A Distributed Hybrid Scheme for Unstructured Peer-to-Peer Networks

Yong Tang Zhan Zhang Shigang Chen
Department of Computer & Information Science & Engineering
University of Florida, Gainesville, Florida 32611–6120
{yt1, zzhan, sgchen}@cise.ufl.edu

Guangbin Fan
Communication Technology Lab
Intel China Research Center, Beijing, China
guangbin.fan@intel.com

***Abstract*— Peer-to-peer (P2P) networks have gained a lot of popularity in recent years. While structured (DHT) networks provide better response time and smaller diameter, which are the advantages over unstructured networks, they are vulnerable to frequent node failure/joins/leaves. There is always a tradeoff for any P2P network to achieve all of these goals. In this paper, a distributed hybrid scheme is proposed for unstructured P2P networks, which combines Markov random walks and peer clustering to achieve a better tradeoff. The scheme has a short response time for most of the queries that belong to the same interest group, while still maintaining a smaller network diameter. More important, we propose a totally distributed clustering algorithm, which means better resilience to network dynamics. The performance of our systems is demonstrated by extensive simulations.**

I. Introduction

Over the past several years we have seen a rapid growth of the applications in Peer-to-Peer (P2P) systems such as data sharing, directory lookup, software distribution, timeshared storage, and distributed indices [1]. The fundamental challenge of constructing a P2P network is to achieve faster response time, smaller network diameter, better resilience to network dynamics, and higher security.

Most recent P2P proposals [1], [2], [3], [4], [5], [6], [7] use distributed hash tables (DHT) to provide data location management in a strictly structured way. Although they offer better performance in response time and network diameter, they are not resilient to frequent node leaves/joins/failure. In addition, DHTs are inflexible in providing generic keyword searches because they have to hash the keys associated with certain objects [8], and they are more vulnerable to the existence of malicious nodes since the identifier associated with the data item is not mapped by the node that provides it.

In unstructured networks such as Gnutella and KaZaA, nodes are interconnected in a random manner, which offers high resilience for the whole network to be tolerable to frequent node leaves/joins/failure. However, basic unstructured networks rely on flooding for user's queries which is expensive in computation and communication overhead. Therefore, scalability has always been a major weakness for unstructured networks [9], [10]. Other P2P architectures such as Morpheus [11] and KaZaA [12] improve the search in their design by designating super-nodes to cache the indices of others. However, introducing the super-nodes causes other problems such as load balancing for P2P networks. Recently, searching through random walks has been proposed [8], [13], [14]. The main problem of random walks is the high network diameter (delay). Although multiple random walks is a way to lower the network diameter, the approach is still problematic when the number of initiated messages is large [14].

It is difficult to simultaneously satisfy all of these requirements and there is a tradeoff in the design space of P2P systems. We believe unstructured systems are more promising due to the inherent restrictions of structured P2P networks.

The work presented in this paper takes the unstructured approach and tries to deliver better tradeoff in design space, especially the fast response time and low diameter that a typical unstructured one lacks. The philosophy behind the design scheme is based on the observation that nodes with similar interest are more likely to share data items among them. We define the concept of similar interest as an association between two nodes. The network is partitioned into clusters with high intra-cluster association. Fast response time and low diameter are achieved by doing random walk on inter-cluster path while flooding the same cluster. More importantly, all these can be done without assuming the global knowledge for each node. At the same time, desirable properties of unstructured network have been retained in our system.

II. Association Graph

A. Problems and Motivation

Most of the recent research have been focusing on constructing the P2P network based on the content of the data items in each node e.g., [15], [16]. Interest shortcut [15] exploits the locality of interest among different nodes. Although the idea is smart, it is asymmetric, that is, node u is interested in data items on node v does not mean vice versa. The concept is also vague. It is difficult to make a subtle, quantitative definition based on it. The associative overlay [16] groups the nodes sharing the same guided rule. However, it relies on a global unique list of guided rules. Even if such a list is available, it has to be updated constantly in order to deal with the dynamics of the interest shift in a P2P environment.

A right step toward the measurement of the similarity on two nodes' interests in a distributed environment is the concept of indexing through semantic space [17]. In this approach, each node is assigned a vector in the latent semantic space. The

1-4244-0355-3/06/$20.00 (c) 2006 IEEE

metric of dissimilarity (distance) is defined as the inner product of the two vectors that correspond to two nodes. Such a method can be applied to DHT-based P2P networks [17]. It can also be used in unstructured P2P networks where nodes that are semantically close to each other form the cluster. Networks with node clustering outperform randomly connected ones by using random walks [13].

There are several unresolved issues for the method above in P2P networks. First of all, the dimensionality in a semantic space is typically high, which means the overhead might be intolerable. Although Singular Value Decomposition (SVD) was used [17] to derive low-dimensional representations, the method is not scalable in terms of both memory consumption and computation time [18]. Secondly, the establishment of the semantic space requires the global information of all data items within the network to form the input matrix for SVD, which is impractical in a typical P2P network considering the possible massive number of nodes and the dynamics of the networks. Finally, the data items resided in a node may not necessarily reflect the node's interest. In addition, one node may have many data items covering a variety of distinct topics. It is conceivable that fitting these topics into one single vector is difficult unless the dimensionality of the semantic space is higher than total number of distinct topics.

With the all these drawbacks of the existing approaches mentioned, We propose a P2P system that tries to solve the drawbacks of mentioned approaches. The basic design scheme of our system is to cluster nodes with similar interest. Each node is aware whether or not the neighbors belong to the same cluster. Upon receiving a query message from the neighbor outside the same cluster, the node will flood the query message within the same cluster while at the same time forward the query message to one of its inter-cluster neighbors randomly. In other words, query messages are handled by intra-cluster flooding and inter-cluster random walks. The logic behind is that the total number of clusters is greatly smaller than the total number of all nodes, thus the network diameter of inter-cluster random walks is limited while intra-cluster flooding is more efficient and can be scaled up with limited cluster size.

B. Weighted Association Between Nodes in P2P Networks

Suppose there are n data items $D = \{d_1, d_2, ..., d_n\}$ available in the whole P2P network. Node u accessed a set of n^u data items $D^u \subset D$ and peer v accessed a set of n^v data items $D^v \subset D$, with $n^u, n^v \leq n$. If node u and node v share similar interests, then it is very likely that they accessed same data items more or less previously. By saying "similar interest", we actually mean that two nodes u and v have a common subset of accessed data items, i.e., $D^u \cap D^v \neq \emptyset$.

As is illustrated in Figure 1, in a P2P network, those nodes with identical interests should access the same data items both before and in the future. Those nodes whose interests are totally different will never access the same data item. Therefore, the size of the common subset $D^u \cap D^v$ can be served as a metric measuring to what extent the interests of two nodes are similar.

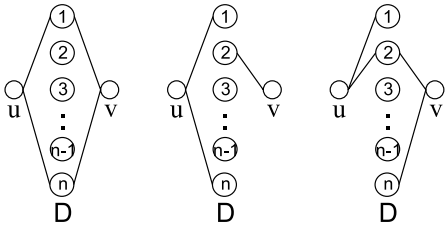

Fig. 1. Association representing similar interests between nodes. left: identical interests, center: totally distinct interests, right: in-between

We formally define *association* as the measurement of similar interests between nodes. Nodes u and v are assigned with two *access vectors* with n elements respectively, i.e., V^u and V^v. Each element within the assigned access vectors is from the set $\{0, 1\}$, with 0 at the i-th row indicating that the i-th data item was not accessed before and 1 vice versa. The *association* $A^{u,v}$, defined as follows, is the inner product of access vectors V^u and V^v, where V^{vT} indicates the transpose of V^v.

$$A^{u,v} = V^u \cdot V^v = V^u V^{vT}$$

To further explain our idea, we give an example based on Figure 1 (right). As we can see, the access vectors assigned to nodes u and v are

$$V^u = \begin{pmatrix} 1 \\ 1 \\ . \\ . \\ 0 \end{pmatrix} \text{ and } V^v = \begin{pmatrix} 0 \\ 1 \\ . \\ . \\ 1 \end{pmatrix}$$

Therefore, the association between nodes u and v can be computed by

$$A^{u,v} = V^u \cdot V^v = \begin{pmatrix} 1 \\ 1 \\ . \\ . \\ 0 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \\ . \\ . \\ 1 \end{pmatrix} = V^{uT} V^v = 1$$

The equation above clearly demonstrates why it is appropriate to use inner product to compute the similarity of interests (association) between nodes. As is shown in the equation, the inner product of access vectors is the summation of all the products at each row. Only when both of the elements at the same row are 1's, will the product of this row contribute to the total summation. This is the case when both two nodes accessed the same data items.

The problem with this definition is that different properties of data items have not been taken into consideration. For instance, data item 1 might be a very large media file of several hundreds MBs while data item 2 might be only a PDF file with several MBs, In addition to that, each peer has to have a complete list of all available data items.

We slightly modify the association into *weighted association* to account for these issues. In this definition, the rows in the access vector that corresponds to the data items not accessed

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

by u and v will be removed. A diagonal matrix with (i,i)-th value $w_{i,j}$ specifying the weight for i-th data item is introduced. Suppose the weight of the data items in Figure. 1 is 0.5, 0.2, 0.3 for data items 1, 2 and n, the weighted association is defined as follows.

$$\begin{aligned} A_w^{u,v} &= V^{uT} W V^v \\ &= \begin{pmatrix} 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0.5 & 0 & 0 \\ 0 & 0.2 & 0 \\ 0 & 0 & 0.3 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} \\ &= 0.2 \end{aligned}$$

Here W is normalized by the summation of the total weight to cancel out the influence of the total number of data items.

The definition of association is advantageous in manifold. First, it is more elegant than interest shortcut. Not only are we looking for the locality of the interest, but also the exact meaning of the similar interest has been defined. Second, instead of having a global unique list, two peers only need to have a list of data items shared by them, i.e., $D^u \cup D^v$, to compute the association. Third, there is no ambiguity between the node's interest and the data items resided in it, as is the case in latent semantic indexing. Instead, similarity of interest is defined as the common subset of the data items that these two nodes accessed. Finally, the definition is symmetric, that is, we have $A_w^{u,v} = A_w^{v,u}$, which is more appropriate in term of the definition of the similarity for interests between two nodes.

C. Association Graph

A weighted association graph is a weighted undirected graph $G = (V, E)$ consisting of a set V of $nodes$ and a set E of weighted $edges$ connecting pairs of nodes. The weight $w_{(u,v)}$ of edge (u, v) specifies to what extent node u has similar interest with node v. More strictly, the weight is defined as the weighted association between node u and node v, obtained from the query history of the nodes. Such a graph is an overlay structure built on top of unstructured P2P networks such as Gnutella.

Problem arises during the construction process. It is impossible to compare each pair of nodes in order to obtain the associations between nodes. We have to find a way to start with and limit the number of node pairs dealt with. We introduce the concept of topology adaption here. In this strategy, the associations are first computed between the nodes and their direct neighbors of their underlying network. After that, each time a query message is processed, the association between the querying node itself and the node that owns the data items will be computed. The newly obtained association is inserted into the list of the previously stored addresses of nodes and their associations. The stored node with the lowest association is dropped.

By assuming that interests of nodes will not shift in a limited time frame, we can piece together a view of the interest relationships on the P2P network after extracting the graph from the history. The interest relationship between nodes in the past provides further directions for our predictions in the future.

The association graph is to construct an unstructured P2P network with good searching properties. Since nodes with similar interests tend to visit each other frequently in P2P networks, the web graph based on the weighted association will likely connect those nodes with weights larger than other nodes. If we think of an edge with a weight greater than one as multiple edges connecting two nodes, then those nodes with similar interest will be connected with higher degrees than average. To use interest-based web graph serving our goal, we make the assumption that, if node u is "interested" in node v and node v is "interested" in node w, then it is likely that u is "interested" in w, compared to another random node. Thus, nodes in the same community will form clusters by nature. In fact, our methods is a better solution than most of the existing algorithms [13] in that it can be implemented in a totally distributed way.

III. Markov Random Walks and Distributed Clustering Algorithm

A. Design Scheme: A Hybrid of Random Walks and Message Flooding

Random walks have been proposed to replace the flooding techniques [14], it reduces the message overhead at the cost of increasing the network diameter (delay). [13] considered node (peer) clustering as a way to increase the performance of random walks. Node clustering means dense connectivities among nodes in the same community, and vice versa. The association graph itself is node clustering if we consider the edge with weight w between two nodes as w independent edges with weight 1. Thus, it can be fully utilized by slightly modifying the typical random walks.

Consider the association graph $G = (V, E)$ with weighted association between node i and j as $A_w^{i,j}$, the transition probabilities $p_{i,j}$ for the random walk is defined as proportional to the weight of the edges.

$$p_{i,j} = \frac{A_w^{i,j}}{\sum_k A_w^{i,k}}$$

If the transition matrix P is organized with its i,j-th element as $p_{i,j}$, then rows in P sum to 1 as P is row stochastic. The transition probabilities are generally not symmetric because the normalization of the association matrix varies.

Although the transition matrix P of the whole graph is discussed, it is only for the purpose of representation. There is no need to compute the whole matrix for any one of the nodes in the network. For any node i it is enough to only keep a transition vector $\mathbf{p_i} = (p_{i1}, p_{i2}, ..., p_{in})$ describing the transition probabilities for its n direct neighbors $1, 2, ...n$.

The underlying philosophy of Markov random walks is that there is a preference in the searching process for those nodes that are likely to share similar interests with the node that initiates the query. In fact, in a real P2P environment, when a user issues a request, the user will check the cache containing the addresses of nodes that answer the previous queries [19].

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

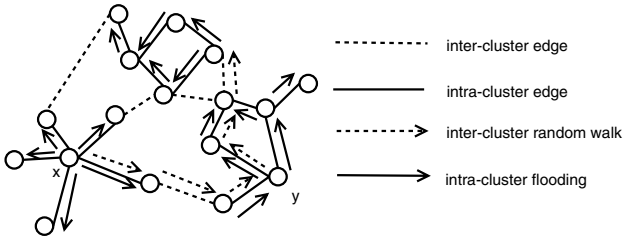

Fig. 2. Hybrid Scheme: Combining message flooding and random walk

One of the main disadvantages of random walks for data item searching is the high network diameter (delay) compared to typical flooding techniques. The fundamental of our design scheme is to combine random walks and message flooding to have a better tradeoff intuitively.

The design scheme can be best illustrated by Figure 2. The network is partitioned into clusters by labeling each edge as *inter-cluster* or *intra-cluster*. When a user of node *x* makes a request, the system will rely on two different strategies following two different kinds of edges at the same time. For intra-cluster edges, the query message will be forwarded to every node within the cluster by flooding. For the node *x* that initiates the search, the query message will be forwarded to one of the inter-cluster neighbors (connected by inter-cluster edges) randomly. The random walk will repeat each time a node receives a query message from its inter-cluster neighbor or initiated by itself. In the case that the node has no other inter-cluster neighbor besides the one from which the node receive the message (e.g., node *y* in the figure), the message will be marked and forwarded to one of the node's intra-cluster neighbors until an inter-cluster edge exists.

It is expected that the majority of the queries will be satisfied by an intra-cluster search while an inter-cluster search is more likely to succeed for those queries looking for data items outside the main interest scope of the user.

B. A Fully Distributed Clustering Algorithm

It is of the most importance to have a fully distributed clustering algorithm that is able to label the inter- and intra-cluster edges when only the information from the direct neighbors is available. Over the past years, clustering in a graph has been studied intensively [20]. However, they are not suitable in a P2P environment in general. First, these methods are basically centralized since a global knowledge of the whole network is required. Second, these methods require a metric describing the pairwise distance between any two nodes. In a P2P network a node can only obtain such a metric for their direct neighbors or limited number of nodes. It is impossible for one single node to store every node within the whole P2P network.

In [21], a centralized clustering algorithm based on the Markov random walks has been proposed. We revisit the Markov random walk and try to find a distributed format of the clustering algorithm that can be used in P2P networks. Following the notations described above, we use $\Pr^t(j|i)$ to denote the transition probability that node i traverses node j after t steps. Since we already know that the one step transition probability matrix P, $\Pr^t(j|i)$ can simply be calculated by a matrix power:

$$\Pr^t(j|i) = [P^t]_{ij}$$

$\Pr^t(j|i)$ indicates the probability for a Markov random walk starting at i and ending at j with exactly t steps. To calculate the probability for a Markov random walk starting at i and ending at j *within* t steps, all possible steps should be summed up as

$$\sum_{k=1}^{t} \Pr^k(j|i)$$

Apparently, if two nodes i and j (not necessary the direct neighbor) shares similar interests, then a Markov random walk starting at i will have a larger possibility $\sum \Pr^k(j|i)$ to hit j with in a limited step t. If i and j share no similar interest, then the process will have a lower possibility since here a Markov random walk that starts at i will likely return to i before reaching j. If we want to partition the nodes within the network into clusters by their interest, then $\sum \Pr^k(j|i)$ can be served as a good metric for pairwise distance between any two nodes, even if only the information of direct neighbors (one step transition probabilities) is known. $\sum \Pr^k(j|i)$ is not suitable for pairwise metric because it is not symmetric. A symmetrical, collaborating one is used instead:

$$f(i,j) = \sum \Pr^k(j|i) \sum \Pr^k(i|j)$$

The distance restriction t works as TTL to control the diameter of the cluster and thus the total network diameter (delay) to fit the case in our proposed scheme. Furthermore, a threshold, $threshold$ for $\sum \Pr^k(j|i)$ has to be introduced to determine whether or not u and v belong to the same cluster. However, it also introduces arbitrariness in deciding the clusters. This problem can be solved by repeating Markov random walk several times so that the distinction between intra-links and inter-links will be sharpened. Only a very small number $threshold \to 0$ is needed for the threshold in this case.

We formally propose a distributed clustering algorithm applying the approach discussed above. Instead of computing $\sum \Pr^k(j|i)$ in a centralized way by transition matrix, the Markov random walk process is simulated to obtain the probability between any two direct neighbor peers and decide if the link connecting these two peers is an intra-cluster link or an inter-cluster link. The algorithm works as follows.

First, at node i, for each edge (i,j), a ticket that corresponds to t is generated. The ticket consists of a source ID $sourceID = i$, time-to-live parameter TTL, and a value $value$ specifying the probability a random walk will pass through this edge. The $value$ is the weight (weighted association) of the edge (i,j) normalized by the total weight for any edge originated from i, that is,

$$value = \frac{A_w^{i,j}}{\sum_j A_w^{i,j}}$$

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

Upon receiving of the ticket from one of its neighbor, node j will send the ticket back to the node $sourceID$ if there is an edge between $sourceID$ and j. Then node j will check to see if TTL has been set to zero. If the TTL is zero, the ticket will be dropped. If the the TTL is not zero, node j will reduce the TTL value by 1. The ticket will be flooded to every edge that node j connects. For any one of the edge (j, k), the $value$ in the ticket will be multiplied by the probability that node j will chooses (j, k) in a Markov random walk. That is, $value$ will be replaced by $value'$ with

$$value' = value \times \frac{A_w^{j,k}}{\sum_k A_w^{j,k}}$$

At the end of the process, any one of the node, e.g., i, will have collected all tickets (with $sourceID = i$) passing through hops that is less than the TTL. They should also know which neighbor forwards the ticket back. Node i can easily sum up the value in the ticket for every edge connecting to its neighbors. The two nodes i, j can either make a decision about whether or not the edge (i, j) is intra-cluster by comparing the obtained value with a threshold, or further forward the ticket with the value summed up to their corresponding edges.

We want to emphasis that, although a distributed clustering algorithm [22] has already been proposed, it is believed that the algorithm proposed in this paper is a better solution. In [22] each node has to maintain a table for every message passing by. It is not a totally distributed algorithm because originators have to be selected first and better performance can only be achieved by placing one originator for each cluster before hand. In contrast, our algorithm is stateless and totally distributed, in which there is no head node or authority in a cluster. It indicates that our algorithm can adapt to the network dynamics efficiently.

IV. Simulation

In this section, the performance of the clustering is studied by simulations. If not explicitly defined, the default value of the number of nodes is $10,000$, the average number of neighbors each node has is 20, the threshold ($threshold$) is 0.025, the TTL is 5, the average query number is 50, the number of nodes in a interest group is 100 and for each node 90% of its queries falls into its own interest group.

The performance of the flooding, random walk, and clustering algorithms are compared, with respect to the number of hops and messages. In Figure 3, it is observed that in the clustering algorithm, the number of hops needed for the majority of the queries is significantly reduced to about 10, when compared with the random walk algorithm. As is expected, the flooding algorithm needs a smaller number of hops for almost every queries than the clustering algorithm. However, the clustering algorithm is still slightly better if a query falls into the interest group itself. In addition to that, the clustering algorithm works much better than the other two algorithms with respect to the number of messages, as is shown in Figure 4. Even if the query does not fall into the interest of the node itself, there is still a great probability that the query

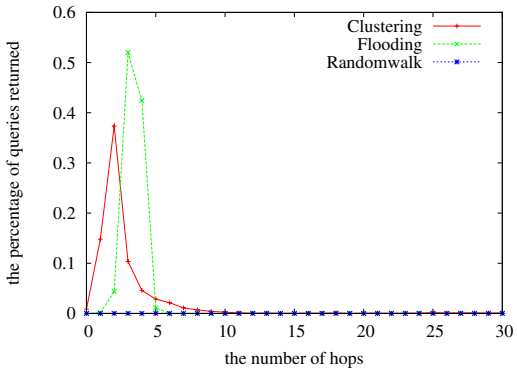

Fig. 3. The percentage of queries returned vs. the number of hops

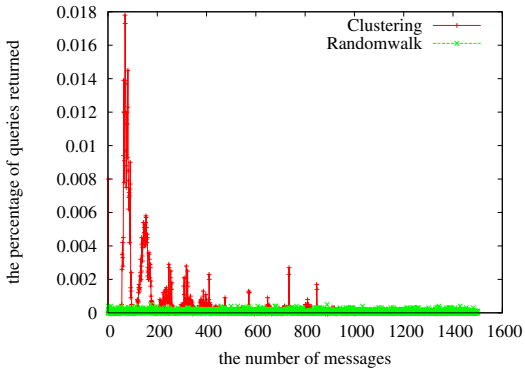

Fig. 4. The percentage of queries returned vs. the number of messages

will be returned within a limited number of messages. The total number of messages is decided by how far away between the interest group of the initiating node and the interest group of the targeted node. If a query falls into the interest of the node itself, the response time is very short for the clustering algorithm. For other cases, a longer response time is acceptable as long as these cases do not happen frequently.

How the cluster size is affected by various parameters is also investigated. In Figure 5, when the threshold is large, the cluster size approaches to 1. In this situation, the clustering algorithm is similar to the random walk. When the threshold

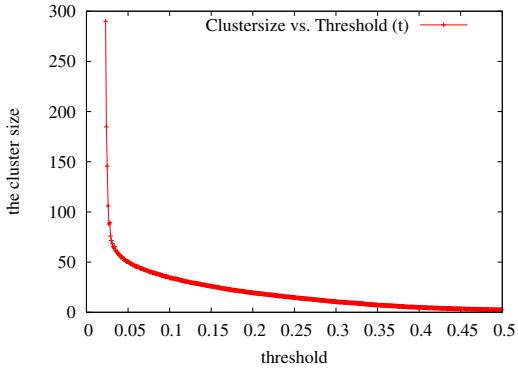

Fig. 5. Threshold vs. cluster size

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

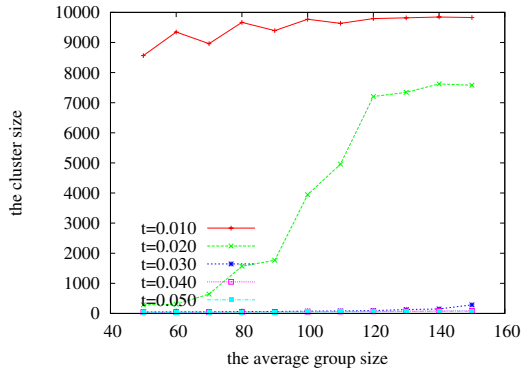

Fig. 6. Query number vs. cluster size

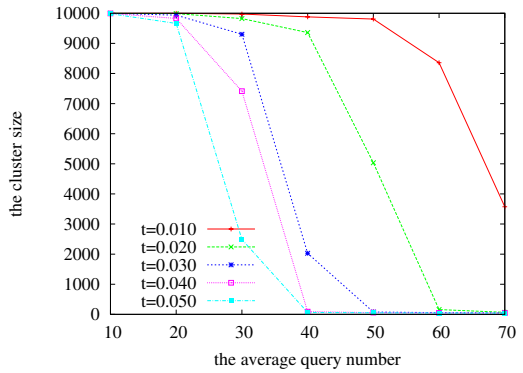

Fig. 7. Query number vs. cluster size

is small, the cluster intends to cover the whole network, the clustering algorithm is similar to the flooding algorithm. Thus, the value of the threshold plays an important role. In fact, in this paper, the threshold is set to be a system-wide parameter. A more delicate approach is to enable each member in the network to decide its threshold independently based on its query times, and the probability variation of its neighbors.

How the group size affects the cluster size is shown in Figure 6. It is observed that when the average group size is increased, the cluster size increases as well. Furthermore, when the threshold is larger than or equal to 0.3, the cluster size increases dramatically along with the increment of the average group size, which has been discussed in the last section.

Figure 7 shows the changes of the number of queries with respect to the cluster size. When the number of queries increases, the cluster size decreases dramatically. It means that, for a random network topology, a structured cluster can be formed within a short time. It also indicates that it is more advantageous to enable a node to decide its threshold independently.

V. Conclusion

In this paper, a hybrid method for unstructured P2P networks that combines the Markov random walk and the clustering algorithm has been proposed. It achieves a better tradeoff among response time and network diameter of the P2P network. A totally distributed clustering algorithm is also presented. The algorithm gives P2P networks better resilience to network dynamics. It can potentially be used in other scenarios when centralized clustering is not possible. The performance of the algorithms is demonstrated by simulations.

References

[1] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: a scalable peer-to-peer lookup protocol for internet applications," *IEEE/ACM Trans. Networking*, vol. 11, no. 1, pp. 17–32, 2003.

[2] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker, "A scalable content-addressable network," in *ACM SIGCOMM'2001*. ACM Press, 2001, pp. 161–172.

[3] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. Kubiatowicz, "Tapestry: A resilient global-scale overlay for service deployment," *IEEE J. Select. Areas Commun.*, vol. 22, no. 1, pp. 41–53, 2004.

[4] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," in *Middleware'2001*, Nov. 2001.

[5] C. G. Plaxton, R. Rajaraman, and A. W. Richa, "Accessing nearby copies of replicated objects in a distributed environment," *Theory of Computing Systems*, vol. 32, no. 3, pp. 241–280, 1999.

[6] D. Malkhi, M. Naor, and D. Ratajczak, "Viceroy: a scalable and dynamic emulation of the butterfly," in *ACM PODC'2002*. ACM Press, 2002, pp. 183–192.

[7] A. Kumar, S. Merugu, J. Xu, and X. Yu, "Ulysses: A robust, low-diameter, low-latency peer-to-peer network," in *IEEE ICNP'2003*, Nov. 2003.

[8] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker, "Making gnutella-like p2p systems scalable," in *ACM SIGCOMM'2003*. ACM Press, 2003, pp. 407–418.

[9] J. Ritter. (2001) Why gnutella can't scale. no, really. gnutella.html. [Online]. Available: http://www.darkridge.com/ jpr5/doc/

[10] K. Sripanidkulchai. (2001, Feb.) The popularity of gnutella queries and its implications on scalability. gnutella.html. [Online]. Available: http://www.cs.cmu.edu/ kunwadee/research/p2p/

[11] Morpheus. (2002) Morpheus file sharing system. [Online]. Available: http://www.musiccity.com/

[12] KaZaA. (2002) Kazaa file sharing network. [Online]. Available: http://www.kazaa.com/

[13] C. Gkantsidis, M. Mihail, and A. Saberi, "Random walks in peer-to-peer networks," in *IEEE INFOCOM'2004*, Mar. 2004.

[14] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, "Search and replication in unstructured peer-to-peer networks," in *Proceedings of the 16th International Conference on Supercomputing (ICS '2002)*. ACM Press, Sept. 2002, pp. 84–95.

[15] K. Sripanidkulchai, B. Maggs, and H. Zhang, "Efficient content location using interest-based locality in peer-to-peer systems," in *IEEE INFOCOM'2003*, Mar. 2003.

[16] E. Cohen, A. Fiat, and H. Kaplan, "Associative search in peer to peer networks: Harnessing latent semantics," in *IEEE INFOCOM'2003*, Mar. 2003.

[17] C. Tang, Z. Xu, and S. Dwarkadas, "Peer-to-peer information retrieval using self-organizing semantic overlay networks," in *ACM SIGCOMM'2003*. ACM Press, 2003, pp. 175–186.

[18] C. Tang, S. Dwarkadas, and Z. Xu, "On scaling latent semantic indexing for large peer-to-peer systems," in *ACM SIGIR'2004*. ACM Press, 2004, pp. 112–121.

[19] A. Oram and A. Oram, *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*. O'Reilly & Associates, Inc., 2001.

[20] R. Kannan, S. Vempala, and A. Vetta, "On clusterings: Good, bad and spectral," *J. ACM*, vol. 51, no. 3, pp. 497–515, 2004.

[21] D. Harel and Y. Koren, "Clustering spatial data using random walks," in *ACM SIGKDD '2001*. ACM Press, 2001, pp. 281–286.

[22] L. Ramaswamy, B. Gedik, and L. Liu, "Connectivity based node clustering in decentralized peer to peer networks," in *Proceedings of the third IEEE Conference on Peer-to-Peer Computing (P2P '2003)*, 2003.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.