International Journal of Parallel and Distributed Systems and Networks, Vol. 5, No. 4, 2002

EFFICIENT DISCOVERY OF IMPLICITLY FORMED PEER-TO-PEER COMMUNITIES[#]

M.S. Khambatti,[*] K.D. Ryu,[*] P. Dasgupta[*]

Abstract

Current peer-to-peer systems are targeted for information sharing, file storage, searching and indexing often using an overlay network. In this paper we expand the scope of peer-to-peer systems to include the concept of "communities". Communities are like interest groups, modeled after human communities and can overlap. They can also exist without anyone knowing about their existence. Communities are created, implicitly when one or more entities claim an interest in the same topic. Our work focuses on efficient methods to discover the formation of these self-configuring communities. We investigate the behavior of randomly created communities and model the complexity of discovery algorithms. Discovering communities on the fly is essential to being able to perform community directed searching. In addition, efficient discovery algorithms allow us to manage quickly changing community structures (dynamic communities, failures, mobile nodes and so on). We use some simulations to discover the architecture of randomly created communities and then perform studies on techniques for discovering communities.

Key Words

Distributed computing, peer-to-peer computing, self-configuring networks, peer communities.

1. Introduction

Peer to Peer (P2P) systems are distributed systems in which logically distinct computing elements called *peers,* having comparable roles and responsibilities, communicate information, share or consume services and resources amongst each other. These systems have the potential to harness massive amounts of storage with modest investment and no central authority [1, 2]. The emergence of file sharing applications such as Gnutella [1], Freenet [2], and Napster [3] has been the catalyst that drew a lot of attention to P2P systems.

The building block of the current P2P systems is the notion of a peer-group, or a number of nodes that participate with each other for a common purpose. In this paper, we discuss a generalization of the notion of peer group to a multiplicity of groups (possibly overlapping) called *peer communities.* While a group is a physical collection of objects, a community is a set of active members, who are involved in sharing, communicating and promoting a common interest.

Our concept of peer communities is loosely based on the idea of "interest groups", for example Yahoo Groups [4] or Usenet Newsgroups. A node in the system claims to have some interests and depending upon the claims of all the peer nodes, the communities are implicitly formed (made up of peers with the same or similar interests). Note that the groups are formed implicitly (i.e. they are self organizing). If a node in New York declares an interest in wombats, and a node in China also declares this interest, then the two of them become part of an implicit, undiscovered community. As is obvious, a node may belong to many different communities and communities may overlap.

In this article, we provide a motivation for the study of P2P communities and illustrate some scenarios to define and discover the community structure. Using simulated models of communities, we show how communities can be formed and discovered. The simulation results have provided us with an insight on behavior of random networks.

2. Motivation

Psychologists have long shown that people have an affiliation motive [5] and a need for information about the world around us [6]. These are some of the instincts that have driven the formation of human groups. This tradition of group forming is not alien to computer science and we witness examples of it in Usenet groups, web communities [7], yahoo groups [4], chat rooms and so on.

With the exception of web communities that have been shown to be self-organized [7] and the alt.* Usenet groups; almost all other present-day groups are a result of *a priori* planning and implementation or at the very least they require some central control or a central authority through which advertisements can be made.

In contrast to this, P2P systems are defined to be completely de-centralized and can also be dynamic. This makes them very attractive as system solutions to the "little people", like home users, small-scale networks, ubiquitous

[#] This research is partly funded by grants from AFOSR, DARPA, Microsoft and NSF.

[*] Arizona State University, Tempe, AZ 85287-5406, USA; e-mail: {mujtaba, kdryu, partha}@asu.edu

computing environments, dynamic coalitions [8] and so on, who now will have the ability to choose their own policies, roles, and responsibilities and change them autonomously.

2.1 Peer Communities

The popularity of schemes to form communities and associations on the web leads to the use of the P2P system structure for realization of underlying community structures. Thus P2P communities are not only a natural extension for arranging distributed systems, but also to enhance the capabilities of each member.

The current crop of P2P systems focuses on global information sharing, searching and replicated file storage. Much of the research is targeted at creating an overlay network (based on node identifiers) and using the overlay network to perform searches on a content addressable space. In our work we extend the above concepts to allow the notion of communities. Communities are useful in structuring the information storage space, discovering resources and pruning the search space. It also aids in better dissemination of useful information. For example, suppose node X belongs to a person interested in Amazonian Biological Catapults (ABC). After X declares this interest, it becomes a member of the community of ABC enthusiasts. Henceforth, all information X wants to share can be placed in a public directory and will be readable/searchable by all members of ABC. This concept can be extended to discover resources, physical devices, network components. It also has some interesting security and access control issues (including trust management).

The above example is interesting in the context of peer communities with no overlapping interests. To enable cross community information exchange we widen the scope of our design, with another example. Consider a digital library. Each node in the library system owns a set of books that it is willing to share with other nodes[1]. The subjects of the books owned form the different communities in this peer architecture. We will discuss this type of system further in the next few subsections.

2.2 P2P Searching

Searching for information is one of the mainstays of P2P systems. Centralized searching (as used by Internet search engines such as Google) has the downside that the central authority controls the indexing and presentation of the information. P2P searching allows anyone to put up information in the search index and then cooperatively search the P2P space. P2P searching techniques include flooding, directed flooding, iterative deepening, directed BFS and so on [9].

Considering the digital library again; if the Computer Science and Medicine communities are disjoint, then searching for medical information by a node belonging to the Computer Science community would not produce any results. If the communities (see fig.1) were linked at some point Q (Q belongs to both communities) then medical information would be found, but at a great search expense, as on the average half the Computer Science community would be searched before the Medical community.

[1] Assume these are non-copyrighted works.

To mitigate such problems, we need community-based query propagation. Thus to provide efficient searching, it is better to target a search for one (or more) target communities, irrespective of the current membership of the searching node.

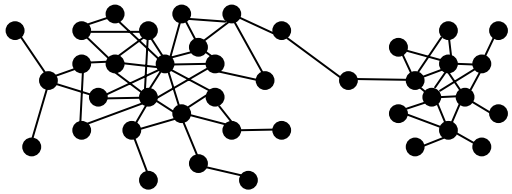


Figure 1. Example of peer communities linked by a common peer

To enable regular searching as well as community-based searching, we must be able to efficiently discover communities, that is, we need searches for community members. This is somewhat complicated by the fact that communities are implicit, self-organizing, dynamic and constantly changing—forming, or breaking down due to changes in the peers.

Our work is targeted to discovery of communities on the fly to enable efficient intra-community as well as inter-community searching. To this end we have studied the community formation characteristics using simulations of large communities and found characteristics that can be exploited to perform discovery and searching. We show that this discovery does not require extensive computation or communication on the part of a peer.

Note that searching is not the focus of this paper, searching is the next step after community discovery algorithms are implemented.

3. Related Work

Contemporary search algorithms are executed on a server or a group of servers that have access to a data repository on behalf of a requesting client machine. The absence of servers within a P2P environment has brought about a number of proposals for efficient search techniques. Some are highly deterministic but can also be very expensive, like flooding, which requires a search query to be sent to every peer existing. An optimization to flooding, called directed flooding, only sends queries to all peers in a certain part of the P2P system based on some knowledge or history. Additionally, [9] has proposed techniques that reduce communication and increase the probability of arriving at a solution to the search query. Examples of these are iterative deepening, which iteratively increases the depth until which to flood; directed BFS, which selects a subset of immediate neighbors heuristically and entrusts them with further propagation of the search query using directed BFS again; and local indices, which requires each peer to maintain a local index of the contents of peers that exist within a pre-determined range.

A considerable amount of research has focused on the analysis of the link structure in collections of objects. Through these analyses, researchers had hoped to derive the procedures for efficient identification and discovery of patterns in the collection. Early attempts to analyze the collective properties of interacting agents have been found in social networks [10], where link structures like cliques, centroids and diameters were studied. The field of citation analysis [11] and

bibliometrics [12] seek to identify patterns in collections of literature documents by using citation links. We give below one such notable definition of web communities that uses an analysis of the link structure of web pages to efficiently identify communities.

"We define a community to be a set of web pages that link (in either direction) to more web pages in the community than to pages outside of the community." [13]

At first glance, the above definition seems just what is needed to identify P2P communities. However a closer look will indicate that if peers are placed in P2P communities based entirely on link analysis, we will not accomplish our goal of allowing peers to simultaneously be members of more than one P2P community.

Perhaps a combination of graph theory and link analysis is needed to correctly identify patterns within collections of peers. For instance, if the links of a peer were classified, as outgoing and incoming links, and if there existed cycles in the directed graph formed by the peers and their links, we would have successfully identified P2P communities. These cycles could be discovered in a depth-first manner by exhaustively traversing the outgoing links to check if they visit a peer twice.

Except for the obvious scalability issue in the above technique, it seemed to guarantee to uncover any pattern that might exist. Yet the procedure would fail for some of the most commonly occurring patterns: the star, where many peers are huddled around a single peer like in a client-server configuration, or the tree, where no cycles would exist. Even in a domain that is purely P2P, one cannot avoid the fact that often many peers will use a small subset of peers or even a single peer as a source of information or for collaborations.

On the topic of discovery, a particularly remarkable solution was proposed by [14]. It offered an approach called HITS that was related to spectral graph partitioning and methods used by the Google search engine [15]. Their recursive definition of *hub* and *authority* web pages work to rank results by a measure of importance and usefulness, thereby identifying key web sites related to some community and also the related websites that might be members of the same community [16]. However their approach is highly dependent on the link topology and therefore cannot effectively aid in the discovery of communities that are ring-based without any dominating members.

Complementary to the HITS algorithm, [13] requires a "seed" web site as the starting point to begin a focused crawl in order to identify a community. Members of the community are discovered using the maximum flow / minimum cut framework using the two sets called source and sink, initially composed of well-known web sites. When mapped to the P2P domain, this link-based technique has the drawback of not truthfully modeling real world P2P communities where peers can simultaneously be members of more than one community.

In trying to empower peers to discover their community membership, we had to find an efficient solution that would be practical, unlike the NP-complete solutions offered by graph partitioning [17].

4. P2P Network Modelling

Formation and discovery of peer communities are significantly dependent on how peers declare and use their common "interests". First we define *attributes* as a method of declaring interests, and then we use these attributes to discover communities (in a simulated model).

4.1 Attribute-Based Communities

Peer communities are formed based on common interests. In our model, common interests are represented by *attributes*, which are used to determine the peer communities in which a particular peer would participate. Attributes can be either explicitly provided by a peer or implicitly discovered from past queries. For example, a housewife can express that she is interested in French wines and house decoration. Such expressions are personal declarations. Also, her repeated web search queries to find "K-12 education in Arizona" can be used to provide implicit information about her interests. There are of course privacy and security concerns in using such information, so we divide interests into three classes – *personal*, *claimed*, and *group*.

The full set of attributes for a peer is called *personal attributes*. However, for privacy and/or security reasons, all these attributes may not be used to determine community membership. A client may not want to reveal some of her personal attributes. Hence, a subset of these attributes is explicitly claimed public by a peer. We call these the *claimed attributes*. The claimed attributes are a subset of the personal attributes.

In addition to personal and claimed attributes, we introduce the notion of *group* attributes. Group attributes are location or affiliation-oriented and are needed to form a physical basis for communities. Every node belongs to one pre-determined group and has a group attribute that identifies the node as a member of this group. For example a computer

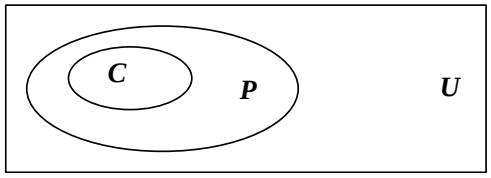


Figure 2. Venn Diagram of attribute sets for a peer.

on the campus of Arizona State University belongs to the ASU group; a home computer using the AOL Internet service provider is part of the AOL group; computer in the office of an IBM employee belongs to the IBM group. The domain name of an Internet connection may be used as the group identifier.

In fig.2 above, *U* is the universe of all attributes, *P* is the set of personal attributes, and *C* is the set of claimed attributes. The group attribute is also considered a personal attribute and it may or may not be one of the claimed attributes. Although it is expected that a node will put the group attribute as part of the claimed attribute set, it does not have to do that.

Now, we formally define a P2P community based on the attributes of each peer.

P2P COMMUNITY: *The non-empty set N of nodes is a peer-to-peer community iff N has a non-null signature.*

SIGNATURE: *Let n be a node and claim(n) be a set containing attributes claimed by n. Consider a non-empty set N of nodes. Then the set resulting from the intersection of claim(k), for all k Є N is called a signature of the set N.*

With this definition, given any collection of peers, we would be able to tell whether the collection is a P2P community or not.

4.2 P2P Network Links

P2P communities are attribute-based, that is, attributes (claimed and group) determine the membership of a node in one or more communities. In addition to attributes we also define an overlay network in terms of "links".

Links are not necessarily needed to form and manage peer-to-peer communities. However, they are needed to feasibly run low-cost algorithms for formation and discovery, as it is conceptually and algorithmically simpler to use the notion of a set of "neighbors" when communicating with other peers. First, let us explain a case where links are essential.

Suppose a node, belonging to domain abc.com claims the attribute "*baseball*". This node is essentially isolated, unless it a priori knows about the other members of the baseball community or the other members of the abc.com community. There is a need for a "seed" to start the community formation and information search needs.

Flooding and querying a central server are two solutions to the isolation problems; however, the first is expensive and the second violates the self-configuring tenet of the peer-to-peer structure. Hence, we propose the use of an overlay network based on links.

When node *X* is born, it needs to have one or more logical neighbors. If it has three neighbors, *A*, *B*, and *C*, then we say that it has three links, $X \rightarrow A$, $X \rightarrow B$, and $X \rightarrow C$. Unlike overlay networks used by some peer-to-peer systems, our link based network is not based on node names, but on user selected neighbors.

Now we discuss how these links are created. Node *X* links to node *A* if:

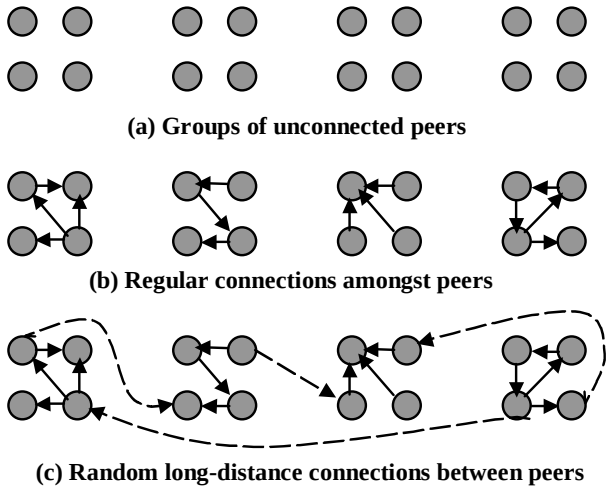
(i) *A* is a special node (server) designated by the domain for peer-to-peer links
(ii) *A* is a node, known to *X* that it trusts
(iii) *A* is a node that belongs to many communities *X* is interested in.

For a novice/new node, (i) may be the most appropriate link. As *X* ages, it finds other nodes and adds these links to improve search speed and information access. The linkages are similar to friendships in real life, or to http links in the Web and are directed by humans.

4.3 Small-World Networks

In the real world, peers form relationships (links) with other peers that have something in common with them. While most of these relationships develop between peers that exist within a common local area network, domain, country or some such location specific factors, a small set of random peers would acquire "long-distance" relationships with peers that are considered remote. The former scenario, termed as *regular connections*, is a direct result of the inclination to form groups as discussed earlier. The other type of connection is described as random because they link peers to other randomly selected peers that are not necessarily within any location proximity of each other. *Random connections* are caused by external human factors; like a company employee who uses her computer to occasionally connect to her bank, while simultaneously participating as a member of the P2P community with other company employees.

Such types of semi-random networks have been described by Watts and Strogatz [18] who studied the properties of large regularly connected graphs of nodes that contain a few random long-distance edges between nodes. They modeled this structure and demonstrated that the path-length between any two nodes of the graph is in fact surprisingly small. As a result, they called such semi-random structures as *small-world networks*.



(a) Groups of unconnected peers

(b) Regular connections amongst peers

(c) Random long-distance connections between peers

Figure 3: Creation of a P2P small-world network

4.4 Modelling P2P Communities

We now present our modeling (simulation) work on forming and discovering P2P communities. For this work, we first create a set of nodes, assign them attributes and then generate links. After that, we run our algorithms for community formation and community discovery and get results on the efficiency of these algorithms.

First, we create a very large number of peers each having a set of *personal attributes* with at least one element. To simplify the process, the attributes are implemented as different letters of the alphabet. Thus we have 26 possible attributes (A-Z), and for each node we pick between 1 to 26 attributes randomly and assign these to the personal attribute set.

Second, from the personal attribute set of each, we randomly select some attributes and make these the claimed attributes of that node.

Third, we randomly divide the set of peers into a set of groups. We assign to each node a group attribute that is same for all nodes belonging to a group.

Finally, we create the links. The link creation is a two-step process; (1) we randomly assign links that are between

group members (regular connections) and (2) we create some lesser number of inter group links (long distance connections).

Thus we generate a set of peers with attributes and links that is representative of a P2P network. We then run further simulations of our algorithms on this network. This process is illustrated in fig.3.

5. Formation and Discovery

We have previously stated an important trait of peer-to-peer communities: its membership depends on the relationships between peers that share common interests. These interest-based communities bear a strong resemblance to the user groups and chat rooms that we previously discussed. We describe the differences and give our formation and discovery algorithm for peer-to-peer systems.

In traditional centralized approaches, a server would be charged with maintaining a record of the interest attributes of each of its clients. New clients would merely register their interests at the server, which would periodically execute a matching algorithm to group peers that share common interests to form communities. This list of communities could be easily obtained from the server at any time. In this way, communities would be formed and clients would know their community membership in deterministic time. Existing examples of such systems are matchmaking servers and dating websites. As a bonus of a centralized approach, clients could also ask the server for a list of all the available communities without incurring any extra computational cost for server-side processing. Alternatively, a server would act as a central directory that lists all the available communities. New clients would then have to browse or search the directory for communities to join; or they could form a new community. A few other examples of these systems can be found in Internet chat sites [19, 20, 21].

In a distributed environment, such as a peer-to-peer network that lacks a central authority, this simple formation and discovery problem becomes much more complex. In order to compensate for the server, the cost of communication will be higher. Peers will need to exchange their interest attributes with each other for the purpose of forming communities and deriving their community membership. In addition, it is unlikely that any bonus information that was previously available for free due to the centralized approach will be similarly obtained without executing another round of communication.

We propose a community formation and discovery algorithm that works without any central authority and optimizes the cost of communication. Our algorithms are autonomous procedures that are executed asynchronously by each peer. Through a simple exchange of interest attributes, we demonstrate how communities can be formed. Further, without any additional communication, we show how peers can also discover exactly to which communities they belong.

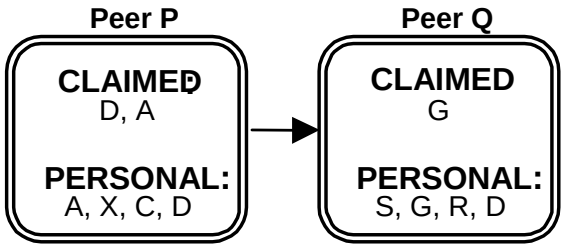
5.1 Forming Communities

The community formation algorithm assists in the establishment of community relationships amongst peers that share common interest attributes.

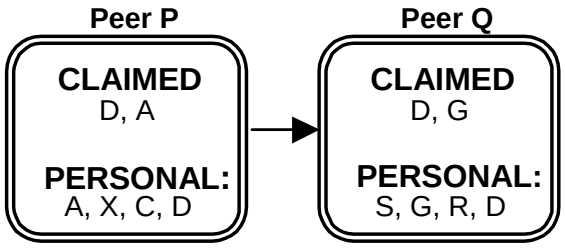
Normally, a network of peers can exchange or advertise their interest attributes with their neighbors in order to find peers with whom community relationships can be formed. There are two mechanisms here: an *active* approach, where a peer actively polls its neighbors for their interest attributes while simultaneously sending its own attributes to them; and a *passive* approach, where a peer merely advertises its interest attributes on its website and relies on the active approaches of other peers for the exchange. We use both these approaches in our community formation algorithm. Peers start by actively exchanging their interest attributes with each other and following the procedure that we lay out. Over time the communities formed will become out dated as peers change their interest attributes. Therefore, we propose that the formation algorithm be repeated periodically. In between these repetitions, we propose that a peer advertises its interest attributes using its website, shared network drive, or a web service. Due to the asynchronous nature of our algorithm, in a large network of peers there will always be some peers in an active mode, at any given time, which can take advantage of these advertisements.

The active exchange of interest attributes is not required in order to form communities. In fact, if all peers only operated in passive mode and tried to form communities based on their claimed attributes, we could significantly reduce the cost of communication. However, using this approach, it is possible that a set of peers that form part of the same group is not identified as a peer-to-peer community because all the peers did not claim (make public) their group attribute. The following situation could occur. A peer-to-peer community with signature X might exist, and a peer that had X in its personal attribute set but had not claimed it, would not be able to join this community and avail of the benefits until it claimed X.

Peers, therefore, need to expose (escalate from the personal list to the claimed list) as many attributes as possible to join the maximum number communities. This escalation can only be achieved by establishing an active communication with other peers. When no more attributes are being "escalated" by the peers, we say that the communication has



(a) Before any communication and adjustment



(b) After communication messages and adjustment

Figure 4: Procedure to escalate attributes (D in this case).

achieved its goal and the collection of peers is *stable*.

To achieve it, we provide an algorithm that uses *attribute escalations* as follows:

1. Each peer *P* communicates with all of its neighbors[2] and all of its neighbor's neighbors. The communication involves a transmission of the claimed attribute set by *P*. The reason we stop after the neighbor's neighbor is explained later.
2. Each peer *Q* receiving such a set from one of its incoming links will find the intersection of the set with its own personal attribute set.
3. If an element is present in the intersection set just computed and absent from the claimed attribute set of *Q*, then it is added to the claimed attribute set of *Q*.

An example with two peers is illustrated in fig.4, where two peers that are linked (shown by arrow) are considered for the illustration.

Note that the attribute escalation algorithm automatically escalates attributes from the personal set to the claimed set. This process may not be desired in practice and a human may be consulted before this escalation is performed. Initially, we assume automatic escalation.

The above procedure causes a change in each peer and therefore results in a change in the communities identified. Since these changes might occur frequently with peers being created, destroyed, or modified, we were concerned about the stability of the attribute escalation technique.

We conducted simulations to measure how many communication steps our algorithm requires to identify all the possible communities. In this simulation, we generate a P2P network using our generation method described in the previous section. The network consists of 10,000 peers and each 100 peers form a group. As described earlier, attribute selections and link formation are random. On the average, we found each simulated peer contains 13 personal attributes and 6 claimed attributes; and is connected to 66 peers through regular connections and 0.1 remote peers through random connections. As shown in fig.5, the average number of claimed attributes per peer stabilized *after one iteration* of the attribute escalation algorithm. This means that the number of times peers needed to communicate before the average number of claimed attributes for all peers becomes constant is just once.

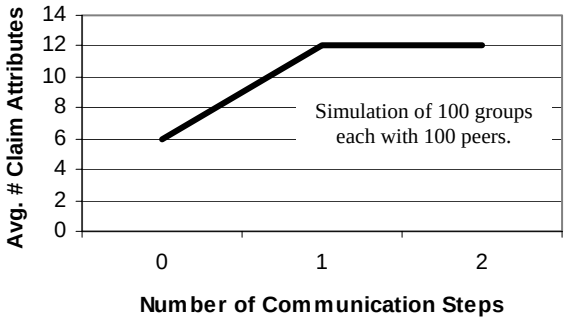


Figure 5: Stabilization of Attribute Escalations.

Further, the computation at each peer is a trivial set algebra. Hence, each new arrangement of a collection of peers is expected to stabilize quickly and with little overhead.

5.2 Discovery Of A P2P Community

Discovery in the context of peer-to-peer communities usually indicates one of the following: discovery of the communities in which the peer is automatically a member, or discovery of the existing communities that a peer might want to join or use. As indicated before, in this section, we focus on the former.

After the escalation of attributes, a peer has only facilitated the formation of communities. It still is not aware of its community memberships. At this stage, a peer knows two things about its interest attributes: the list of its claimed attributes, and the subset of the claimed attributes that were escalated. A peer can assume to be a member of communities that share an interest in the attributes that have been recently escalated. This assumption makes sense because the only reason that they were escalated was due to communication with a known peer (known directly or indirectly) that shares the same interest. However, the attributes of a peer that were already claimed before the escalation process may not be common with other known peers. The peer cannot, therefore, assume to be a member of a community based on these un-escalated claimed attributes. Only after analyzing the received interest attributes, will a peer be able to distinguish between: being part of an existing community, or being the initiator of a new community. The reason for this differentiation will become clear when we describe our analysis algorithm below.

Peers can use this analysis method to clearly discover their community memberships. Two kinds of communities will be discovered: the communities that peers are explicitly a part of, by virtue of their common group attribute or other claimed attributes: and the communities in which peers become members, by virtue of their claimed attribute set after escalations. We ran simulations to test the effectiveness of our algorithm in discovering peer-to-peer communities.

5.2.1 Definition and Algorithm

Before describing our discovery algorithm, we define two new terms:

OUTLINK WEIGHT: *The weight given to each claimed attribute based on the percentage of outgoing connections from a peer that can reach, after at most one indirection, other peers claiming the same attribute.*

INLINK WEIGHT: *The weight given to each claimed attribute based on the percentage of incoming connections to a peer that arrive directly from other peers claiming the same attribute.*

The constraint of at most one indirection is necessary to restrict the maximum depth up to which peers will be examined since more than two levels deep resulted in an unacceptably high number of communication messages. See Section 5.3 for more information. Further, our simulations have shown that by using only one level of indirection, peers find an average of eight communities. Probing to two levels of

[2] Neighbors are directly linked peers.

indirection is unnecessary, as it did not increase the average number of communities discovered by a peer.

These weights can be used in conjunction with a pre-defined threshold, after the stabilization step has been performed, to assist peers in discovering their community membership. If the outlink weight exceeds a threshold, the peer will assume that it is a member of the community indicated by the claimed attribute corresponding to that weight.

We define the membership of a peer for a community in a discovery process as follows:

MEMBERSHIP: *A node n is a member of a peer-to-peer community N with signature set S if S is a subset of claim(n) and every element of the intersection set I = claim(n) ∩ S has an outlink weight greater than threshold T.*

The procedural pseudo-code for our membership discovery is shown in fig.6.

```
MEMBERSHIP-DISCOVERY (peer)
/* executes at each peer simultaneously */

    for all claimed attributes
        compute outlink weight
        compute inlink weight
    end-for

    for all subsets S of claimed attributes set
        for all attributes in S
            if outlink weight > T then
                check next attribute
            else
                break-for
                /* try another subset */
        end-for

        if no break-for was executed
                peer is member of community with signature S
end-for
```

Figure 6: Community Discovery Algorithm

By including threshold T in the definition of membership, we can have various degrees of peer-to-peer communities. For instance, if the threshold is kept very low, then a community will be formed even with a small number of peers that share a common interest. If instead, it is desired that peer communities only form when there are many peers that share a common interest, the threshold can be set to a higher value. Although not required to be pre-set, the threshold value would give the network of peers a uniform behavior if all peers agreed upon its value.

Finally, we define two characteristics of peers that can be derived from the outlink and inlink weight computation. We do not use these definitions in this paper, but give them to hint at further uses for our discovery algorithm.

INVOLVEMENT: *The average of outlink weights for elements of the intersection set I, is directly proportional to the node's involvement in a peer-to-peer community.*

RESPONSIBILITY: *If a node n has a inlink weight associated with any member of the intersection set I, and the weight is greater than threshold B then the node is said to be a responsible member of the peer-to-peer community. The inlink weight is directly proportional to the responsibility of the node.*

The values of threshold B can differ for each community and this allows for various types of communities, like *cohesive communities*, with high peer involvement; *supportive communities*, with high peer involvement and high peer responsibility; *sprawling communities*, with low peer involvement.

5.2.2 Simulation Results

We measure the effectiveness of our algorithm by the difference in the number of peer-to-peer communities, in which a peer discovers membership.

We ran simulations on peer networks of two different sizes: one with 1,000 peers: and the other with 10,000 peers. The maximum number of personal attributes we allowed for each peer was 20, and a peer was explicitly placed into only one group, as described earlier. For both simulations, we set the outlink threshold to 40%.

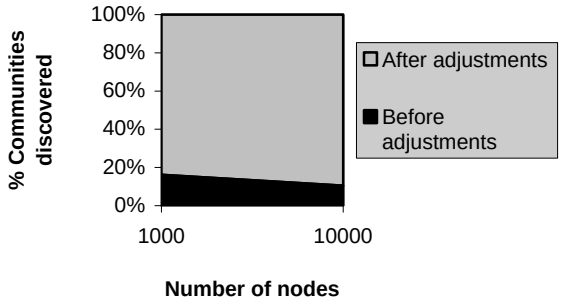


Figure 7: Percentage of communities discovered. The graph shows the average number of communities discovered by all peers before and after weigh calculation followed by attribute escalations. The outlink threshold used was 40%.

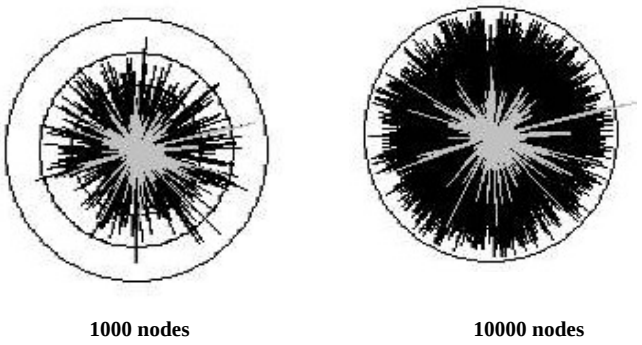


Figure 8: Number of communities discovered. The graphs show the number of communities discovered by the peers. The inner gray area is before and the outer black area is after weight calculation followed by attribute escalations. The outlink threshold used was 40%. The circles have values 0, 5, 10, and 15 from inside to outside.

The graphs shown in fig.7 and 8 are the results of our simulations. They demonstrate the performance of using the claimed attribute set and the outlink threshold for the discovery of peer-to-peer communities. As illustrated in fig.7, the majority of peers have discovered communities, other than the ones, in which they were explicitly placed. Our simulations have shown that before any communication, the average number of communities known to a peer was 0.5. This number increases to an average of 8.5 after the execution of the membership procedure at each peer.

5.3 Peer Reachability

We conducted experiments with 1,000 peers and 10,000 peers to traverse the outgoing links from each peer. We then counted the number of peers that could be reached after going to the direct neighbors (Level-1), in-direct neighbors after one indirection (Level-2), and in-direct neighbors after two indirections (Level-3).

Table 1
Number of peers reached in various levels (For 10,000 peers)

Max. Level traversed	Number peers reached
Level-1	65.8141
Level-2	4380.7874
Level-3	18929389.64

In fig. 9 the graph was plotted using a logarithmic *y*-axis. It shows how the number of peers that can be reached increases with greater depth.

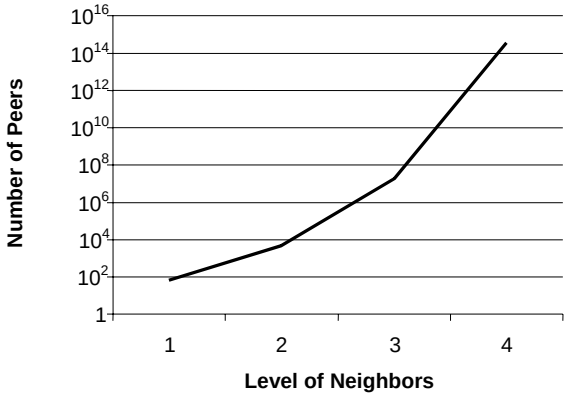


Figure 9. An exponential increase in the number of peers that can be reached from a peer

6. Discussion

Our techniques for P2P community formation and discovery can be applied to an existing collection of peers that already have links amongst each other such that they form a small-world network. Although various systems, like social networks, the Internet, and the collection of computer scientists demonstrate small-world characteristics, there is no guarantee that a P2P system such as a digital library would form a small-world network.

We solve this uncertainty by enforcing certain rules on new peers that want to join the P2P system. By virtue of these rules, the P2P system that forms is a small-world network and also exhibits a power-law characteristic for the distribution of the number of neighbors of each peer.

Initially, a very small number *N* (we ran successful simulations with 2 peers, 5 peers and 10 peers) of fully connected peers is chosen. These peers are the starting seed of the P2P system and would be setup in an explicit manner by a group or an individual person. Any peer that wants to join the P2P system henceforth needs to know at least one peer that is already a part of the network. The incoming peer forms links to the selected peer and *N-1* peers chosen from the selected peer's neighbors with some probability distribution. The neighbors of the selected peer that have more links to other peers are chosen with a higher probability than other neighbors.

This algorithm allows a P2P system to form and then grow in a scalable manner. Our simulations showed that the resulting networks generated had a low characteristic path length comparable to a corresponding random network; however, the value of the clustering coefficient remained as high as that of networks with *N* regularly connected peers.

If a P2P software program were developed, it could enforce these simple rules to ensure that the P2P network generated would exhibit small-world network characteristics. Then this software could avail of our techniques to structure the peers into communities that can be efficiently formed and discovered.

7. Conclusions

P2P communities, is a method for arranging large numbers of peers in a self configuring peer relationship based on declared attributes (or interests) of the participating peers. This method is expected to have an impact in sharing of resources and pruning of search spaces based on the interests of the clients.

This paper shows that the attribute based clustering of peers can be made to work, by defining an overlay network, consisting of links. Links are user-directed connections based on experience and can be fine tuned for search and sharing performance. We show that the peer formation algorithm stabilizes in two rounds using the escalation technique. Additionally, we introduced an efficient community discovery procedure using weights and threshold. Our simulations of the discovery procedure have confirmed that peers can quickly discover numerous memberships in different P2P communities using very little computation and communication messages

We are currently working on extending our results in peer communities to include fast and directed search algorithms and algorithms that handle privacy, security and trust management.

References

[1] Gnutella. http://www.gnutelliums.com/

[2] I. Clarke, O. Sandberg, B. Wiley, & T. W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Workshop on Design Issues in Anonymity and Unobservability*, Berkeley, CA, USA, July 2000, 311–320.

[3] http://www.napster.com

[4] Yahoo Groups. http://groups.yahoo.com/

[5] H.A. Murray, *Explorations in personality* (Oxford University Press, New York, 1938).

[6] L. Festinger, Laboratory Experiments: The Role of Group Belongingness, in J. G. Miller, ed., *Experiments in Social Process* (McGraw-Hill, New York, 1950).

[7] G.W. Flake, S. Lawrence, C.L. Giles, & F.M. Coetzee, Self-Organization and Identification of Web Communities. In *IEEE Computer, 35*(3), March 2002, 66-71

[8] K. Jiang & P. Dasgupta, SIMS: A Secure Information Management System for Large-Scale Dynamic Coalitions, In *The 2nd. DARPA Information and Security Conf. and Exposition* July 2001.

[9] B. Yang and H. Garcia-Molina. Efficient Search in Peer-to-peer Networks. In *International Conf. On Distributed Computing Systems*, Vienna, Austria, 2002.

[10] J. Scott, *Social network analysis: a handbook* (SAGE Publications, 1991).

[11] E. Garfield, *Citation Indexing: Its Theory and Application in Science* (Wiley, New York, 1979).

[12] H.D. White and K.W. McCain, Bibliometrics, In *Annual Review of Information Scence and Technology*, Elsevier, 1989, 119-186.

[13] G.W. Flake, S. Lawrence, and C.L. Giles, Efficient Identification of Web Communities, *Proc. 6th International Conf. on Knowledge Discovery and Data Mining*, ACM Press, New York, 2000, 150-160.

[14] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proc. of the 9th Annual AVM-SIAM Symp. on Discrete Algorithms*, 1998, 668-677.

[15] L. Page. PageRank: bringing order to the web. *Stanford Digital Libraries working paper* 1997-0072, 1997.

[16] D. Gibson, J. Klienberg, and P. Raghavan. Inferring web communities from link topology. In *Proc. 9th ACM Conference on Hypertext and Hypermedia*, 1998.

[17] M.R. Garey and D.S. Johnson. *Computers and intractability: A guide to the theory of NP-completeness* (W.H. Freeman, New York, 1990).

[18] D. Watts and S. Strogatz, Collective Dynamics of "Small-World" Networks. *Canadian J. Math*, vol. 8, no. 3, 1956, 399-404.

[19] http://chat.yahoo.com

[20] http://chat.msn.com

[21] http://www.icq.com

Biographies

M.S. Khambatti is a PhD candidate in computer science and engineering at the Arizona State University. His research interests include peer-to-peer systems and distributed operating systems. Khambatti received a Masters in computer science and engineering from the same university and a Bachelors in computer engineering from the University of Pune, India. He is the winner of 3 national awards for his senior year project in 1999. He has been the chair of the IEEE Computer Society student chapter at ASU during 2001-02 and has also been the co-chair of the Workshop on Research in Computer Science 2002.

K.D. Ryu received the BS and MS degree in computer engineering from Seoul National University in Korea, in 1993 and 1995 respectively. He received the PhD degree in computer science from University of Maryland at College Park in 2001. He is an assistant professor in the Department of Computer Science and Engineering at the Arizona State University. His current research interests include grid-computing, peer-to-peer computing and embedded system performance tuning. Dr. Ryu's current projects include COIN peer-to-peer computing infrastructure, and -Watch performance monitoring and tuning tool for embedded systems. Dr. Ryu is a member of IEEE Computer Society and the ACM.

P. Dasgupta received his Ph.D. in Computer Science in 1984 from the State University of New York. His prior education was at the Indian Institute of Technology. He is on the faculty of Arizona State University, and teaches courses in Applied Cryptography and Distributed Operating Systems. He has over 15 years of research experience in Distributed Computing, Operating Systems, Security and Networking. His research is funded by NSF, DARPA, AFOSR, Microsoft and Intel.