peers will be lost. In another approach [15], documents are accompanied by keywords, which semantically represent the contents of the documents, and these keywords are classified according to semantic categories. Clusters of peers are formed based on semantic categories. The drawback is that if each peer was to keep a lot of information and as the number of categories increases and the number of peers in a cluster increases or decreases, each peer has a huge overhead for storing and updating the information.

Another approach based on the idea of finding "good peers" in a P2P system is proposed in [10]. Peers that have sent a "good" response to a peer's request are entered in a special list by the peer, following the assumption that these peers may also have good resources for subsequent queries in this area. This works if the user is consistently searching for a single semantic category several times in one session. However, users search typically for more than one semantic category at a time. Therefore, this approach will keep generating a lot of traffic in the network as the user switches her semantic category.

3. Proposed Routing Approach

In this study we propose a model in which peers keep a list of other peers ("friends") who they see as being similar to them according to some criterion (e.g. semantic area of interest). Each peer can have many different criteria and a list of peers associated with each criterion. The system has a number of peers, and each peer shares some files, representative of the peer's interests. Peers share these files with all other peers in the network. The files are classified into categories according to the same criteria as above, i.e. reflecting some semantic areas of interest. Peers can show interest in different categories. A category is defined as an area characterized by a set of topics or keywords [16]. For example, topics like distributed databases and peer-to-peer systems characterize the area of distributed systems, which is a category in our model.

Peers can learn about another peer's interests since they keep track of all the peers who responded to a query in a given category. The response obtained from a given peer is taken as an indication of that peer's interest because the peer is keeping and sharing documents pertaining to the category being queried. A peer can also learn about the interests of other peers by analyzing who initiated a query in a category. Thus each peer can classify according to their interests the peers from whom it received responses and the queries can be forwarded to peers in the same interest groups. A peer maintains separate lists of "friends" for all of its interest categories and adds peers to the lists based on evidence about their interests. When a peer queries and gets responses, it keeps track of all those peers who returned the responses; they are assumed to be interested in that category. For example, if peers, A, B and C responded to peer G's query in category X, then G adds A, B and C to its "friends" list related to category X. Reversely, all the responding peers A, B, C will know that G is interested in category X, since it generated a query in this category, so they will add G to their lists of friends in category X. If any of the 'friend-peers" does not generate the response, it will use its own "friends" list for the category of the query to propagate it further and finding an answer to the query will have a higher likelihood.

Thus the benefit of this routing approach is that queries will travel less and success is more likely to be achieved with a smaller number of hops. If the query is sent to random peers who have no interest in that category the query will most likely not succeed, since most likely they don't have the file. If they don't have the file, they will have to forward the query further to other random peers. Of course, it can happen that one of the peers on the way has the file. The chance is higher the higher the number of queries spreading. Therefore, flooding is essential in Gnutella. However, most of the huge number of generated queries gets aborted due to exhausting their time to live and replies are delayed as a result of the traffic generated by the flooding algorithm. If a peer has the file, it will generate a response and it won't forward the query further, so there will be no more traffic generated further from this peer in the network. Since the likelihood of a peer on a semantic routing chain to have the queried file is higher, responses will come with a higher likelihood, faster and will generate less traffic.

Another advantage of this routing approach is that it allows for subjectivity. Maintaining a central directory of peers and their interests is of course, possible and would add significant efficiency. However, this would be a Napster like centralized architecture with a central point of failure. In addition, it would require an agreed upon list of categories.

In our approach each peer builds its network according to its own criteria - both for how it defines the category of interest and how it decide whether to add or remove peers from its "friends list". The lists of friends reflect similarity of tastes in a category between the peers. For example, a number of peers may be interested in a particular general category, but a peer may be interested in a few specific topics in that category; therefore, its friends lists will contain peers with interest in these topics only, while another peer may be interested in another set of topics and will have a different list of friends. These are subjective criteria and are hard to measure and sum

up objectively. Imposing an "objective" measure of similarity can be meaningless. In a centralized approach each peer sends the number of documents it shares in each category to the central server when the peer joins the system. The server keeps a list of all the peers it communicated with along with the number of documents it shares. In this centralized approach, the usefulness of a peer would have to be an objective measure, for example, the number of documents shared by a peer in a category. Peers can also report their satisfaction with regards to any peer to the central server, which computes a "reputation" value for each peer, which can be looked up by other peers. The disadvantage of the centralized approach is that such "average" objective measure cannot capture the specific needs of all peers. A peer that is highly specialized won't score very high on an objective measure since only a few peers would use it. However, it may be invaluable for a specific peer who shares exactly these special interests. Also, in a centralized approach, the list containing peers is complete. In our model, where in each peer maintains its own list, the list is not complete as only those peers who responded in a category are kept.

3.1 Strength of Relationships

A peer attaches a strength value to each relationship with a peer from its friends-list for the category. The strength of the relationship with a peer from a friends list related to one category is independent of the strength of the relationship with the same peer if it happens to be in a list for a different category. That means that if x is the strength that peer P assigns to its relationship with peer E for category X and y is the strength of relationship for peer E in another category Y, y is independent of x. For example peer P may have downloaded documents from E in two different categories X and Y: many times in X and only a couple of times in Y. Therefore x will be higher than y. The strength of relationship is updated after interactions with the other peer. The evidence taken into consideration when updating the strength of relationship includes the success rate the peer had with queries sent to the other peer, the reliability of the other peer while downloading documents, (e.g., does it stay active or disconnects when document is being downloaded), the quality and the usefulness of the resource.

The "friends list" is updated by the following mechanism. After the interaction or file transfer takes place, based on the corresponding interaction value (success or failure), the peer calculates the strength of the relationship for that peer. The formula to calculate the strength of the relationship is given in below

$$S^{C}_{AX}=R^{C}_{AX}/Q^{C}_{A} \quad (1)$$

where, S^{C}_{AX} is the relationship strength between peers A and X, R^{C}_{AX} is number of interactions (i.e., peer is satisfied with response) by peer X to queries in category C issued by peer A and Q^{C}_{A} is total number of queries issued in that category by peer A. The strength of the relationship is maintained between 0 to +1, where 1 denotes a strong relationship.

3.2 Semantic Routing

When a query is initiated or is received by a peer, the query is classified into a category and the request is sent to

the peers from the "friends" list associated with that particular category. This is beneficial since the query is now circulating among peers who have shown interest in that category and there is a greater probability of the document being found faster.

A peer can send a query to all peers in its "friends-list" for the category of the query. Depending on the length of the friends list a lot of traffic can be generated since it is in fact broadcasting on a smaller scale. Therefore, it is a better solution to forward queries only to a small number of "best friends", based on the "strength of relationship". If the value of the relationship strength is high it indicates that the peer has been particularly helpful, reliable and shows a greater success rate in getting responses. Therefore, peers are ordered in the "friends lists" based on the relationship strength and new requests are sent only to a few peers from the top.

3.3 Discovering New Friends

A P2P system is highly dynamic: peers come and go and change their interests. Consequently there is always a need by peers to discover new friends in the system. In our model, new friends are discovered by sending queries to a few peers that do not belong to the "friends lists" of the interest group of the querying category. That means that the neighborhood of a peer, i.e., the peer to which a new query is sent is formed by m unknown peers and n-m friends, where n can have different values. The standard Gnutella neighborhood has size n = 7. An unknown peer once chosen will send the query to its friends (if it has any) in the querying category. The peers chosen by this independent peer for forwarding the query further may be new or not related to the querying peer and still find responses. Thus, a peer chosen outside the friends list group lets the query go to new or unrelated peers who may have either entered the system recently or acquired document(s) of the interested category recently. This enables discovering new friends with the tradeoff that more messages will be generated. The worst case will be that all the peers are chosen from outside the interest group of the querying category since the peer doesn't have friends yet and they have completely different interests and have no "friends" lists in the category of the query. This will result in a standard Gnutella routing based on the neighbors that are currently on line. The result is no new friends are discovered but messages will be generated until the query expires. The above scenario indicates that it is necessary to strike a balance between choosing friends versus unknown peers for forwarding a query, i.e., choosing a good value for m. This is necessary to derive benefits from past experience and still be able to explore

4. Experimental Design

The main objective is to show that by learning the other peers's interests, building friends lists and exploiting their friends lists to route queries semantically, peers can get more relevant resources faster and with less traffic generated.

Our experimental model has some modifications as compared to the standard Gnutella in order to simplify the protocol for the simulation. The first modification is that peers who have the requested files send directly responses back to the peer who originated the query and not through the queried path. This reduces the overall time for a reply to come back to the peer originating the query as compared to Gnutella, and it leads to non-anonymity in the system, as the owner of the file and the downloading peer become known. The second modification is that the file is sent back by the responding peer to the peer who originated the query in response to the query, i.e. there is no

"query hit" response sent back, followed by a transaction for downloading the file initiated by the peer who originated the query; the whole interaction happens at once These two modifications are only for the purpose of the simulation and they do not impact the validity of the results if they would be applied to a real Gnutella system.

We use the described simplified protocol as a basis for the simulation and compare results of two versions of the system: a version with "friends list" with the version of without "friends list". A further simplification is necessary in order to be able to obtain computationally feasible simulation. In the baseline model, we can assume without loss of generality that there is only one category of interest in the system, and that there are some peers that are interested in the category and the rest of peers in the system are not interested in that category. This assumption can be justified as follows. Recall that every peer keeps its "friends lists" for different categories separate. The relationship strength with a given other peer in one list is not influenced by the strength of the relationships with the same peer in any other lists it features in. Since the peers in the friends lists are used to send queries only for the particular category for which the peer has generated the query and there is no interaction between queries, the speed of learning of new friend peers in a given category will not influence the speed of learning of friends in a different category. In fact, the system can be virtually broken down into several independent subsystems for the different categories in the system, and each subsystem can be simulated independently. It is enough to simulate the system with only one category. For the purpose of the simulation R^{C}_{AX} , in the formula presented in (1), is the number of responses by peer X to queries in category C issued by peer A instead of the number of satisfactory interactions. This simplification was necessary due to the complexity of the simulation resulting of the introduction of another random variable to represent the peer's satisfactions.

In the beginning there are no relationships among the peers in the system, since there have been no interactions between them yet. Interactions happen as queries are generated and responses arrive. The peer originating the query keeps track of which peers respond to each of its queries. It then calculates the strength of the relationship with that peer according to the formula in (1).

Generally users in the Gnutella system decide if they want to keep or delete a file that they have downloaded. To keep the simulation simple and to avoid such decision making we assume in the simulation that the file queried is not downloaded and shared by the querying peer. It can query again for the same file. Thus the file distribution in the simulation remains constant throughout the simulation run.

The simulation has been implemented in Java on JADE, a multi-agent platform [1]. Hundred (100) peers were created.

Two hundred (200) unique files from the category of interest are created in the system. Peers generate queries that are random natural numbers, F = 1,2,3… representing files. Queries are generated by the system by randomly choosing peers from the list of all peers in a fixed interval of time. The file number to be queried is randomly generated. The simulation system1 contains a varying number of peers with and without files and a single category of interest. The set-up of the system is such that the file distribution varies among the peers sharing files, i.e., there are approximately 57% peers that share ten files, 29% peers sharing twenty files and 14% peers sharing forty files. This file distribution remains consistent, though the actual number of peers sharing files in the system changes. The relationship strengths for all peers are calculated after a set time and sorted so that the list is updated. Thus a peer can use the most recent update of relationship strength to pick peers to forward queries.

We experimented with different proportions of peers sharing files in the category: 8%, 10%, 15%, 30%, 50%, 70%, 90% and 100% of the total number of peers simulated in the system. In each of these cases, the remaining peers are considered to be not interested in the category and thus they do not share files of that category. They just facilitate the circulating of messages in the system. Since the percentages of peers in the category in each simulation differ, to obtain comparable results, each peer on average is assumed to query files twenty (20) times. Thus the time taken for the simulation varies according to the percentage of peers in the category as the total number of queries differs in each simulation. The numbers of neighbor-peers to which a peer sends its requests is set to five (5). There are two sets of experiments differing in the number of neighbors chosen from the friends list. In the first experiment a peer has a neighborhood of four peers from its friends list (the top four with strongest relationship) and one peer is chosen randomly from the rest of the peers in the system. In the other experiment three peers are chosen from the friends list, and two peers are chosen randomly from the rest of the peers in the system. The TTL (time to live) of the requests is 4.

For the purpose of comparison, another system with 100 peers, all interested in the same category, was implemented. This is a good case for the standard Gnutella system as all peers in the system are interested in the same category and all peers share at least a few files from that category. Thus the connected neighbors may have the file for which the peer is querying, thus obtaining faster response and fewer message are generated. If the distribution were chosen such that about 10% peers are interested in the category then 90% peers would just be forwarding requests and the few peers that share files could be dispersed in the system. This will lead to increased number of messages and hops. The file distribution among peers remains consistent with that of the above experiment. Here each peer at set-up picks five other different peers randomly and makes them its neighbors. The neighborhood is fixed throughout the length of the simulation, as in the standard Gnutella. During the simulation we collect data so that we can find the average time taken for each query to receive a hit (measured in number of hops), and the number of messages circulating in the system. The average number of hops for each query is calculated first and then the average number of hops for all queries. Similarly, the number of messages for each query is noted and summed up and then the average number of messages for all queries is calculated. Each set of simulation runs were repeated three times and the average of these runs were used to obtain graphs. Our hypothesis is that the "friends list" reduces the time for searches and reduces the number of messages circulating in the system.

5. Other recommendations

The graphs in Figure1 and Figure2 are drawn from the data gathered from this experiment. In Figure1, the straight line indicates the average number of hops when no friends list is maintained, the lighter line with squares indicates the average number of hops when four peers from the friends list are in the

neighbourhood and darker line with diamonds indicates the average number of hops when three peers from the friends list are in the neighbourhood. From the graph above we see that when four neighbours are chosen from the "friends" list, the average number of hops increases from 1.84 (when 8% of the peers in the system are interested in the category) to 2.68 (when 90% of peers are interested in the category).

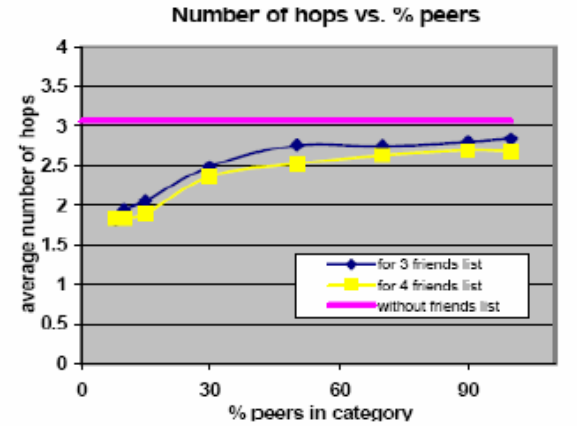

Figure1 Average hops vs. percentage of peers in category

The average number of hops seems to level off at 2.68 hops. When three neighbors are chosen from the "friends" list, there is a slight increase in the number of messages and the average number of hops increases from 1.81 (when 8% of the peers are in the category) to 2.80 (when 90% of the peers are in the category). The number of hops seems to be leveled off at 2.84 hops.

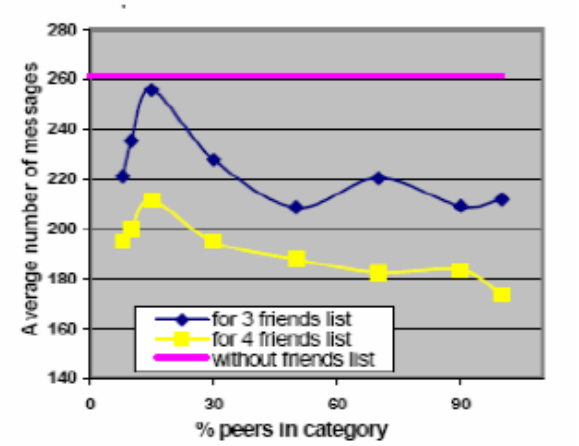

Figure 2 Average messages vs. percentage of peers in category

Line with squares indicates the average number of messages when four neighbors are from the friends list and the darker line with diamonds indicates the average number of messages when three peers neighbors are from the friends list When four neighbors are from the friends list the average number of messages decreases from 194.85 (when 8% of the peers are interested in the category) to 173.54 (when 100% of the peers are interested in the category), with a slight increase in between to 211.54 (when 15% of the peers are interested). Similarly when three neighbors are from the friends list the average number of messages decreases from 221.06 at 8% in category to 211.82 for 100% in category, with a slight increase in between to 255.73 for 15% in category**.**

6. Conclusions And Future Work

The objective of the project is to investigate the use of semantic routing to optimize search and quality of service in the Peer-to-Peer environment. We simulate a Peer-to- Peer type of environment with JADE multi-agent system platform. In our model each peer builds a "friends list", for each category of interest and uses it for searching files in the network. From the results obtained we see that creation of "friends list" helps in reducing search time for queries and reduces the number of messages circulating in the system.

Future work will include an investigation of how the system behaves when peers are programmed to learn from other peers' queries. At present, each peer discovers on its own information about other peers by sending queries and building "friends list" for that category. However, a peer can also learn by observing the traffic in the system, i.e., by keeping track of queries passing through it and the peers that initiated these queries, and adding those peers to its "friends list" in that category. Current and future work is focused on simulating a dynamic system where peers can join and leave the system and allowing peers to change their interests, i.e., switching from the interest group to the non-interest group. The distribution of files in the system would be changed .

7. References

[1] JADE2.61: ttp://sharon.cselt.it/projects/jade/

[2] Jovanovic, M. (2001), "Modeling large-scale peer-to-peer networks and a case study of gnutella," M.S. thesis, University of Cincinnati.

[3] Kalogeraki, V., Gunopulos, D., and Zeinalipour-Yazti, D. (2002), "A local search mechanism for peer-to-peer networks", Proceedings of the Eleventh International Conference on Information and Knowledge Management, McLean, USA.

[4] Krishnamurthy, B., Wang, J., and Xie, Y., (2001), "Early Measurements of a Cluster-based Architecture for P2P Systems", Proceedings of the First ACM SIGCOMM Workshop on Internet Measurement Workshop, San Francisco, CA, 105-109.

[5] Lv, Q., Cao, P., Cohen, E., Li, K., and Shenker, S. (2002), "Search and replication in unstructured peer-to-peer networks", Proceedings of the 16th international conference on Supercomputing, New York, USA, 84 – 95.

[6] McCarty, C. (2002) " Measuring Structure in Personal Networks". Journal of Social Structure, Vol. 3, No.1.

[7] Milgram, S (1967), "The Small World Problem", Psychology Today, 60-67.

[8] Milojicic, D. S., Kalogeraki, V., Lukose, R., Nagaraja, K., Pruyne J., and Richard, B. (2002), "Peer-to-Peer Computing". Internal Report Hewlett Packard, March 8.

[9] Newman, M. E. J. (2000), "Models of the Small World: A Review", Journal of Statistical Physics, 101, 819-841.

[10] Ramanathan, M. K., Kalogeraki, V., and Pruyne, J. (2001), " Finding Good Peers in Peer to Peer Networks", Hewlett Packard Technical Reports.

[11] Ripeanu, M. (2001), "Peer-to-Peer Architecture Case Study: Gnutella Network", Proceedings of IEEE 1st International Conference on Peer-to-peer Computing, Linkoping Sweden.

[12] Saroiu, S., Gummadi, P., and Gribble, S. (2002), "A measurement study of peer-to-peer file sharing systems", In Proceedings of Multimedia Computing and Networking (MMCN), San Jose, California.
[13] Sripanidkulchai, K. (2001), "The popularity of Gnutella queries and its implications on scalability", Featured on O'Reilly's www.openp2p.com website, February 2001.
[14] Sripanidkulchai, K., Maggs, B., and Zhang, H. (2003), "Efficient Content Location Using Interest-Based Locality in Peer-to-Peer Systems", Proceedings IEEE INFOCOM 2003, San Franciso, USA.
[15] Triantafillou, P., Xiruhaki, C., and Koubarakis, M. (2002), "Efficient Massive Sharing of Content among Peers", IEEE
Workshop on Resource Sharing in Massively Distributed Systems, Vienna, Austria.
[16] Vassileva, J. (2002), "Motivating Participation in Peer to Peer Communities", Proceedings of the Workshop on Emergent Societies in the Agent World, ESAW'02, Madrid, Spain. Springer Verlag LNAI 2577, 141-155.
[17] Vassileva J. (2002) Supporting Peer-to-Peer User Communities, in R. Meersman, Z. Tari et al. (Eds.) "On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and ODBASE" Coordinated International Conferences Proceedings, Irvine, 29 Oct - 1 Nov. 2002, Springer Verlag LNCS 2519, Springer Verlag: Berlin- Heidelberg, 230-247.
[18] Vassileva, J., (2004), "Harnessing P2P Power in the Classroom", To appear in Proceedings of Intelligent Tutoring Systems (ITS'2004), Maceio, Brazil.
[19] Vaucher, J., Babin, G., Kropf, P., and Jouve, Th. (2002), "Experimenting with Gnutella Communities", Distributed Communities on the Web (DCW 2002), Sydney, Australia. LNCS 2468, Springer Berlin, pp.85-99.
[20] Venkataraman, M., Yu, B., and Singh, M. P., (2000), "Trust and Reputation Management in a Small World Network", Proceedings of Fourth International Conference on MultiAgent Systems, 449-450.
[21] Waloszek, G. (2002), "Personal Networks", SAP Design Guild, Ed. 5: Collaboration, http://www.sapdesignguild.org/editions/edition5/personal_ networks.asp
[22] Wang, X., Ng, W., Ooi, B., Tan, K., and Zhou, A., (2002), "BuddyWeb: A P2P-based Collaborative Web Caching System", a position paper in Peer to Peer Computing Workshop (Networking), Pisa, Italy.
[23] http://xena1.ddns.comp.nus.edu.sg/p2p/BuddyWeb.ps.
[24] Wellman, B. "An Electronic Group is Virtually a Social Network", In Sara Kiesler ed. (1997), Culture of the Internet, Lawrence Erlbaum, Hillsdale, NJ, 179-205.
[25] Yang B., and Garcia-Molina H. (2002), "Efficient Search in Peer-to-Peer Networks", The 22nd International Conference on Distributed Computing Systems, Vienna, Austria.