Algorithm Designing for Searching in Unstructured Peer-to-Peer Network

Rakesh Rathi, Kapil Saini, Sumit Kumar Bola
1,2,3Dept. of Computer, Govt. Engineering College, Ajmer, Rajasthan, India

Abstract
Peer-to-Peer system is now one of the fastest growing and most popular applications. Centralize Peer to peer systems always suffer from the problems of single point of failure, low availability, denial of service attacks. Searching of the required data is a vital issue in both centralized and decentralized P2P network. Many methods have been implemented for searching in P2P network such as Flooding, Random Walk, Expanding Ring or Iterative deepening, K-Walker Random Walk, etc. Some of these generate large traffic while others take long searching time. A probabilistic approach with Two Level K Walker Random Walk for searching has been studied and enhanced in this paper so that these issues can slightly overcome. Advantages of two level walk are further reducing collision of nodes and helping in searching the distant nodes. But it has slightly increased the response time. This disadvantage can be overcome when there are a lot of dull nodes in the network with this enhancement.

II. Related Work
Peer-to-peer (P2P) computing has emerged as a popular model aiming at further utilizing internet information and resources. In an unstructured P2P system, no rule exists that strictly defines where data is stored and which nodes are neighbors of each other. To find a specific data item, early work such as the original Gnutella [1, 8] used flooding, which is the Breadth First Search (BFS) of the overlay network graph with depth limit D. D refers to the system-wide maximum TTL of a message in terms of overlay hops. In this approach, the querying node sends the query request to all its neighbors. Each neighbor processes the query and returns the result if the data is found. This neighbor then forwards the query request further to all its neighbors except the querying node. This procedure continues until the depth limit D is reached. Flooding tries to find the maximum number of results within the ring that is centered at the querying node and has the radius: D-overlay-hops. However, it generates a large number of messages (many of them are duplicate messages) and does not scale well.

Flooding is the basic method of searching in unstructured P2P networks however the blind flooding based search mechanism causes a large volume of unnecessary traffic, and greatly limits the performance of P2P systems. Our study shows that a large amount of unwanted traffic is divinable and can be avoided while searching in P2P networks [3, 9]. Flooding is the predominant search technique in unstructured peer-to-peer (P2P) networks. If we measure performance as the number of exchanged messages per distinct response, flooding with small time-to-live performs well in regular networks. However, its performance deteriorates as the timeto- live increases, or if the topology of the underlying network is not regular. In addition, flooding has poor granularity [11, 13].

Many alternative schemes have been proposed to address the problems of the original flooding. These works include iterative deepening, k-walker random walk, modified random BFS, two-level k-walker random walk [3, 9], directed BFS, intelligent search, local indices based search, routing indices based search, attenuated bloom filter based search, adaptive probabilistic search, and dominating set based search. They can be classified as BFS based or Depth First Search (DFS) based. The routing indices based search and the attenuated bloom filter based search are variations of DFS. All the others are variations of BFS. In the iterative deepening and local indices, a query is forwarded to all neighbors of a forwarding node. In all other schemes, a query is forwarded to a subset of neighbors of a forwarding node [9].
III. Comparison of Existing Search Algorithms

Searching strategies in unstructured P2P systems are either blind search or informed search. In a blind search such as iterative deepening, no node has information about the location of the desired data. In an informed search such as routing indices, each node keeps some metadata about the data location [3,9,11]. To restrict the total bandwidth consumption, data queries in unstructured P2P systems may be terminated prematurely before the desired existing data is found; therefore, the query may not return the desired data even if the data actually exists in the system. An unstructured P2P network cannot offer bounded routing efficiency due to lack of structure.

Among those algorithms, Adaptive Probability Search (APS) is the most efficient algorithm. APS is based on k-walker random walk and probabilistic (not random) forwarding [4]. Another interesting algorithm is Two-Level Random Walk in which walkers are searching for an object in two levels. So it reduces the redundancy of nodes [4, 7].

IV. Adaptive Probabilistic Search

APS utilizes quantitative data in the form of probabilistic information for the purpose of guiding search operations. The major difference with random walkers is that in APS a node makes use of answers from earlier searches to probabilistically direct future search walkers, instead of random walker search. In APS, each node maintains a table for the forwarding probability to each neighbor for each resource. The value of each entry in the table mirrors the relative probability of this node’s neighboring node to be selected as the next hop in a future request for the specific object. Each node stores a relative probability (e.g., an unsigned integer value) for each of its neighbors for each (directly or indirectly) requested object. So, for R such objects and N neighbors, O(R × N) space is needed. For a typical network node, this amount of space is not a burden. On nodes with limited storage capacities, index values for objects not requested for some time can be erased. This can be achieved by assigning a time-to-live value on each newly-created or updated index, or by expunging the least recently (or frequently) used indices. Each search or update message carries path information, storing a maximum of TTL peer addresses [4].

APS exhibits many plausible characteristics, such as:

- High accuracy
- Low bandwidth consumption
- Large number of discovered objects
- Robust and adaptive behavior in rapidly-changing environments

V. Enhanced Two Level TTL with APS

In this extended technique we used 2-level TTL with APS and we have also included the concept of tabu search. A Two Levels TTL for Unstructured P2P Network using Adaptive Probabilistic Search technique has been improved [7]. It is an efficient search algorithm, which increases the total number of nodes searched and reduces the redundancy or average number of times a particular node is searched [17]. So collision of nodes can be further reduced and also distant objects can also be search efficiently. Two level walk will also help in further reducing message overhead. Only disadvantage will be increased in response time.

In the case of Two Level Random Walk we generate K1 threads which will be less than K which we have used in K Walker Random Walk [7]. At the edge nodes where TTL1 expires and the search is unsuccessful then second level will start and these K1 threads will exploit in K2 threads subsequently a new TTL2 will be initialized which will be less than TTL1. In this case we are generating fewer threads so chances of collision will decrease than other searching algorithms.

A. Algorithm

1. Steps

- In the first step, every new node that is to be added in the random graph will make its neighbours aware of its object population.
- Every node than make a tabu list on the basis of a threshold value. (it will check that nodes object population is less than or greater than of that threshold, if it is less than the threshold then the node is added in the tabu list)
- In the third step we start the level one TTL. K1 threads are generated.
- After expiration of TTL1, the edge node will first check the tabu list.
- Every edge node mark its neighbours who r not in the tabu list. It will send the query to these nodes. If all the neighbours are there in tabu list then go to the other walker.

2. Pseudo Code

Assumptions

Each node maintains a Tabu list on the basis of a threshold—T
(min. no. of objects that a neighboring node should poses) k1 = k2 = k = k3 – No. of walkers in each level of algorithm ttlcount – counter for time to live value
l1 = ttl2 = ttl - Time to live for each level of search level – level number
kcount – counter for k, i.e. number of walkers in any level
level = 1
kcount = 0

while (level <= 2)
{
    while (kcount <= k3)
    {
        while (ttlcount <= ttl)
        {
            if(level == 2)
            {
                Check tabu list;
                If(all neighbors are in tabu list)
                Then go to label;
                Else
                process list of neighbors that are eligible for search;
            }
            select a neighboring node by applying APS(check probabilities) and Process the node;
if object is not found then
    increment ticount by one;
else
    come out of the loop (exit);
end if

In this technique every node will generate a tabu list. Which contains neighboring nodes those have very less no. of objects (less than a predefined threshold). In the first level of TTL, this tabu list will not be considered.

It works in the following manner. When a node wishes to send a query with a certain search key, it composes a search message and broadcasts it to k1 randomly selected neighbors. The message has an initial TTL1 = 11 hops. When an intermediate node receives this message, it checks the TTL1 timer. If the latter is still more than 0 then it decrements the timer by one, selects one random neighbor and forwards the message to it. This process continues until one of the nodes, say node E, receives the message with an expired TTL1 timer (i.e. TTL1 = 0). We call such a node an edge node. The message will then “explode” into k2 search messages forwarded from this node. Specifically, node E will compose a message with TTL1=0, and a second timer TTL2=k2. When the first level of TTL is complete on some path and the searched object is not found then the edge node will first check the tabu list. If its all neighbors are there in the tabu list then the search is terminated. If it has at least one neighbor that is not there in the tabu list then it will compare the probabilities as in last algorithm.

So in the second level of the algorithm every node will first check the tabu list. If it doesn’t have at least one neighbor, that is not in the tabu list then the search will be terminate. This way, the same number of search threads can be generated (k=k1*k2) but with a larger number of nodes searched and a smaller probability of redundant searches to the same nodes using the same number of total search steps.

VI. Conclusion
A new search technique Two Levels TTL for Unstructured P2P Network using Adaptive Probabilistic Search which also includes the concept of tabu search and helps in further enhancing the performance of APS is designed. Further work involves reducing the size of the query message which will reduce the bandwidth. Advantages of two level walk will further reduce collision of nodes and also helps in searching the distant nodes in the network. Our proposed technique has all the advantages of two level ttl with APS algorithm and it also overcomes the disadvantage of increased response time when there are a lot of dull nodes present in the network.

References

ISSN : 0976-8491 (Online) | ISSN : 2229-4333 (Print)
Kapil Saini is pursuing his M. Tech. (CS) Fourth Semester from Govt. Engineering College Ajmer. He has done his b-tech in computer science from Rajasthan Technical University. His areas of interest are Peer to Peer networks and mobile networks.

Sumit Kumar Bola is pursuing his M-tech (IT) Fourth Semester from Govt. Engineering College Ajmer. He has done his B-tech in Computer Science and Engineering from Shri Mata Vaishno Devi University. His areas of interest are Mobile Ad-hoc network, wireless sensor network, Peer to Peer network and algorithm designing.