A Trust-Based Robust and Efficient Searching Scheme for Peer-to-Peer Networks

Jaydip Sen

Innovation Lab, Tata Consultancy Services Ltd.,
Bengal Intelligent Park, Salt Lake Electronics Complex, Kolkata – 700091, India
Jaydip.Sen@tcs.com

Abstract. Studies on the large scale peer-to-peer (P2P) network like Gnutella have shown the presence of large number of free riders. Moreover, the open and decentralized nature of P2P network is exploited by malicious users who distribute unauthentic or harmful contents. Despite the existence of a number of trust management schemes for combating against free riding and distribution of malicious files, these mechanisms are not scalable due to their high computational, communication and storage overhead. Moreover they do not consider the quality-of-service (QoS) of the search. This paper presents a trust management scheme for P2P networks that utilizes topology adaptation to minimize distribution of spurious files. It also reduces search time since most of the queries are resolved within the community of trustworthy peers. Simulation results demonstrate that the proposed scheme provides efficient searching to good peers while penalizing the malicious peers by increasing their search times as the network topology stabilizes. The mechanism is also found to be robust even in presence of a large percentage of malicious peers.

Keywords: P2P network, topology adaptation, trust management, semantic community, malicious peer, iterative DFS.

1 Introduction

The term *peer-to-peer* (P2P) system encompasses a broad set of distributed applications which allow sharing of computer resources by direct exchange between systems. The goal of a P2P system is to aggregate resources available at the edge of Internet and to share it cooperatively among users. Specially, the file sharing P2P systems have become popular as a new paradigm for information exchange among large number of users in Internet. They are more robust, scalable, fault tolerant and offer better availability of resources than traditional client-server systems.

Depending on the presence or absence of a central server, a P2P system can be either a fully distributed or a partially distributed system respectively [1]. In the fully distributed architecture, both resource discovery and download activities are distributed. A partially distributed P2P system may be further classified as a *structured* or an *unstructured* network. In a structured network, there are certain restrictions on the placement of the contents and the network topology. For example, CHORD [2] has a ring topology and uses *distributed hash tables* for locating resources in a P2P system.

M. Soriano, S. Qing, and J. López (Eds.): ICICS 2010, LNCS 6476, pp. 77–91, 2010.

In unstructured P2P networks, however, placement of contents is unrelated to the topologies of the networks. Unstructured P2P networks perform better than their structured counterparts in dynamic environment. However, they need efficient search mechanisms to locate resources and also suffer from several problems such as: fake content distribution, free riding (peers who do not share, but consume resources), whitewashing (peers who leave and rejoin the system in order to avoid penalties) etc. In fact, the three main problems currently existing in unstructured P2P networks are: (i) fake content distribution, (ii) lack of scalability in searching techniques, and (iii) free riding. In the rest of this section, these issues are discussed briefly.

Fake content distribution- Open and anonymous natures of P2P applications lead to complete lack of accountability of the content a peer provides in the network. Malicious peers use these networks to do content poisoning and to distribute harmful programs such as Trojan Horses and viruses [3]. *Distributed reputation based trust management systems* have been proposed to provide protection against malicious content distribution. These schemes are of two types: *gossip-based* and *topology adaptation-based*. In gossip-based schemes, a peer evaluates trustworthiness of the resource provider from past transaction and recommendation from its neighbors [4]. The main drawbacks of these schemes are their high message exchange overheads and their susceptibility to misrepresentation. In topology-adaptation schemes, a peer connects to those peers who provide authentic files. Guo et al. have proposed a trust-aware adaptive P2P topology to control free-riders and malicious peers [5]. In [6] and [7] topology adaptation is used to reduce inauthentic file download and banish free riders.

Search scalability - The second major problem with P2P system is poor search scalability. Usually controlled flooding, random walker or more recently topology evolution are used to locate resources in unstructured networks. Besides, topology adaptation is also used for efficient search in P2P network by forming *semantic communities*. In semantic communities, file requests have high probability of being satisfied within the community they originate from [8]. This increases search efficiency.

Free Riding - Though in ideal P2P systems the role of each peer is the same (as the producer as well as the consumer of resources), in reality the peers are heterogeneous entities with varying capacities with most of them trying to maximize their individual gain from the system rather than trying to fulfill the system goal. Experiments conducted on Gnutella - a popular content sharing P2P network, reveal the presence of significant numbers of free riders. It has been observed that about 70% of the users do not share any files, and only 1% of the peers provide answers to 50% of the queries in the network [9]. Various trust-based incentive mechanisms are presented in [10] to encourage sharing of large number of authentic files. Zhuge et al. [8] have proposed a trust-based probabilistic search algorithm to improve search efficiency and to reduce unnecessary traffic in P2P networks.

In this paper, an integrated approach towards solving the above problems in unstructured P2P networks is proposed. The rest of the paper is organized as follows. Section 2 discusses the motivation and objective of the work. Section 3 discusses some related work existing in the literature. Section 4 presents the proposed algorithm for trust management. Section 5 introduces various metrics to measure the performance of the proposed algorithm, and also presents the simulation results. Finally, Section 6 concludes the paper while highlighting some future scope of work.

2 Motivation and Objectives

Although topology adaptation can be used to combat inauthentic downloading as well as to improve search scalability, no work has been carried out to use topology adaptation to achieve both goals simultaneously. In this paper, an *adaptive trust-aware algorithm* is proposed that is robust, scalable and lightweight. It uses a trust management algorithm via topology adaptation for unstructured network to minimize inauthentic download and to discourage free riding while optimizing the QoS for search. The proposed scheme constructs an overlay of trusted peers where neighbors are selected based on trust rating and content similarity. It increases search efficiency by taking advantage of implicit semantic community structures formed as a result of topology adaptation since most of the queries can be resolved within the community. The search strategy is self-adjusting, which adapts as community grows based upon local information only to provide best performance (in terms of increasing query hits and reduced message overhead). It provides incentives to share high quality files and punishes malicious peers who provide fake files while discouraging free riding.

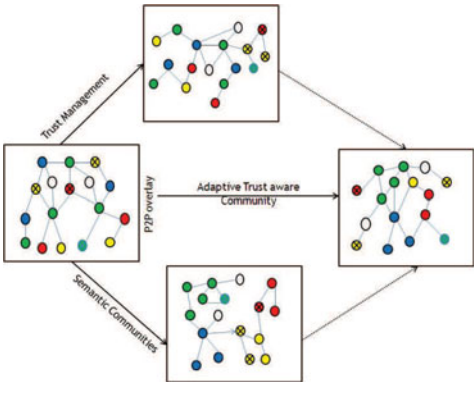

Fig. 1. The effect of trust management and semantic community in P2P overlay. Peers sharing same contents are shown with crosses while free riders are shown in white color.

Figure 1 shows the effect of trust management and semantic community on a P2P overlay. The peers sharing a particular content type are marked with same color. Within each content category, there are malicious peers who provide fake files. These nodes are shown with crosses. An ideal trust management scheme segregates good peers from the malicious ones as shown in Figure 1. However, it does not bring the peers sharing similar contents on the neighborhood of each other to form *interest-based communities*. Semantic communities adapt topology to form cluster of peers sharing similar contents, but they ignore trustworthiness of peers and hence do not punish malicious nodes. The proposed mechanism in this paper does both the functions. It adapts the topology to bring good peers with matching interest to the vicinity of each other as illustrated with dotted lines in Figure 1. The peers who provide good community service are rewarded and offered topologically better positions. In addition, the malicious peers and free riders are banished to the fringe of the network.

3 Related Work

Several propositions exist in the literature for efficient and secure searching in P2P networks. In [11], a searching mechanism is proposed that is based on discovery of *trust paths* among the peers in a P2P network. The authors have argued that since the trust path discovery does not allow resource replication, it is very sensitive to parameter choices in selective forwarding algorithms. A global trust model based on *distance-weighted recommendations* has been proposed in [12] to quantify and evaluate the peers in a P2P network.

The work on the proposed mechanism in this paper is motivated from the trust management techniques in [6] and [7]. In [6], a protocol named APT for the formation of adaptive topologies has been proposed to reduce inauthentic file downloads and free ridings, where a peer connects to those peers from whom it is most likely to download satisfactory content. It adds or removes neighbors based on *local trust* and *connection trust* which are decided by its transaction history. The scheme follows a defensive strategy for punishment where a peer equally punishes both malicious peers as well as neighbors through which it receives response from malicious peers. This strategy is relaxed in RCATP, where a peer connects to those peers having higher *reciprocal capacity* [7]. Reciprocal capacity is defined based on peers's capacity of providing good files and of recommending source of download. In addition, a response selection mechanism is proposed to reduce the probability of download from malicious peers. Due to adequate number of connections between reciprocal peers, connectivity is better in RCAPT than APT. It also reduces cost incurred by unsatisfactory download. However, overhead of topology adaptation is very high in RCAPT. The proposed scheme differs significantly from APT and RCAPT as discussed below.

First, the proposed scheme never deletes links in the original overlays to avoid network being fragmented; only edges added by topology adaptation may be deleted. However, unlike the scheme proposed in this paper, the other two approaches do not quantitatively measure network connectivity. Second, the authors of APT and RCAPT have not evaluated robustness of their algorithms in presence of increasing percentage of malicious peers in the network. In contrast, the robustness of the proposed scheme has been extensively studied in presence of malicious nodes. Third, as APT and RCAPT both use flooding to locate resources, they have poor search scalability. The proposed scheme tunes itself to take the advantages of semantic communities to improve QoS of search. Fourth, the scheme punishes malicious peers by blocking query initiated by them. Finally, RCAPT was simulated on power law network containing 550 peers. It lacks an effective mechanism to disseminate trust information in a large network. In the proposed scheme, the *least recently used* (LRU) data structure and *depth first search* (DFS) are used to make it scalable.

4 The Proposed Trust-Aware Algorithm

This section is divided into two parts. In the first part, the various parameters for simulating a P2P network are discussed. In the second part, the proposed algorithm is presented in detail.

4.1 Environment Definition

To derive meaningful conclusion from the proposed algorithm, the proposed scheme have been modeled in P2P networks in a realistic fashion. The factors that are taken into consideration are as follows.

(1) *Network topology and load*: The topology of a network plays an important role for the analysis of trust management and search procedure. Following the work in [6][7], the network has been modeled as a *power law graph*. In a power law network, degree distribution of nodes follows *power law distribution*, i.e. fraction of nodes having degree L is L^{-k} where k is a network dependent constant. Prior to each simulation cycle a fixed fraction of peers chosen randomly is marked as malicious. As the algorithm proceeds, the peers adjust topology locally to connect those peers which have better chance to provide good files in future and drop malicious peers from their neighborhood. The network links are categorized into two types: *connectivity link* and *community link*. The connectivity links are the edges of the original power law network which provide seamless connectivity among the peers. To prevent the network from being fragmented they are never deleted. On the other hand, community links are added probabilistically between the peers who know each other. A community link may be deleted when perceived trustworthiness of a peer falls in the perception of its neighbors. A limit is put on the additional number of edges that a node can acquire to control bandwidth usage and query processing overhead in the network. This increase in network load is measured relative to the initial network degree (corresponding to connectivity edges). Let *final_degree(x)* and *initial_degree(x)* be the initial and final degree of a node x. The *relative increase in connectivity* (RIC) is constrained by a parameter known as *edge_limit*.

$$RIC = \frac{final\_degree(x)}{initial\_degree(x)} \leq edge\_limit \tag{1}$$

(2) *Content distribution*: The dynamics of a P2P network are highly dependent on the volume and variety of files each peer chooses to share. Hence a model reflecting real-world P2P networks is required. It has been observed that peers are in general interested in a subset of the content on the P2P network [12]. Also, the peers are often interested only in files from a few content categories. Among these categories some are more popular than others. It has been shown that Gnutella content distribution follows *zipf distribution* [13]. Keeping this in mind, both content categories and file popularity within each category is modeled with *zipf distribution* with $\alpha = 0.8$.

Content distribution model: The content distribution model in [13] is followed for simulation purpose. In this model, each distinct file $f_{c,r}$ is abstractly represented by the tuple (c, r), where c represents the content category to which the file belongs, and r represents its popularity rank within a content category c. Let content categories be $C = \{c_1, c_2,\ldots,c_{32}\}$. Each content category is characterized by its *popularity rank*. If the ranks of $c_1 = 1$, $c_2 = 2$, and $c_3 = 3$, c_1 is more popular than c_2 and hence replicated more than c_2 and so on. Also there are more files in category c_1 than c_2.

Each peer randomly chooses between 3 to 6 content categories to share files and shares more number of files in more popular categories. Table 1 shows a fictitious content distribution for illustration purpose. The category c_1 is more replicated as it is

most popular. The *Peer 1* shares files in three categories: c_1, c_2, c_3 where it shares maximum number of files in category c_1, followed by category c_2 and so on. On the other hand, *Peer 3* shares maximum number of files in category c_2 as it is the most popular among the categories chosen by it, followed by c_4 and so on.

Table 1. Hypothetical content distribution in peer nodes

Peers	Content categories
P_1	$\{C_1, C_2, C_3\}$
P_2	$\{C_2, C_4, C_6, C_7\}$
P_3	$\{C_2, C_4, C_7, C_8\}$
P_4	$\{C_1, C_2\}$
P_5	$\{C_1, C_5, C_6\}$

(3) *Query initiation model*: The authors in [13] suggest that peers usually query for files that exist on the network and are in the content category of their interest. In each cycle of simulation, active peers issue queries. However number of queries a peer issues may vary from peer to peer, modeled by *Poisson* distribution as follows. If M is the total number of queries to be issued in each cycle of simulation and N is the number of peers present in the network, query rate $\lambda = M/N$ is the mean of the *Poisson* process. The expression $p(\#quries = K) = \frac{e^{-\lambda}\lambda^K}{K!}$ gives the probability that a peer issues K queries in a cycle. The probability that a peer issues query for the file $f_{c,r}$ depends on the peer's interest level in category c and rank r of the file within that category.

(4) *Trust management engine*: A trust management engine is designed which helps a peer to compute trust rating of other peer from past transaction history as well as recommendation from its neighbor. It allows a peer to join the network with default trust level and gradually build its reputation by providing good files to other peers. In the proposed scheme, each peer maintains a *least recently used* (LRU) data structure to keep track of recent transactions with almost 32 peers at a time. Each time peer i downloads a file from peer j, it rates the transaction as positive (tr_{ij}=1) or negative (tr_{ij}=-1) depending on whether downloaded file is authentic or fake. $S_{ij} = \frac{1}{TD}\sum tr_{ij}$ is the fraction of successful downloads peer i had made from peer j, where TD is the total number of downloads. Peer i considers peer j as trustworthy if $S_{ij} < 0.5$, and malicious if $S_{ij} < 0$. If $0 \le S_{ij} < 0.5$, peer i considers peer j is average trustworthy. Peer i may seek recommendations from other peers about peer j only when information is not locally available. It is the trust management engine which makes the proposed scheme robust and light-weight.

(5) *Node churning model*: P2P networks are transient in nature. A large number of peers join and leave the network at any time. This activity is termed as node *churning*. To simulate node churning, prior to each *generation* (a set of consecutive searches), a fixed percentage of nodes are chosen randomly as inactive. These peers neither

initiate nor respond to a query in that generation and join the system latter with their LRU structure cleared. Since in a real world network, even in presence of churning, the approximate distribution of content categories and files remain constant, content of nodes undergoing churn is exchanged which in effect assigns each of them new content as well as keeps content distribution model of the network unchanged.

(6) *Threat model*: Malicious peers adopt various strategies (threat model) to conceal their behavior and disrupt system activity. Two threat models are considered in the proposed scheme. The models are denoted as *threat model A* and *threat model B*. The peers who share good quality files enjoy better topological position due to topology adaptation. In *threat model A*, malicious peers attempt to circumvent this by providing good file occasionally with probability, known as *degree of deception* to lure other peers to form communities with them. In *threat model B*, a group of malicious peer joins to the system and provides good files until their connectivity reaches to *edge limit*, and then start spreading fake content in the network.

4.2 The Proposed Trust-Aware Algorithm

The network learns trust information through the search and updates trust information and adapts topology based on the outcome of the search. The following criteria are kept in mind while designing the algorithm: (1) It should improve search efficiency as well as search quality (authentic file download). (2) It should have minimal overhead in terms of computation, storage and message passing. (3) It should provide incentive to share large number of high quality files. (4) It should be self policing in the sense that a peer can adjust search strategy based on local estimate of network connectivity. The major steps of the algorithm are: (i) search, (ii) trust checking and (iii) topology adaptation. Each of these steps is discussed in the rest of this section.

4.2.1 Search

The *time to live* (TTL) bound search is used which evolves along with topology. At each hop, query is forwarded to a fraction of neighbors, the number of neighbors is decided based on the local estimate of network connectivity, known as probability of community formation, $Prob_{com}$, computed at the query initiating node, using relative increase in degree. It is defined at node x as follows:

$$Prob_{com} = \frac{degree(x) - initial\_degree(x)}{initial\_degree(x).(edge\_limit - 1)} \tag{2}$$

When $Prob_{com}$ is low, a peer has the capacity to accept new community edges and expand community structures. Higher the value of $Prob_{com}$, lesser the neighbors choose to disseminate queries. As the simulation proceeds, connectivity of good nodes increases and reaches a saturation level. So, they focus on directing queries to appropriate community which may host the specific file rather than expanding communities. For example, if peer i can contact at most 10 neighbors and $Prob_{com}$ of i is 0.6, it forwards query to 10 * (1 – 0.6) = 4 neighbors only. The search strategy modifies itself from initial *TTL limited BFS* to *directed DFS* with the restructuring of the network. The search is carried out in two steps– *query initiation* and *query forward*.

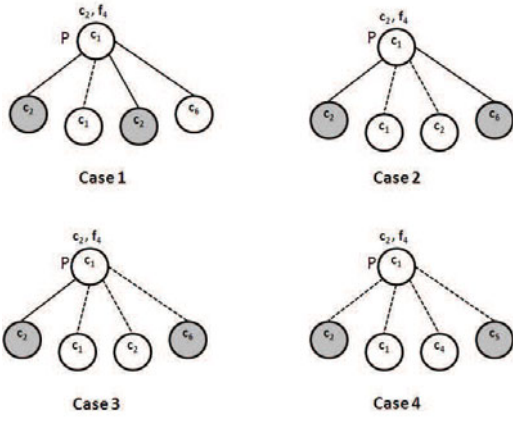

Fig. 2. Neighbor selection at P for query string (c_2, f_4). Community edges and connectivity edges are drawn with solid and dotted lines respectively. Nodes that dispatch query are shaded.

Query initiation: Initiating peer forms a query packet containing the name of the file (*c, r*) and forwards it to a fixed fraction of neighbors along with $Prob_{com}$ and TTL value. The query is disseminated using the following *neighbor selection rule*. The neighbors are ranked based on both trustworthiness and the similarity of interest. Preference is given to the trusted neighbors sharing similar contents. Among the trusted neighbors, community members having content matched to the query are preferred. When there is insufficient number of community links, query is forwarded through connectivity links also. The various cases of neighbor selection are illustrated in Figure 2. It is assumed that in each case only two neighbors are selected. When the query (c_2, f_4) reaches node P, following cases may occur. In *Case 1*, P has adequate number of community neighbors sharing file in category c_2, hence they are chosen. In *Case 2*, there is an insufficient number of community neighbors sharing file in the requested category, the community neighbors sharing c_2 and c_6 preferred to the connectivity neighbor c_2 to forward query. In *Case 3*, the only community neighbor who shares file is c_2. Hence c_2 is chosen in this case. From the remaining connectivity neighbors, the most trusted one - c_6 is selected. In *Case 4*, only connectivity neighbors are present. Assuming all of the connectivity neighbors are at the same trust level, the matching neighbor c_2 is chosen and from the rest c_5 is selected randomly.

When a query reaches peer i from peer j, following actions are performed by peer i. These actions constitute the *query forward* step.

Query forward: (i) *Check trust level of peer j*: Peer i checks trust rating of peer j through *check trust rating* algorithm (explained later). Accordingly decision regarding further propagation of the query is taken. (ii) *Check the availability of file*: If the requested file is found, response is sent to peer j. If TTL value has not expired, the following steps are executed. (iii) *Calculate the number of messages to be sent*: It is calculated based on the value of $Prob_{com}$.(iv) *Choose neighbors*: Neighbors are chosen in using *neighbor selection rule*. The search process is shown in Figure 3. It is assumed that the query is forwarded at each hop to two neighbors. The matching community links are preferred over connectivity links to dispatch query. Peer *1* initiates

query and forwards it to two community neighbors *3* and *4*. The query reaches peer *8* via peer *4*. However, peer *8* knows that peer *4* is malicious from previous transactions. Hence it blocks the query. The query forwarded by peer *5* is also blocked by peer *10* and *11* as both of them know that peer *5* is malicious. The query is matched at four peers: *4*, *6*, *9* and *13*.

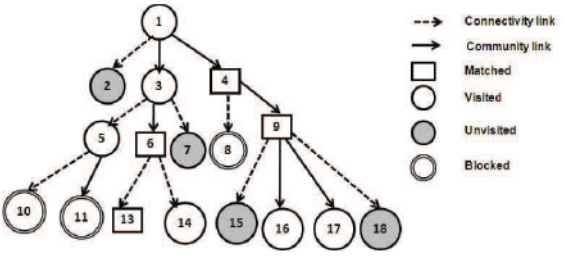

Fig. 3. The breadth first search (BFS) tree for the search procedure initiated by peer 1

Topology Adaptation: Responses are sorted by the initiating peer *i* based on the reputation of resource providers and peer having highest reputation is selected as source of download. The requesting peer checks the authenticity of downloaded file. If the file is found to be fake, peer *i* attempts to download from other sources until it finds the authentic resource or no more sources exist and updates the trust rating and possibly adapts topology after failed or successful download, to bring trusted peers to its neighborhood and to drop malicious peers from its community. The restructuring of network is controlled by a parameter known as *degree of rewiring* which is the probability with which a link is formed between two peers. This parameter allows trust information slowly to be propagated as happens in real network. Topology adaptation consists of the following operations: (i) *link deletion*: Peer *i* deletes the existing community link with peer *j* if it finds peer *j* as malicious. (ii) *link addition*: Peer *i* probabilistically forms community link with peer *j* if resource is found to be authentic. If $RIC \leq edge\_limit$, for both peers *i* and *j*, only then an edge can be added subject to the approval of resource provider *j*. If peer *j* finds that peer *i* is malicious, it doesn't approve the link. In the example shown in Figure 4, peer 1 downloads the file from peer 4 and finds that the file is spurious. It reduces the trust score of peer 4 and deletes the community link 1-4. It then downloads the file from peer 6 and gets an authentic file. Peer 1 now sends a request to peer 6, and the latter grants the request after consulting its LRU and the community edge 1-6 is added. The malicious peer 4 loses one community link and peer 6 gains one community edge. However, the network still remains connected by connectivity edges, shown in dotted lines.

Check trust rating: Trust rating is used at various stage of the algorithm to make decision about possible download source, to stop a query forwarded from a malicious node and to adapt topology. A *least recently used* (LRU) data structure is used at each peer to keep track of *32* most recent peers it has interacted with. When no transaction history is available, a peer seeks recommendation from its neighbors using *trust query*. When peer *i* doesn't have trust score of peer *j* in its LRU history, it first seeks recommendation about *j* from all of its community neighbors. If none of its community neighbors possesses any information about *j*, peer *i* initiates directed DFS search.

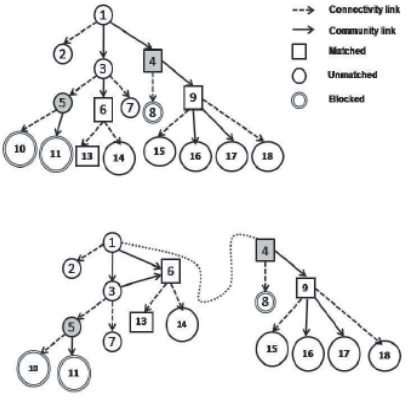

Fig. 4. Topology adaptation based on outcome of the search in Figure 3. Malicious nodes are shaded in gray color.

5 Performance Evaluation

To analyze the performance of the proposed algorithm, a set of metrics are defined that measures the efficiency and quality of searches. These metrics are defined below.

(1) *Attempt ratio* (AR): A peer keeps on downloading the file from various sources, based on their trust rating till it gets the authentic file. AR is the probability that in the first attempt, the authentic file is downloaded. Ideally, AR should be high.

(2) *Effective attempt ratio* (EAR): The malicious peers may also increase their search quality in order to hide their true nature. Hence, the search quality achieved by good peers should be measured relative to that provided by malicious peers. EAR measures the cost of downloading an authentic file by good peers relative to malicious peers. EAR is measured against various percentage of malicious peers and the percentage of malicious peers for which EAR drops to zero is noted. If *P(i)* be the total number of attempts made by peer *i* to download an authentic file, EAR is given by (3):

$$EAR = \left(\frac{1}{M}\sum_{i=1}^{M}\frac{1}{P(i)} - \frac{1}{N}\sum_{j=1}^{N}\frac{1}{P(j)}\right)*100 \tag{3}$$

M and *N* are the number of good and malicious peers issuing queries in a particular generation. For example, EAR= 50 implies that if a good peer, on the average, needs one attempt to download an authentic file, a malicious peer needs two attempts.

(3) *Query miss ratio* (QMR): Since there is no semantic community initially, and a low value of TTL is used for file searching, there will be a high rate of query misses in the first few generations of search. However, as the algorithm executes, the query miss is expected to fall down for good peers. For malicious peers the rate of decrease of query miss will be very slow since queries from malicious peers are blocked. QMR is defined as the ratio of the number of search failures to the total number of searches in a generation.

(4) *Hit per message* (HM): Due to the formation of semantic community, number of messages required to get a hit is expected to fall down. HM measures the search efficiency achieved by the proposed algorithm, and is defined as the number of query hit per message irrespective of the authenticity of the file being downloaded.

(5) *Relative increase in connectivity* (RIC): After a successful download, a requesting peer attempts to connect to the resource provider by forming a community edge if approved by the resource provider. This ensures that peers providing good community services are rewarded by having increasing number of community neighbors. The metric RIC measures the number of community neighbors a peer gains relative to its connectivity neighbors in the initial network topology. If $D_{init}(i)$ and $D_{final}(i)$ are the initial and final degrees of the peer i, and N is the number of peers, then RIC for peer i may be computed using (4). Due to the incentive scheme in the proposed mechanism, the connectivity of good peers is expected to increase significantly with time.

$$RIC = \frac{1}{N}\sum_{i}\frac{D_{final}(i)}{D_{init}(i)} \tag{4}$$

A discrete time simulator written in C is used for simulation. In the simulation, 6000 peer nodes, 18000 connectivity edges, 32 content categories are chosen. The *degree of deception* and the *degree of rewiring* are taken as 0.1 and 0.3 respectively. The value of the *edge_limit* is taken as 0.3. The TTL values for BFS and DFS are taken as 5 s and 10 s respectively. The discrete time simulator simulates the algorithm repeatedly on the power law network and outputs all the metrics averaged over generations. Barabasi-Alabert generator is used to generate initial power law graphs with 6000 nodes and approximately 18000 edges. The number of search per generation is taken as 5000 while the number of generations per cycle of simulation is 100.

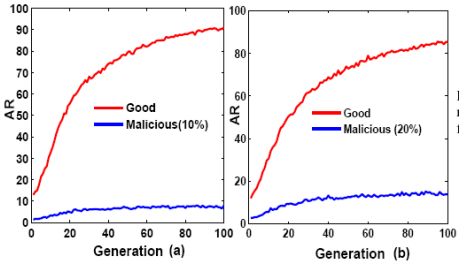

Fig. 5. AR vs. percentage of malicious nodes: 10% in (a) and 20% in (b)

To check the robustness of the algorithm against attack from malicious peers, the percentage of malicious peers is gradually increased. Figure 5 illustrates the cost incurred by each type of peers to download authentic files. As the percentage of malicious peers is increased, cost incurred by malicious peers to download authentic files decreases while that of good peers increases, This is illustrated in Figure 6 using EAR.

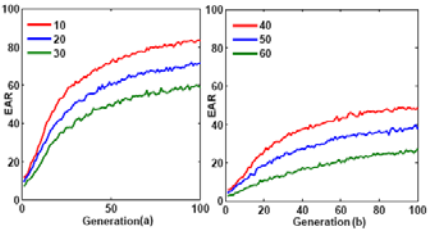

Fig. 6. EAR vs. search generation for various percentages of malicious nodes

As evident from Figure 6, with 10% malicious peers in the network, EAR is 80; i.e., on the average, if a good peer needs one attempt to download an authentic file, a malicious peer needs 5 attempts to do so. The peers who share high quality files acquire good reputation and earn more community edges and eventually disseminates query through the community edges only. As the queries are forwarded via trusted peers at each hop, the probability of getting authentic file in the first attempt increases. However, as the queries forwarded by malicious peers are blocked by good peers, they need more attempts to download good files. As the percentage of malicious peers in the network increases, EAR drops to 20. Up to 60% malicious peers, good peers have higher probability to get an authentic file in the first attempt than malicious peers. So the proposed algorithm can withstand against malicious peers till the percentage of malicious peers is below 60.

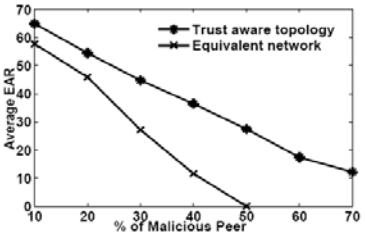

Fig. 7. Avg. EAR vs. % of malicious nodes in networks with and without trust management

The performance of the proposed protocol is compared with an equivalent power law network with no trust management. Since the proposed algorithm allows for addition of community edges, to keep the number of edges in both networks equal, additional edges are introduced between similar peers in the equivalent network. Figure 7 shows the comparison of the average EAR values. In the network without trust management, EAR drops to zero with 50% malicious nodes. However, in the network with trust management, even with 60% malicious peers, EAR is maintained at 20. This clearly demonstrates the robustness of the proposed trust management algorithm.

Figure 8 shows QMR experienced by both types of peers for varying percentages of malicious peers in the network. Initially, QMR is high as no interest-based communities are formed and the searching is essentially a blind one. As the simulation proceeds, peers with similar content interests come closer to each other and queries are forwarded through the community edges. As a result, QMR drops for good peers. It is observed from Figure 8 that steady state value of QMR for good peers is less than 0.2,

and QMR is independent of the percentage of malicious peers. This is a significant performance achievement of the proposed algorithm. For malicious peers, the steady state value of QMR is 0.4. The high QMR for malicious peers is due to the fact that the queries form the malicious peers are blocked by the good peers. Thus the proposed algorithm effectively rewards the peers who share large number of good files.

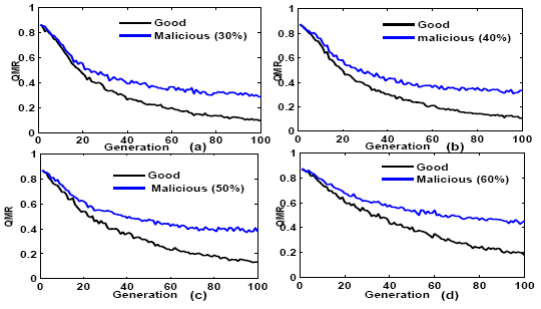

Fig. 8. QMR variations with various percentages of malicious peers in the network

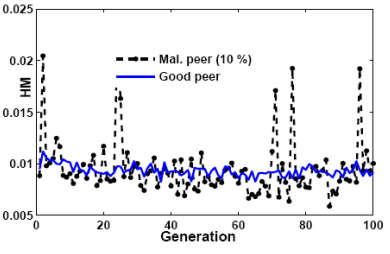

Fig. 9. HM vs. generation of search for each type of peer for 10% malicious peers

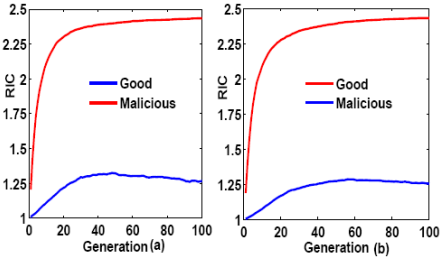

Fig. 10. RIC vs. % of malicious peers in threat model *A* - (a) 20% , (b) 40% peers malicious

Figure 9 shows variation of HM for both types of peers. Though HM for good peers reaches a steady state as a result of topology adaptation, for malicious peers, it fluctuates. HM for malicious peers are sometimes higher than good peers. It is due to the fact that the queries forwarded by malicious peers are blocked resulting in their higher HM values. The hit here does not mean authentic hit. The authentic hit of good peers is higher than that of malicious peers as they have higher AR.

Figure 10 shows the variation of RIC for each type of peers under threat model *A* (Section 4.1). It may be observed that RIC for the good peers increases to 2.4

constrained by the edge limit, whereas for malicious peers, RIC does not increase beyond 1.2. With the increase in the percentage of malicious peers, the saturation rate slows down but the final value remains the same. This shows that the proposed algorithm provides better connectivity to the peers who share large number of authentic files and isolates the malicious peers. Figure 11 shows variation of RIC under threat model *B* (Section 4.1). Since in this scenario, a malicious peer provides fake files when it has achieved higher connectivity independently and stops acting maliciously after it has lost sufficient number of edges, fluctuation in RIC persists throughout the simulation.

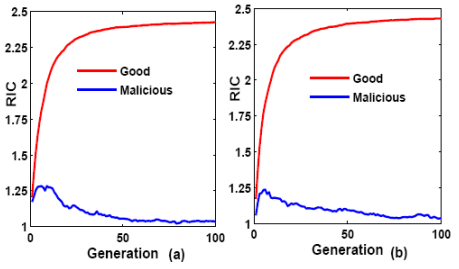

Fig. 11. RIC vs. % of malicious peers in threat model *B* - (a) 20% , (b) 40% peers malicious

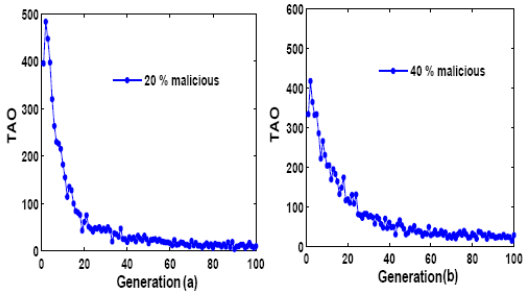

Fig. 12. Overhead due to topology adaptation with - (a) 20%, (b) 40% peers malicious

Finally, the overhead due to the topology adaptation mechanism is analyzed. The load due to topology adaptation is measured by the metric *topology adaptation overhead* (TAO), which is the number of community edges added or deleted in a generation. Figure 12 shows the variation of TAO for different percentages of malicious peers. It is observed that TAO starts falling from an initial high value and oscillates with small amplitudes. This is due to the fact that initially the edge capacities of peers are not saturated and they acquire community edges rapidly. As the simulation proceeds, good peers acquire relatively stable neighborhood resulting in steep decrease in TAO. For higher values of generations, TAO fluctuates slightly since good peers delete existing edges with malicious peers as soon as the malicious peers are discovered, and acquire new community edges with good peers. With increase in percentage of malicious peers, fluctuations of TAO also increase as there is more addition and deletion of community edges. However, in any cases, TAO becomes drastically low once the community topology becomes matured. This shows that the proposed algorithm introduces negligible overhead to the system.

6 Conclusion

In this paper, the challenges in P2P file sharing networks are highlighted and a mechanism is proposed that solves multiple problems e.g., inauthentic download, free riding and poor search scalability in an open P2P network. It is shown that by topology adaptation, it is possible to isolate the malicious peers while providing high query hit for good peers. Simulation results have shown that protocol is robust even in presence of a large percentage of malicious peers. Although popular file sharing networks like Gnutella [1] exhibits super-peer architecture, the proposed scheme has been designed for an unstructured network. Making the scheme compliant to super-peer architecture constitutes a future plan of work.

References

1. Risson, J., Moors, T.: Survey of Research Towards Robust Peer-to-Peer Networks. Computer Networks 50(7), 3485–3521 (2006)
2. Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H.: Chord: A Scalable Peer-to-Peer Lookup Service for Internet Application. In: Proc. of ACM SIGCOMM (2001)
3. Schafer, J., Malinks, K., Hanacek, P.: Peer-to-Peer Networks Security. In: Proc. of the 3rd Int. Conf. on Internet Monitoring and Protection (ICIMP), pp. 74–79 (2008)
4. Abdul-Rahman, A., Hailes, S.: A Distributed Trust Model. In: Proc. of the Workshop on New Security Paradigms, pp. 48–60 (1997)
5. Guo, L., Yang, S., Guo, L., Shen, K., Lu, W.: Trust-Aware Adaptive P2P Overlay Topology Based on Super-Peer-Partition. In: Proc. of the 6th Int. Conf. on Grid and Cooperative Computing, pp. 117–124 (2007)
6. Condie, T., Kamvar, S.D., Garcia-Molina, H.: Adaptive Peer-to-Peer Topologies. In: Proc. of the 4th Int. Conf. on Peer-to-Peer Computing (P2P 2004), pp. 53–62 (2004)
7. Tain, H., Zou, S., Wang, W., Cheng, S.: Constructing Efficient Peer-to-Peer Overlay Topologies by Adaptive Connection Establishment. Computer Communication 29(17), 3567–3579 (2006)
8. Zhuge, H., Chen, X., Sun, X.: Preferential Walk: Towards Efficient and Scalable Search in Unstructured Peer-to-Peer Networks. In: Proc. of the 14th Int. Conf. on World Wide Web (WWW 2005), Poster Session, pp. 882–883 (2005)
9. Adar, E., Huberman, B.A.: Free Riding on Gnutella. First Monday 5(10) (2000)
10. Tang, Y., Wang, H., Dou, W.: Trust Based Incentive in P2P Network. In: Proc. of the IEEE Int. Conf. on E-Commerce Technology for Dynamic E-Business, pp. 302–305 (2004)
11. De Mello, E.R., Moorsel, A.V., Fraga, J.D.S.: Evaluation of P2P Search Algorithms for Discovering Trust Paths. In: Wolter, K. (ed.) EPEW 2007. LNCS, vol. 4748, pp. 112–124. Springer, Heidelberg (2007)
12. Li, X., Wang, J.: A global Trust Model of P2P Network Based on Distance-Weighted Recommendation. In: Proc. of IEEE Int. Conf. of Networking, Architecture, and Storage, pp. 281–284 (2009)
13. Crespo, A., Garcia-Molina, H.: Semantic Overlay Networks for P2P Systems. Technical Report, Stanford University (2002)
14. Schlosser, M.T., Condie, T.E., Kamvar, S.D., Kamvar, A.D.: Simulating a P2P File-Sharing Network. In: Proc. of the 1st Workshop on Semantics in P2P and Grid Computing (2002)