Community based Search on Power Law Networks

Tathagata Das*, Subrata Nandi† and Niloy Ganguly‡
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur
Kharagpur, West Bengal - 721302, India
*tathagata.das1565@gmail.com, †snandi@cse.iitkgp.ernet.in, ‡niloy@cse.iitkgp.ernet.in

Abstract—**In this paper, we have presented a novel algorithm for searching Peer-to-Peer (P2P) networks which forms power law topology. It is completely decentralized in nature and hence is implemented independently by each individual peer participating in the network. Instead of flooding mechanisms, it uses random walk and proliferation techniques to search for peers having similar interests. Besides the search, it is also equipped to change the neighborhoods of the peers based upon their proximity with the queried item. This topology evolution coupled with search proliferation helps the P2P network to form interest-based communities, as a result of which the search efficiency of the network improves, as more and more individual peers perform decentralized search.**

Keywords- community structure, power law, semi-structured P2P

I. Introduction

Efficient search in unstructured P2P networks is becoming a real challenge, as the P2P user base is increasing exponentially. Search efficiency can be improved upon by maintaining some information from previous search experiences locally in the peer nodes. Another alternative is to restructure the network such that the nodes containing similar content or data profiles are moved closer to each other. The idea essentially is to go for a community-based semi-structured network. P2P network, being an appropriate example of socio-technological networks [2], inherently has the potential for developing a community structure. Therefore, the later approach seems to be intuitively appealing and does not involve the overhead of maintaining search related information, as in the previous case. A brief survey of the existing literature on P2P community formation is presented next.

A survey on different community formation algorithms in the context of P2P has been given in [2]. Reference [3] proposed dynamic adaptation of the network topology, driven by the history of successful searches. Their objective was to maintain a list of acquaintances that connect peers that share similar interests along with some random links. Content-oriented and traffic-oriented restructuring of an initial random graph topology has been studied with an objective to create a community in [1]. A mathematical model has been setup to evaluate the community structure property of a network in [6]. It has also designed a heuristic backtrack greedy algorithm to optimize the evaluation metric of a given overlay network. A novel protocol has been proposed in [4] which changes the neighborhood of the peers based upon their proximity with the queried item in a grid topology.

In general, it is found that there is no study on the effect of community formation in power-law topologies even though it has been used as a representative topology for modeling P2P networks [6]. This work takes inspiration from our previous work in [4]. However, the algorithms are entirely redefined to suite the realistic power law topology.

The main objective of this paper is to propose an algorithm, which identifies inherent community edges among peers and evolves the topology accordingly. New edges are added among similar nodes keeping the original overlay edges intact. The topology evolution algorithm ensures bounded increase in the node capacities (i.e. average number of neighbors that a node can maintain). We also propose an evaluation metric to have a quantitative measure of the goodness of the community structure which has evolved.

The rest of the paper is organized as follows: section II presents the proposed algorithm and the evaluation metrics. Section III discusses about the experiments performed with an analysis of the results. Finally, we conclude in section IV.

II. Search Algorithm

This section is divided into three parts. In the first part we describe the framework chosen to model the P2P environment. The second part describes our community based search algorithm and the third one presents the evaluation metrics.

A. Environment Definition

The factors which are important for simulating P2P environments are the network topology and the information content of the peers, the details of which are elaborated below.

1) Topology: The initial topology of the P2P network is considered to be power law. According to the characteristic heavy tailed nature of power law graphs, few nodes have very high degrees while a very large percentage has very low degrees. Moreover, this initial degree of each node is considered as the initial network load of each node, that is, the average number of connections it can maintain in the P2P network.

2) Profile Distribution: In a file sharing P2P network, each peer has some data that it shares with other peers in the network. The subject of that data, along with the type of search queries that a peer generates is closely linked with its likes and dislikes. Based on this idea, each peer in our network design has two *profiles* - *information profile* and *search profile*. The information profile (P_I) of the peer is formed from the data which it shares with other peers in the P2P network. The search profile (P_S) of a peer is built from the informational interest of the user. For our purpose of simulating the P2P system, we represent the profiles as 8 bit binary values; each of the unique 256 profiles represents a coarse-grained interest of a peer.

The information profile is distributed in the power-law network based upon a Zipf's distribution, that is, some profiles are excessively popular in the network, but most others are not very common. The search profile is a marginally mutated bit pattern of the information profile, which in inline with the philosophy that a peer's interest is similar to its information content.

3) Query Matching: A query in this system is defined as a 8-bit binary value. Similarity of a query M and a peer's profile P is defined as $Sim(M, P) = d - HD(P, M)$ where d is the number of bits in the profile ($d = 8$ in our case), HD is the *Hamming Distance* between the profiles P and M.

B. The Algorithm

The community based search algorithm consists of two parts, the dynamics of packet movement through the network and the topology evolution initiated as a result of search.

1) Packet Movement: The search in the P2P network is initiated at, lets say, a node (peer) U. It sends search query message M to its neighbors, carrying the search profile (P_S) of U as the query to be searched. This message packet undergoes random walk in the network, until it comes across a node whose profile matches closely with the query profile. In this case, the query packet undergoes proliferation, with the aim of finding more nodes with similar profiles in the neighborhood. Some of the proliferated packets also get mutated, in order to expand the horizon of the search to similar items.

The idea of proliferation on match is to initiate an intensified search in the neighborhood. Pulling together of nodes having similar profiles would enable the proliferated queries to find nodes with similar profiles faster, thus increasing the efficiency of search.

2) Topology Evolution: Whenever there is a match with the search query, the matched node should come closer to the node that has initiated the search, in order to increase the efficiency of future searches. To achieve it, we want to connect the matched node to another node that is closer to the initiator node, to reduce its distance from the initiator. We develop an approximate solution to the problem of node selection.[1] It is schematically illustrated through Figure 1. We explain it next for clarity.

EDGE ADDITION: When a search is initiated by a node U, the query packet M makes a random walk through the network storing the complete path it has followed. If a node A matches with the query, then node A can look up the path

[1]Accurate selection of node for distance reduction requires each node to maintain the shortest distance matrix from itself to all other nodes in the network, so that it can select a node that is closer to the initiator node. The high network overhead involved in dynamically maintaining the inter-nodal distances of all nodes urged us to develop an approximate algorithm for node selection.

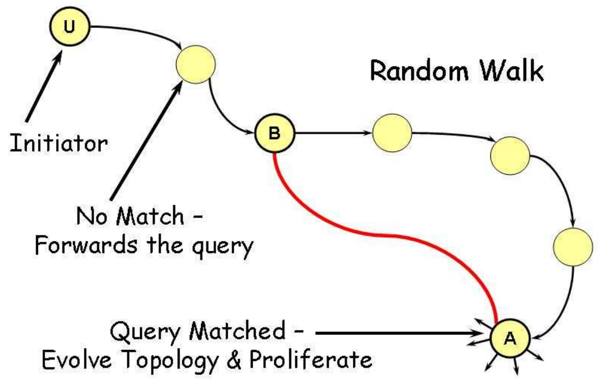

Fig. 1. Random walk based search and topology evolution

it has followed and based on various parameters, it can select an intermediate node B in the path. A community link AB is added to reduce the distance between the similar nodes U and A.

The factors that determine the selection of the intermediate node B are as follows.

- `Age` - It is the number of times a community edge has been added to or deleted from a node. If the age of node A is more, then the node B will be selected closer to the initiator node U.
- `Search count` - It is the number of times a node has initiated a search. A node that has a higher search count is expected to have a better community structure formed around it. Therefore, if the initiator node U has a higher search count than that of node A, then B is selected closer to U to bring A into the community of U.

EDGE DELETION: The above scheme results in the addition of edges; however an unbounded addition of community edges in the network is not desirable. Hence, for every edge AB that is added, we probabilistically delete an edge from A and B with probabilities p_A and p_B respectively such that $p_A + p_B = 1$. Simultaneously, in order to limit the increase in the network load of each node, the values of p_A and p_B are calculated based on the increase in degree over the initial degree. Let us define the relative increase in degree X_n of a node n as

$$X_n = \frac{Deg_n - InitDeg_n}{InitDeg_n} \tag{1}$$

where $InitDeg_n$ = initial degree of node n and Deg_n = current degree of n. The corresponding values of the nodes A and B are X_A and X_B, respectively. The probabilities of deletion of a link in node A is made proportional to its relative increase in degree. Hence the probabilities are calculated as follows.

$$p_A = \frac{X_A}{X_A + X_B}, \; p_B = \frac{X_B}{X_A + X_B} \tag{2}$$

In this scheme, only the newly added community edges are deleted. The initial network is maintained as an overlay network to prevent the break down of the network. This leads to situations where we do not find any edges to delete. As a result, a net increase in the edge count of the network is observed during our experiments.

C. Evaluation Metrics

In order to measure the impact of the rewiring on the network, we have designed a few metrics. These metrics evaluate the performance of the search algorithm as well as the goodness of community structure of the resultant graph. They are as follows.

1) Community Structure: The aim of our algorithm is to essentially bring the nodes having similar information profiles closer to each other in the network. In other words, given a search profile P, we want to bring all nodes with information profile P closer to each other, so that finding one of them enables us to efficiently find all of them. Here, similarity between nodes is measured by the Hamming Distance (HD) of their information profiles. Let this set of nodes be S_P. In order to measure their inter-nodal distance, we devise a metric that calculates the weighted average distance $\delta_{u,v}^{'}$ between the nodes in the network (say u and v), weighted by their degree of similarity with the base profile P (i.e. $HD(P_u, P)$ and $HD(P_v, P)$). This is calculated as follows.

$$\delta_{u,v}^{'} = \delta_{u,v}[1 + \frac{HD(P, P_u) + HD(P, P_v)}{2d}] \tag{3}$$

where d = number of bits in the profile. This weighted distance is averaged over all the pairs of nodes in the S_P as follows.

$$\mathcal{D}_P = \frac{1}{|S_P|^2} \sum_{\forall u,v \varepsilon S_P} \delta_{u,v}^{'} \tag{4}$$

Now, this must be calculated for all possible search profiles P. Hence, the final metric is calculated by adding the $\mathcal{D}_P$ for all the 2^d profiles. In order to give more weightage to the more popular profiles, the $\mathcal{D}_P$ is weighted by the relative frequency of the profile P. Hence the final metric is calculated as

$$\mathcal{D} = \sum_{\forall P} \mathcal{D}_P \frac{N(P)}{N} \tag{5}$$

where N = number of nodes in the network and $N(P)$ = number of nodes having profile P. A decrease in the value of $\mathcal{D}$ reflects the fact that the similar nodes in the network have moved closer to each other.

2) Search Efficiency: To measure the efficiency of search on networks, the following metric is used.

- Total search hit count - Total number of nodes matched in a single search attempt till TTL expires.
- Cumulative search hit count - The total number of search hits upto t TTL.

3) Topology Overhead: The addition of edges by the algorithm to the nodes causes them to maintain more number of connections. This network overhead, incurred to make the search more efficient, needs to be quantified. For a node n, we define its degree as its network load. We have already defined X_n in section II(B). This is the relative increase of a node's degree beyond its initial degree. Hence, we define the same as the relative increase in the network load of node n from its initial load. The average increase in the network load of all nodes is calculated as

$$X_{avg} = \frac{1}{N} \sum_{\forall n} X_n \tag{6}$$

III. Experiments

In order to test the efficiency of the algorithm, we decided to test it in a simulated network. For this purpose, we designed a discrete-time state-based simulator, which simulates the algorithm repeatedly on a given network and finally outputs the average statistics. The details are provided below.

A. Simulation Model

In our simulation framework, a single search query is propagated up to SearchTTL hops. A set of search queries executed on random nodes constitute a generation. All performance metrics are averaged over a generation. A number of generations performed on the same network constitute a simulation. Multiple simulations are performed on different networks for averaging the performance of the algorithm.

The details of simulation of the algorithm are given below. In all the simulations we have executed the searches for 15 hops/TTL, 100 searches per generation and 10 generations of search per simulation. 100 such simulations were performed for averaging the results.

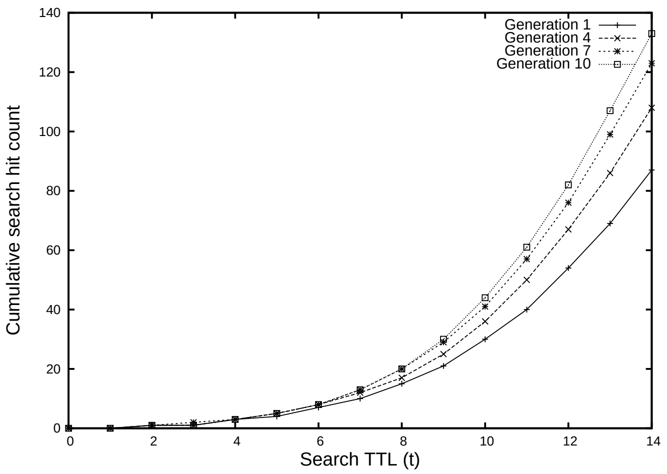

Fig. 2. Trend of search hit count in different generations of search. X-axis is the search TTL and Y-axis is the average total search hits accumulated in x TTL, i.e. cumulative search hits.

B. Simulation

The algorithm is run on a power law network of 1000 nodes and 3984 edges, thus having an average degree of 7.968. Gamma of the degree distribution is 1.95.

Averaging the results over all the simulations, we find that on average 2102 edges were added to the network, producing a 52.76% increase in the number of edges. The resultant increase in network load X_{avg} was 60.11%. Figure 2 shows the trend of cumulative search hits over search TTL in different generations of search. As is evident, search in generation 10 picks up much faster than search in lower generations (say 1). Also, figure 4 shows the total search hit count in each generation. We see that there is continuous increase of around 7% per generation in the search results. Finally, in Figure 3, we find the value of the metric proposed in section II(C) decreases over generations, signifying that the similar nodes in the network have moved closer to each other. This confirms that the underlying community structure has successfully formed with time.

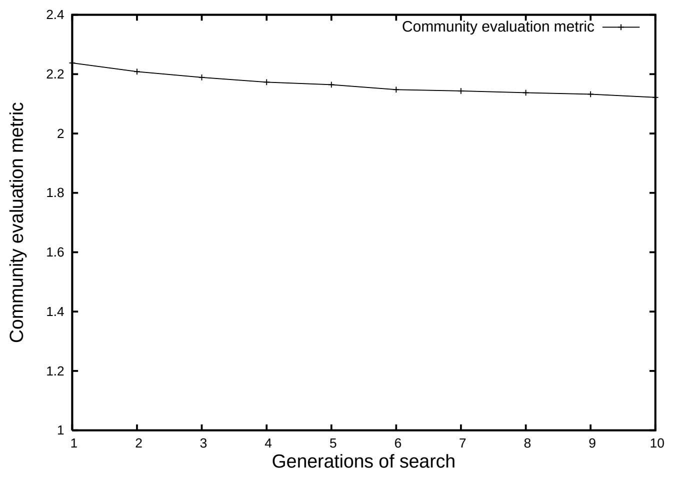

Fig. 3. Variation of the community metric with generations of search. The X-axis is the generations of search, and Y-axis is the value of the community metric after x generations.

C. Fairness Test

As observed in the simulation, an overhead of increased average degree is incurred by the network to improve the search efficiency. The obvious question that arises is whether our search efficiency by community edge addition produces better results compared to random edge addition in an equivalent graph. In other words, if we are to let the system incur the cost of increased degree, then does community edge addition by our algorithm produce better results than random edge addition.

In order to answer this question, the same search algorithm without any topology evolution is executed on the same initial power law network of 1000 nodes and 3984 edges. As more edges get added to the actual network, equivalent number of edges are added randomly to the equivalent network. Also, the proliferation rate is so maintained such that the average number of packets generated while searching remains same in both the networks.

The final results are shown in figure 4. As we can see, keeping all other factors like network size and packets generated constant, we find that the total hit count (averaged over multiple simulations) is more in case of community edge addition. After 10 generations of search, the community edge addition produces 12.7% more results per search than random edge addition. This shows that the community formation definitely produces more optimized graph structure, with respect to random search, compared to random edge addition to the initial network.

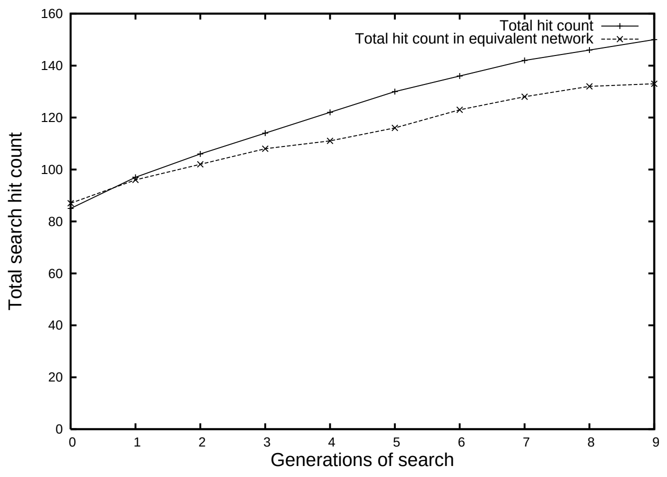

Fig. 4. Total search hit count per search with respect to generations of search. The X-axis is the generations of search, and the Y-axis gives the average total search hit generated in the x^{th} generation.

IV. Conclusion and Future Work

We have presented the performance of our proposed search algorithm and have demonstrated that it successfully improves the performance of random walk based search algorithms. Also, we have proposed a new metric to evaluate search related community structures in P2P networks. Our future work include the development of a mathematical framework for theoretical evaluation of the performance and also analyzing the performance of this algorithm in other network topologies like random graphs and small world networks.

References

[1] German Sakaryan and Herwig Unger, *Influence of the decentralized algorithms on topology evolution in P2P distributed networks*, in Design, Analysis, and Simulation of Distributed Systems (DASD), pages 12-18. Orlando, USA, 2003.

[2] Anna Puig-Centelles, Oscar Ripolles and Miguel Chover Centelles, *Reviewing P2P network coummunity detection*, in the IASTED European Conference on Internet and Multimedia Systems and Applications (EuroIMSA). Chamonix, France, 2007.

[3] V. Cholvi, P. A. Felber, and E. W. Biersack, *Efficient search in unstructured peer-to-peer networks*, in Proceedings of the Sixteenth Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA). Barcelona, Spain, 2004.

[4] Niloy Ganguly, Geoff Canright and Andreas Deutsch, *Design Of An Efficient Search Algorithm For P2P Networks Using Concepts From Natural Immune Systems*, in the 8th International Conference on Parallel Problem Solving from Nature (PPSN VIII). Birmingham, UK, 2004.

[5] Hanhua Chen and Hai Jin, *Identifying Community Structure in Semantic Peer-to-Peer Networks*, in Proceedings of the 2nd International Conference on Selmantic, Knowledge and Grid (SKG). Guilin, China, 2006.

[6] Farnoush Banaei-Kashaniy and Cyrus Shahab, *Criticality-based Analysis and Design of Unstructured Peer-to-Peer Networks as Complex Systems*, in Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid). Tokyo, Japan, 2003.