Efficient Search Techniques in Peer to Peer Networks

Tarunpreet Bhatia, Dr Deepak Garg
Computer Science Department
Thapar University, Patiala

ABSTRACT
In order to use Internet resources efficiently we need to search and locate information efficiently. System performance diminishes by either duplicating a large quantity of data on each and every node or flooding query to all the nodes in the network. Firstly, this paper reviews various searching algorithms. Search techniques can be classified as blind search in which information about neighbors is not kept by the peer and informed search where peers store information for routing queries to other nodes. It discusses how range queries can be processed efficiently by rotating scheme over structured P2P systems and secure searching algorithm based on topology adaptation which penalizes the malicious peers. Genetic algorithm providing parallel search are also covered in the paper. Lastly, it focuses on merits, demerits and applicability of these algorithms in different situations.

Keywords: P2P search techniques, P2P optimization techniques, P2P networks

1. INTRODUCTION
Peer-to-peer, or abbreviated as P2P architecture is a type of network in which each workstation has equivalent capabilities and responsibilities. These differ from client/server architectures where some computers are dedicated to serve others. Thus the key features of P2P systems are

- Multiple peers (at edge)
- Distributed resources
- Peers have equal functionality

The advantages of P2P Networks are:
(i) An important goal in peer-to-peer networks is that all clients provide resources. With the arrival of nodes demand on the system increases which increases the capacity of the system.
(ii) The distributed nature of P2P networks increases robustness in case of failures by replicating data over multiple peers.
(iii) The upside of the peer-to-peer is that it is relatively inexpensive and fairly simple to set up and manage.
Peers in network contribute resources like storage space, bandwidth etc. The participating nodes mark some part of their resources as shared allowing other peers in network to share resources. Consider a file is stored in node1 which can be downloaded by node2. Now if some other node in the network wants to access that file it has option of either accessing it from node1 or node2. The path followed depends on the searching algorithm used. The purpose of file sharing in P2P architecture is to accept user's query in the form of keywords described by regular expressions, to search and locate the positions of data and return results of the query (pointers) to the user.
P2P network consists of all the participating peers as network nodes. If one peer knows the location of another peer in the P2P network then there is a directed edge from the former node to the latter. Search query travels along a sequence of edges. When user submits a query, source node forwards the query to all or set of its neighbors based on some routing policy. The node receiving a query either processes the query over its local collection or sometimes without processing forwards the query to its neighbors if results are not found. Otherwise it will reply back to source node via a reverse path as traveled by query. The result set of a query consists of union of results from all the nodes that processes the query and reply back. Data is scattered across a large number of peers. So this architecture demands efficient search techniques for retrieval of data. The basic idea is to reduce number of hops taken by a query in order to get evaluated.

2. P2P ARCHITECTURES

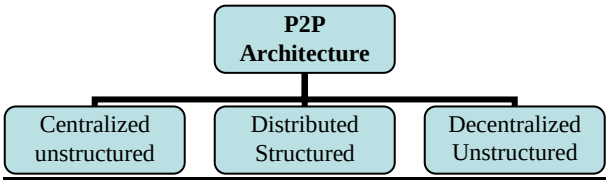


Figure1: P2P Architecture

Centralized Unstructured Architecture: There are special nodes acting as servers which store information about the resources that are provided by other peers. Example is Napster that keeps document indexes up-to-date in a centralized node. Query is evaluated fast but has single-point of failure.

Distributed Structured: The resources are distributed evenly among all the participating nodes in a predefined manner and servers are used. When a peer wishes to look up information it uses DHT to find out where particular piece of information is stored. Its search overhead is small but maintenance overhead is huge for highly dynamic system.

Decentralized Unstructured: Broadcasting search is the basic algorithm for Decentralized Unstructured. It selects its neighbors arbitrarily. There are no servers for storage of document index information so no control over topology. E.g. Gnutella It needs less maintenance overhead, simple and robust. But the demerit is that peers become overloaded and does not scale up with increase in queries that are submitted.

3. SEARCH TECHNIQUES
Search techniques are classified as:
Blind techniques: Nodes do not keep information about an object location. They are as follows:

- Expanding ring search and Blocking expanding ring search
- Iterative deepening
- Random walker

Informed techniques: Nodes gather some metadata which assist in search operation. They are as follows:

- Percolation search
- Range Query (DHT)
- Chord overlay for range query
- Robust Trust Management Search Mechanism
- Resource search based on genetic algorithm

3.1 Blind Search Techniques

3.1.1 Expanding ring search (ERS)

Expanding ring search is first flooding algorithm based on controlled TTL [1]. It involves successive flooding searches, each with different TTL value. The source node S first sends query messages to its neighbors with initial TTL value. Messages are forwarded until TTL expires or message becomes redundant. If within timeout interval no response comes, source assume that this has failed and start new iteration by larger TTL value. It reduces huge broadcasting overhead by controlling TTL value.

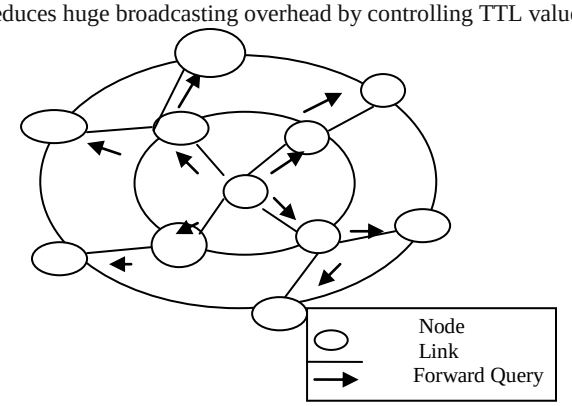


Figure 2: ERS

Blocking expanding ring search (BERS)

It is extended version of ERS [2]. If within timeout interval no response comes, source starts new iteration by larger TTL value from all the nodes of last attempt not the source node.

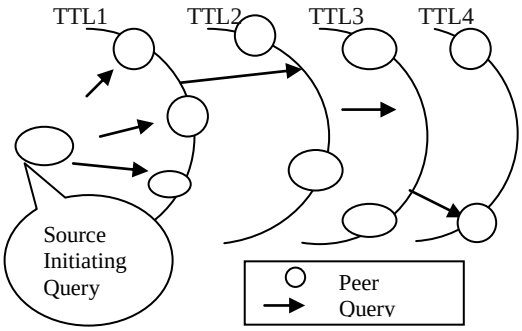


Figure 3: BERS

3.1.2. Iterative deepening

In iterative deepening [3], there are multiple breadth-first searches with successively larger depths, and query propagates until it is satisfied, or the maximum depth D has been reached. To implement the iterative deepening, a *policy* that tells the depth at which different iterations have to occur is needed. For example, if we have three iterations: the first iteration searches to a depth d1, the second to depth d2, and the third to depth d3. Thus Policy is specified as P = {d1, d2, d3}. We must also specify the time between successive iterations.

Procedure Iterative-deepening

1. Set current depth = d1
2. Source node S initiates BFS with depth d1
3. Node at depth d1 processes query and stores it temporarily.
4. The query is frozen at all peers that are d1 hops from the source.
5. S receives Response messages from peers which have processed the query.
6. Now S will wait for some specified time say T. If the query has been satisfied, then S does nothing; else S will start the next iteration, initiating a BFS of depth d2.
7. To initiate the next BFS, S has to send a Resend with a TTL of d1. Instead of reprocessing, a node receiving Resend message will forward the message, or if the node is at depth d1, it will drop Resend message and unfreeze the matched query and forward the query to all its neighbors with TTL = d2-d1.
8. Every query is assigned a unique identifier to match queries with resend message. The Resend message contains this identifier.
9. The process continues in similar manner to other levels of policy. As d3 is depth of the last iteration, there is no need of queries to be frozen at depth d3, and no other iteration is initiated by S.

3.1.3. Random Walk

Random walk (Markov chain) is a construction of path from source to destination by taking successive random steps in different directions. The Markov chain ensures the system is memoryless i.e. past states are irrelevant for predicting the future states. In Random Walk [4] query message is forwarded to one randomly selected neighbor. The message will come to an end when either the number of results satisfied, or message becomes redundant, or its TTL expired. It is based on the DFS (Depth-First Search) algorithms. The main concept behind it is to limit the message broadcasts by having control over the number of peers.

Christos Gkantsidis et. al. proposed in [5] that in an unstructured P2P networks random walker technique is better than flooding, if the following two conditions are valid :

- Cluster Formation

Peers in the network can form clusters so whole topology can be partitioned into two tiers, the lower tier represents clustering and the higher tier represents representatives from each cluster forming super nodes to ensure good global connectivity.

A virtual cluster is formed among members sharing similar contents and hence a search for a resource by any peer of that cluster is more successful in the cluster than searching in the rest of the network. However in case of flooding no such clusters exist so probability of results returned by random walk in less hops is more than flooding. Information about peers forming cluster is stored in address cache of each peer.

- Repeating Same Search Requests

When a request is issued by a peer, the user can re-issue the same request multiple times in hope of searching more peers for results. Since there is no randomness in flooding it would mostly give already known results provided the topology does not change between the repeated requests. But random walks choose neighbors randomly so it will follow different path and thus search new sources that provides results.

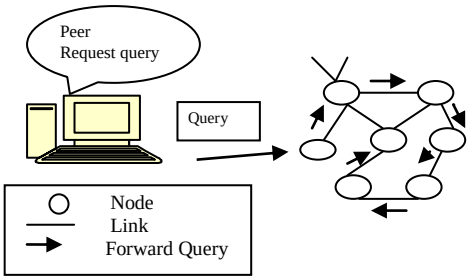


Figure 4: Random walk in P2P

The disadvantage is that response time is much longer so k-walkers algorithms (teeming) a little variation in which query message is forwarded to k neighbors chosen randomly.

3.2 Informed Search Techniques

3.2.1. Percolation Search

Percolation search technique is an efficient searching technique for unstructured P2P networks following Power Law (PL) distribution. In a PL distributed networks certain peers are very highly connected though their number is less, and most of the other peers have low degree distribution. A peer is said to be highly connected if degree of peer is >= the max degree in the P2P network/2. The percolation search algorithm takes advantage of this by sending query to these highly connected peers because then the probability of getting results at these peers is high. Nima S, et. al. proposed in [6] that the entire process of search can be divided into three phases -

- Content List Implantation

A peer in this phase sends its list of contents to other peers in the network. It is accomplished by taking a random walk through the peers in the network, and duplicating its content list at each peer encountered during the random walk of size O(log*N*) where N = total number of peers in the network. This walk is carried by all the N peers so total number of contents is O(*N* log*N*) and the average cache size is O(log*N*). This process ensures availability of every content and query on atleast one highly connected peer. In figure 5, a random path is selected from peer S and reaches peer A. It implants its content list at peer A and continues random walk. It also saves its content list at peer B and peer H which is a highly connected peer.

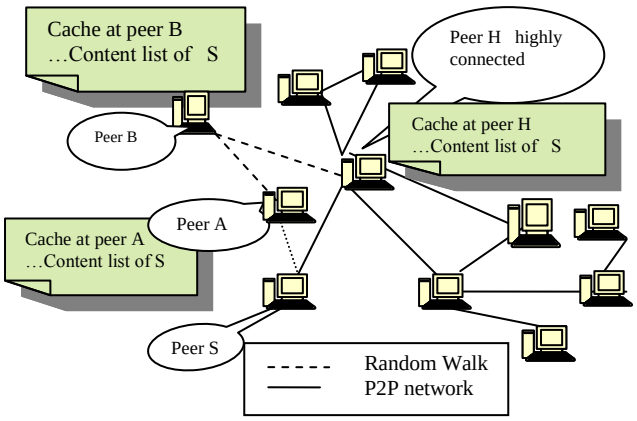


Figure 5: An example showing the content list implantation of peer S.

- Query Implantation

It is the first phase taking place whenever some peer issues a query. The requesting peer uses random walk and at each successive step implants query at the peer encountered. So now all the peers that have query implanted on them take part in search process along with query initiator. This process ensures availability of query on at least one high degree peer. If this happens then the query can be answered in minimum hops.

- Bond Percolation

In this phase actual lookup for search of key value is performed. The peers lookup their routing table in order to find which neighbors are to be chosen to forward query and sends them query probabilistically. Probability of query being sent to a neighbor = percolation threshold / exponent of PL distribution

The example in figure 6 shows a percolation search. Peer S is the query initiator and it has cached the query on peers A and B. All four peers S, A, B and H choose the neighbor to whom they have to send the query probabilistically. Peer D has the result and peer H knows this because it has the content list of D implanted on it. H queries D and D replies with the result.

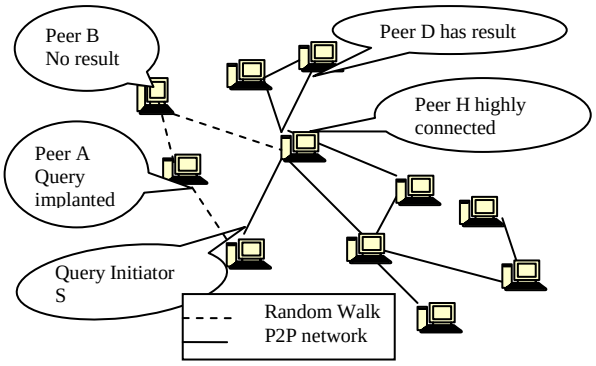


Figure 6: An example showing the bond percolation search phase

3.2.2. Range Query

Range queries search for a range of key values between the given upper and lower limit of key values. Range queries are used effectively in the DHTs using order preserving hash functions. DHTs using Chord are not suitable for answering range queries because of random distribution of keys. If the semantically related keys are stored near to each other then results can be generated in the minimum number of hops. It demands the use of a data structure that promotes semantic relationship.

Trie is such a data-structure. It is an ordered tree for storing strings in which there is one node for every common prefix. The strings are stored in extra leaf nodes (Definition by the National Institute of Standards and Technologies). Tries cluster semantically close data items which leads to efficient processing of range queries. P-Grid using trie as the overlay network is such a DHT. P-Grid associates all the peers with key values from the keyspace. Each peer stores some part of the overall key space and also maintains routing information in routing table for forwarding queries and requests. The keys stored in peers are calculated by order preserving hash function i.e. given two input strings a and b, if a is prefix of b the key(a) is also prefix of key(b).

The peer at each level of the tree has references to other peers that do not form the peer's subtree at that level, enabling the implementation of prefix routing. The data keys stored by the peers in their disks are prefixed with the path of the peer. The information about the path to other peers is contained in routing

table so that the peer can route a query to other peers in the network.

For e.g. the query is looking for data key '111' and is forwarded to peer 5 by the query initiator. Peer 5 knows that key is stored in some peer in the subtree where path start with '1'. Peer 5 knows the address of peer 3 and so the query is forwarded to peer 3. Peer 3 after looking at the query knows that it should go to a peer in the subtree where paths start with '11' and so it forwards the query to peer 6. Peer 6 happens to have the key the query is attempting to find so the search ends here.

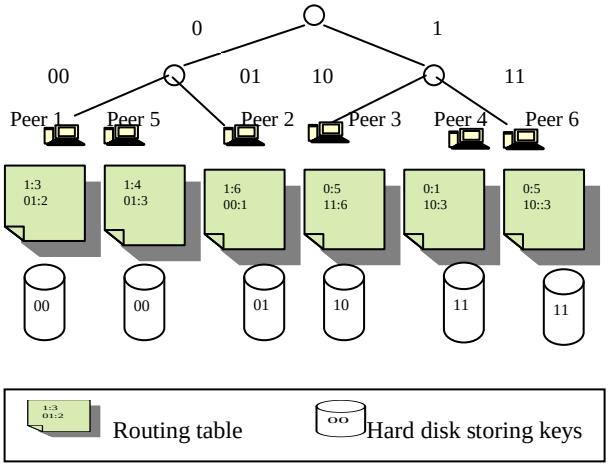


Figure 7: An example of P-Grid.

Anwitaman Dattal proposed in [7] that a range query can be processed on an order preserving DHT in following ways –

- Min-max traversal algorithm: In this algorithm, peer containing data items belonging to lower bound of range processes query first and then forward it to a peer holding next partition of key space until a peer having upper bound of range as data item is found.
- Shower algorithm: Shower algorithm does concurrent processing of range queries. The range query is forwarded to any peer that is responsible for holding any of key space partition within the given range, and then using routing table of the peer it is forwarded to other partitions in the given range.

3.2.3. Chord as an overlay network

Like tries it is also an overlay network built on top of a DHT. In this logical ring is constructed with all participating nodes. Order preserving hash function is used to distribute keys evenly over logical ring. DHT maps the nodes in identifier space and keys are also mapped to same space using some hash function H. Key value k is stored at node having identifier closest to H(k). This property ensures storage load balancing by random distribution of keys but destroys order of keys. So it is efficient for match queries, in order to process range queries efficiently order preserving hash function is used.

Rotating scheme [8] is used to balance storage load and new nodes are added in ring to share load of existing overloaded nodes. Adaptive replication scheme [8] for access load balancing in which overloaded node replicates some of its keys to underloaded nodes and then forwards query to them.

Average load desired on each node is L=K/N Where K= number of keys to be stored and N= total number of nodes to be stored.

A node n is overloaded whenever its load crosses the threshold value and in terms of access load it is overloaded if access count exceeds threshold chosen and underloaded if it is less than lower limit.

- Rotating scheme:

Let us say in figure 8 query for range [9, 19] is initiated at node 48. H (9) = 18 say. Node 48 asks node 16 to find successor of key 18 and it turns out to be node 23. All the keys that are within required range are retrieved and then forward query to node 34. This process is repeated until query reaches node 44 that contain the upper limit of key range 19.

Algorithm for Node_Join(n, b) – None n is the new joining node and b is bootstrap node to whom new nodes contact first before joining. l is list of overloaded nodes.

```
1.  If b.load > T Then
2.      Put b in the list l of overloaded nodes;
3.  s = b.successor   /* Forward the token to b 's successor */
4.  findMostOverloadedNode(b, s, l);
5.  Sort the list l of overloaded nodes in ascending order;
6.  While(there exists an overloaded node in  l )
7.      Check the status of the overloaded node;
8.      If Available Then
9.          The joining node n shares load with
    the overloaded node
10.         Exit
```

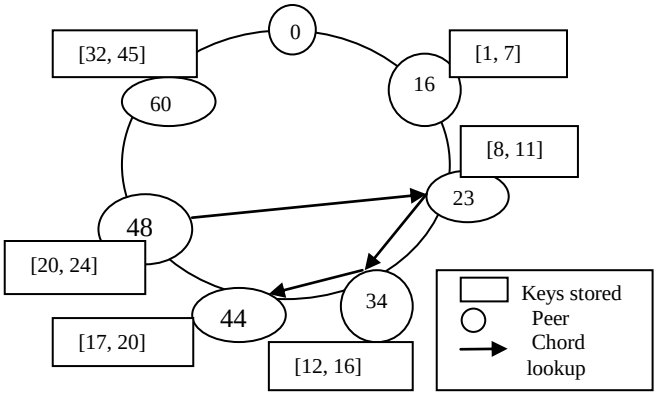


Figure 8: Range query processing

If there is no overloaded node then the new node shares load with the node which is most loaded.

Improvement: In order to alleviate token conflict problem we can use alternate strategy i.e. instead of finding lists of overloaded nodes just reserve first overloaded node for first token and when second token comes it will see this node is reserved and try other overloaded node. The worst case is O(N) steps to find overloaded node.

- Adaptive replication scheme:

The main aim is to minimize dropped requests by replicating keys over underloaded nodes in terms of access requests received by node.

Algorithm for Replication (*n*) – b is initiator i.e. bootstrap node, l is list of underloaded nodes, key is the key values of node, r is replica holder.

```
1. If b.load > max_lmt Then
2.     s = b.successor;   /* Forward the token to n's successor */
3.     Send token to get a list of all underloaded nodes;
```

4. Sort the list of underloaded nodes in ascending order;
5. Req_Fwd = If b.load - max_lmt // No. of requests to be forwarded for replication
6. While there exist an underloaded node in list l
7. Contact the underloaded node;
8. If Available Then
9. If r.Req_Rcv ≥ b.Req_Fwd Then
10. Invoke Replicate(b, r, doc);
11. Exit;
12. Else
13. Req_Remaining = b.Req_Fwd – r. Req_Rcv
14. b.Req_Fwd = b. Req_Remaining;
//Now it has to forward remaining requests
15. Invoke Replicate(b, r, key);

The figure 9 depicts how the query initiated at node 10 for key 38 is answered. Node 43 is overloaded node so its replica holders are node 35 and 62. By chord lookup node 10 sends query to node 35 then to 43. After receiving request, node randomly picks its replica holder say 62 and route query to it.

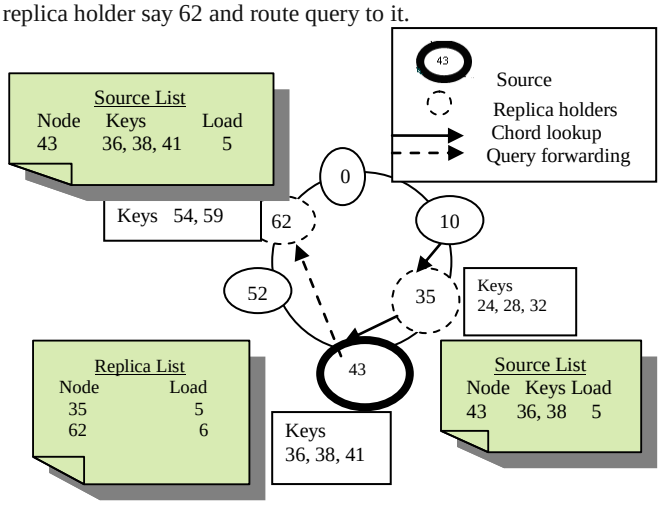


Figure 9: Replication

3.2.4. Robust Trust Management Search Mechanism

There are several trust management algorithms that deal with malicious peers but they have one or the problems of communication latency, high computation cost etc. A robust trust management scheme [9] is based on topology adaptation and neighbors are selected on the basis of their trust ratings and similarities in contents. A TTL bound search is used. Network has been modeled as a power law graph in which there are certain nodes having high degree of connectivity and most others low according to [10]. There are two types of links in the network:

- Connectivity links: These form the edges that exist between nodes initially in power law network.
- Community links: They are added dynamically between peers based on their contents.

Algorithm for Query initiation

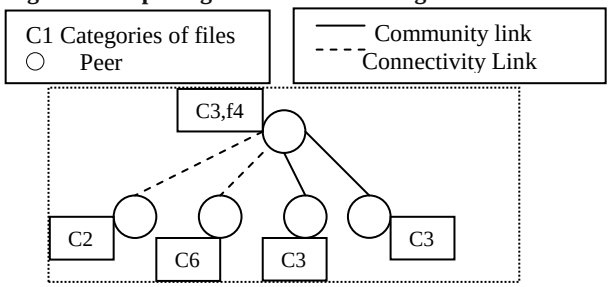
1. Source node forms a query packet for file(c, r) where c stands for category and r for rank along with TTL.
2. Calculate number of neighbors whom packet is to be forwarded. No. of neighbors = (1-$Prob_{comm.}$)*(Actual neighbors that can communicate) where $Prob_{comm.}$ = connectivity index

If we have 20 neighbors and $Prob_{comm.}$= 0.4 then no. of neighbors to whom packet is sent say n = (1-0.4)*20= 12.

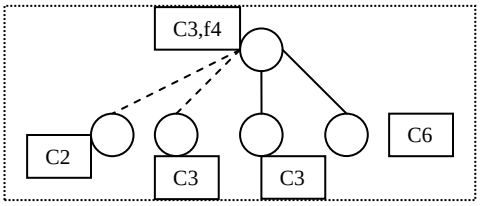
3. Select neighbors using some selection rule.
4. Now just forward packet to selected set of neighbors.

Neighbor selection rule: Trusted neighbors having similar contents that are linked by community links are preferred. If community links are not sufficient then connectivity links are used.

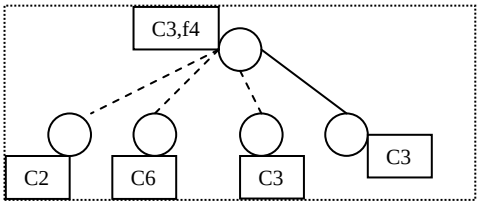
Figure 10: depicting various cases of neighbor selection



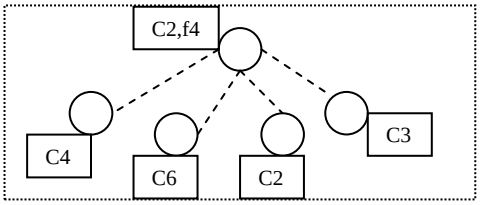
Case 1: Both neighbors with community edges are selected and have files of desired category C3.



Case 2: One neighbor having files of desired category C3 with community link is selected and community link have files of category C6 is selected as it is more trusted.



Case 3: Only one community neighbor that share file is C3 is chosen. From the remaining connectivity neighbors, most trusted C2 is selected.



Case 4: only connectivity neighbors are there, assume all are at the same trust level, the matching neighbor C3 is chosen and from the remaining C4 is selected randomly.

Algorithm for Query forwarding

1. As the query packet f(c, r) from node i reaches node j, j will check trust rate of peer i.
2. If trust<=0 then
3. node j will block further propagation of packet and exit
4. else
5. if file is available then

6. send response to peer i
7. TTL=TTL-1
8. If TTL<=0
9. Calculate no. of neighbors to whom to send packet
10. Select set of neighbors using above rule
11. Forward query to selected neighbors

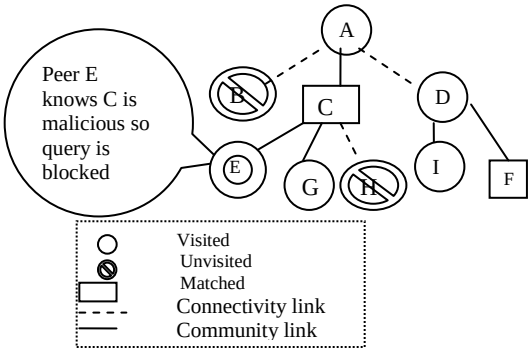


Fig 11: Search Procedure

For e.g. suppose a query is forwarded to two neighbors. Query is initiated at peer A and neighbors selected are C and D. Query reaches E from C and knows C is malicious from previous messages so block it. In the end query is matched at nodes C and F.

Algorithm for Topology adaptation
After receiving the result set from a set of peers, source peer sorts them on the basis of reputation of peers and download the file from the most reputed peer instead of just following the shortest path. Meanwhile it updates trust rating and adapt to network topology after successful or failed download. It consists of link deletion and link addition. This process is initiated by node i which receives responses for the requested file from node j

1. A Set S of responses
2. Sort set on the basis of trust rating of node j that provides responses
3. do
4. Download file from source j having highest trust rate
5. Check file status and update trust rate of j
6. Update network(i,j)
7. Until file is not spurious or no more source for file exists

Algorithm for Update network(i,j)
1. If source j is malicious then
2. delete existing community edge(i, j)
3. else
4. If approve Link(i, j) then
5. Create community edge(i, j)

Algorithm for Approve Link(i,j): There is maximum limit on the number of edges a node in network can have given by edgelmt to control bandwidth so RIC (Relative increase in connectivity) of node is checked against this constraint before adding any edge.
1. if RIC(i) or RIC(j) >= edgelmt then
2. return false
3. else
4. if trust(i)<0
5. return false
6. else
7. return true

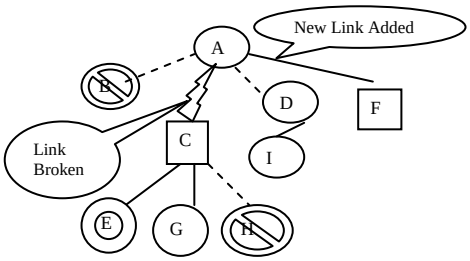


Figure 12: Topology adaptation

In figure 12 let us say node A gets response from node C and F. It will download file from node C and found file spurious so it will dynamically update trust rate of node C and deletes community edge A-C. Then it will download from F and if file is not spurious add community edge between A-F.

3.1.5. File search based on genetic algorithm

Genetic algorithm based on the process of biological evolution provides parallel search. P2P file search problem is simply an optimal path selection i.e. if results are obtained from several nodes in network then the node with the shortest distance from given node is chosen. In GA, we have survival of the fittest property. Fitness function here has certain parameters e.g. low latency or less number of hops. The common fitness function that can be used is [11]

$$f = \frac{1}{\sum_{j=1}^{l-1} C_{g(j),\, g(j+1)}}$$

F=fitness value of chromosome, l represents the number of the nodes in a path, g(j) represents the gene (node) of the j-th locus in the chromosome, C represents the edge weight i.e. link cost.

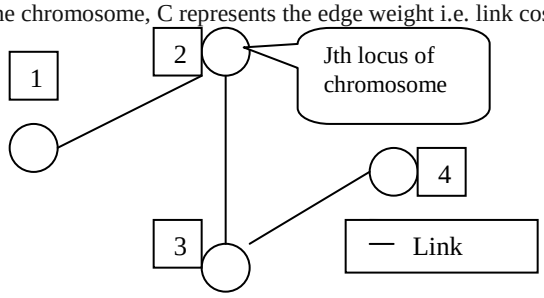


Figure 13: Chromosome representation

So taking into consideration genetic evolution, P2P file search include following steps [12]:

- Genetic Representation: For this IP address of each node is chosen and chromosome is a sequence of nodes from source to destination.
- Create an initial population: DFS is used to search path from source to destination and create *N* unique chromosomes for the first generation where *N* is the population size.
- Selection: In this process the fittest chromosomes are selected for the next generation. After selection, an iteration cycle is completed. The solution converges when good chromosomes in the previous and current generation are the same.
- Crossover: For this crossover point is selected first. It is a node common between two chromosomes other than the

source and destination. E.g in figure 14 there are two paths for reaching Destination D from source A. So two chromosomes are < A, B, C, G, H, D> and <A, C, E, F, D>.Crossover node is C and children chromosomes after crossover operation are <A,B,C,E,F,D> and <A,C,G,H,D>.

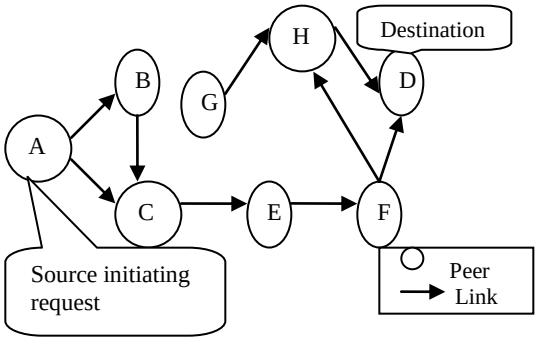


Figure 14: Structured P2P network

- Mutation: The purpose of mutation is to find the optimal path solution by mutating some nodes in a chromosome. One node is selected from sequence of nodes in a chromosome as mutation point. Among neighbors of that node one neighbor is selected as alternative route from previous node (gene). We proceed in this manner till destination is reached. Let a chromosome is having sequence of nodes as *<A, B, C, E, F, D>* as shown in figure 14, we choose a mutation point, say node E. Then we mutate the chromosome starting from the mutation point by choosing a neighbor node G based on some criteria. Hence, we get *<A, B,* C, *G, H, D>* as *G* is adjacent to *C* and *H* is adjacent to *F*.

Table1. Comparison of Different Search Techniques

Search Method	**Used in**	**Performance**	**Merits**	**Demerits**	**Complexity**
ERS	First TTL controlled technique performed on unstructured networks.	Flooding based so works same for all cases	1. Simple, low overhead	1. Non deterministic while evaluating results. 2. More overshooting messages than flooding if resources are far away from source node.	In the worst case messages are sent to all the nodes in the last round so message complexity is O(N) where N is total number of nodes in the network
Iterative deepening	Performed on unstructured networks when mere satisfaction is needed not number of results.	Performs well when popular and well-replicated items have to be discovered.	1. Improves scalability.	1. Creates large amount of duplicate messages. 2. Slow query processing. 3. More time taken because of multiple iterations.	It is TLBF time limit based flooding
Random Walk	Performed on Unstructured P2P networks where topology does not changes frequently.	Works best with repeated queries and clustering of peers on the basis of content similarity	1. Results are fast so fewer searches. 2. Less overshooting message as compared to ERS. 3. Less taxing on resources.	1. Non deterministic so success rate vary due to random choice of neighbors for forwarding queries. 2. Queries for popular and less popular data items are propagated in the same way. 3. The peers available for a long time could be chosen more often.	It is PLBF probabilistic limit based flooding. Logarithmic search time
Percolation search	Performed on unstructured P2P networks.	Works best with networks that follow power law distribution	1. Ability to answer complex queries using scalable resources. 2. Very fast results as many peers process query simultaneously. 3. Overall traffic is reduced.	1. Much taxing on resources because of pre-search requirements. 2. There is partial coverage problem as some nodes that are available for long time are chosen more often than others.	Search in logarithmic time

Range Query (Trie DHT)	Performed on DHT P2P networks	Works best when content similarity of keys is maintained	1. Ensures storage load balancing. 2. Min max algorithm uses resources efficiently. 3. Shower's algorithm is fast	1. Suffers from logical data fragmentation i.e. access to top most nodes in tries is more often than leaf nodes.	Hash based. Logarithmic search time
Chord as an overlay network	Performed on structured DHT P2P networks and widely used in multi-hop networks (e.g., ad-hoc and sensor networks)	Works best when content similarity of keys is maintained	1. Ensures both storage load and access load balancing. 2. Failure of peers wont cause system failure as data is replicated across several peers.	1. Worst case look up is very slow. 2. Failure of some nodes can cause incorrect look up.	Hash based and Logarithmic search time
Robust trust management scheme	Performed on unstructured P2P networks	Works best with power law networks	1. Robust and scalable. 2. It not only improves search efficiency to good peers but quality of search also i.e. simultaneously increasing search time for malicious peers. 3. Minimal overhead in terms of computation cost and storage	1. Much taxing on resources because of pre-search requirements and extra cache storage.	Logarithmic search time
File search based on genetic algorithm	Performed on structured P2P networks	Works well with practical P2P system	1. GA are used to decrease bandwidth for queries[12] 2. Better global capability 3. Parallel search and robust	1. Unable to use feedback information of networks. 2. While converging to global solution they do large amount of redundancy computation.	

5. CONCLUSION

Several advanced search techniques have been surveyed for P2P networks. Every technique is suitable for particular type of networks and requires certain conditions to perform well. The algorithms discussed are either flooding based or DHTs. DHT based are fast but there is overhead of maintaining topology and updating as node frequently enters or leaves the network and keys have to be more precise because of exact matching of key and address part. Flooding techniques on the other hand are not scalable but can find close matches. So the need of the hour is to analyze how these search techniques use internet resources and try to modify them so that these techniques can give more efficient, accurate and reliable results.

6. REFERENCES

[1] Chang N, Liu M Revisiting the ttl-based controlled flooding search: optimality and randomization. In: Mobile computing and networking (2004).

[2] Pu IM, Shen Y Enhanced blocking expanding ring search in mobile ad hoc networks. In: 3rd international conference on new technologies, mobility and security (NTMS) (2009).

[3] D. Tsoumakos and N. Roussopoulos, "Analysis and Comparison of P2P Search Methods," Proc.1st Int. Conf. Scalable Information Systems (2006).

[4] Reza Dorrigiv, Alejandro L´opez-Ortiz, and Paweł Prałat, "Search Algorithms for Unstructured Peer-to-Peer Networks," proc. of 32nd IEEE Conference on Local Computer Networks (2007).

[5] Christos Gkantsidis, Milena Mihail, Amin Saberi, "Random Walks in Peer-to-Peer Networks", in proceedings of IEEE Infocom (2004)

[6] Scalable percolation search on complex networks by Nima Sarshara, Oscar Boykinb, Vwani Roychowdhurya Theoretical Computer Science (2006).

[7] Range queries in trie-structured overlays Anwitaman Datta, Manfred Hauswirth, Renault John, Roman Schmidt, Karl Aberer Ecole Polytechnique F´ed´erale de Lausanne (EPFL) CH-1015 Lausanne, Switzerland.

[8] Dynamic storage and access load balancing for answering range queries in peer-to-peer networks Zaher Al Aghbari & Ibrahim Kamel & Ahmed Mustafa, Springer Media, LLC (2010).

[9] Secure and Privacy- Aware Searching in Peer-to-Peer Networks Jaydip Sen TCS Innovation Labs, Tata Consultancy Services Ltd.

[10] Tain, H., Zou, S., Wang, W., Cheng, S.: Constructing Efficient Peer-to-Peer Overlay Topologies by Adaptive Connection Establishment. Computer Communication (2006)

[11] Ahn CW, Ramakrishna RS A genetic algorithm for shortest path routing problem and the sizing of populations. IEEE transactions on evolutionary computation (2002).

[12] Wong WY, Lau TP, King PI Information retrieval in P2P networks using genetic algorithm. In: Special interest tracks and posters of the 14th international conference on World Wide Web, Chiba (2005).