Taxonomy-Based Routing Indices for Peer-to-Peer Networks

Luca Pireddu Mario A. Nascimento
Department of Computing Science
University of Alberta, Canada
{luca,mn}@cs.ualberta.ca

Abstract

On the one hand, the lack of structure of Peer-to-Peer (P2P) networks is key to their robustness and accessibility. On the other hand, this same lack of structure creates difficulties in efficiently searching the contents of the network. This search problem must be addressed for P2P networks to grow beyond the world of file sharing. To this end, we present a novel approach for describing the documents accessible through peers as a taxonomy. We propose a scoring function which is used to route queries within the network based on such taxonomical information, as well as the number of results desired by the query. The scoring function aims to minimize the number of network messages required to answer a query. When comparing to a sequential query forwarding algorithm, our simulations have shown that our proposed technique is able to reduce the number of messages generated for a query by a factor of 10. Also, our experiments show that limiting the "time to live" of a query is likely to make queries more expensive.

1 Introduction

Peer-to-peer (P2P) networks have great potential to become an excellent information sharing tool thanks to their accessibility. Clear evidence of this characteristic is their success as a file sharing technology [1, 8]. Yet, P2P networks have insofar failed to to become a method for *information* sharing. Part of the reason surely lies in the fact that the P2P networks that are currently popular, e.g., Gnutella [8], are limited as *file* sharing tools due to their name-based search system. However, some research has been presented to supersede this simple search system with content-based search [3, 6].

The next step to progress P2P networks to the level of a useful information sharing tool is to address the issues present in organizing the communication required to search the network. P2P networks are characterized by a lack of structure. This makes them robust to sudden node failures on the network. Unfortunately, it also complicates the task of systematically querying it. Traditionally, searches in such unstructured P2P networks are based on file names and performed by flooding all peers with a query message. Flooding propagates a query messages to every node within a specified radius (the query's *Time-to-Live* or TTL) from the initiator of the search. Therefore, flooding can quickly retrieve any related documents within TTL hops of the initiator. However, it leads to significant network usage by generating a large number of query messages. More importantly, the amount of traffic scales poorly as the number of nodes in the network increases. Consequently, flooding is not a scalable searching method for P2P networks in general.

The only work we are aware of that addresses the network usage problems associated with flooding are the Routing Indices presented in [2]. The authors demonstrated that their technique is effective under their test circumstances. However, this technique scores query routes assuming that the network has the topology of a regular tree, which is not the typical case.

For the purposes of this work we assume that the contents of a P2P network node are partitioned into categories of a predefined taxonomy. In addition to every peer node having this tree information, we also assume the peers knows how many documents it has in each category. Given that, it is reasonable to assume that every node in the peer's taxonomy tree can also store the total number of documents in the nodes beneath it. For simplicity we assume that leaf nodes are unique and that every peer is aware of the same taxonomy. While this model would likely not be as useful for the public at large (e.g., audio files sharing community), there are many domains where it does fit. For instance, in academic publications, such as many from ACM's Proceedings, each paper is described by a set of keywords extracted from an existing taxonomy. Another domain of application is that of biology where living forms are categorized under a formally defined taxonomy.

The type of query we consider, which could be posed from any of the peer nodes, searches for a given number of documents in a given topic, i.e., a given leaf node in the taxonomy.

However, it is clearly impractical to assume that every peer in the network knows about every other peer. Therefore, to reduce the amount of information that has to be transferred between peers to describe their collections, our scheme takes advantage of the hierarchical nature of the taxonomy. Peers that are closer are described more precisely, i.e., by lower levels of the taxonomy, and therefore in greater detail. On the other hand, peers that are further away are described by higher levels of the taxonomy and are thus described with a smaller amount of information. For instance, let us assume a simple taxonomy tree with a root node with two leaf nodes, i.e., categories, only. Peers one hop away from a given peer P know exactly how many documents P has in each category. Peers at a distance of two hops know only the overall number of documents that P has, but not how many in either category. Finally, network nodes three hops or farther away would not know anything about P's contents. This guarantees that the amount of information shared does not grow with the network size but rather with the taxonomy's size, which is considered to be much mode stable. Hence, the routing scheme can make decisions about how to direct a query messages based on a combination of the estimated number of hops required to fulfill a query via a particular peer, and the estimated number of related documents accessible via a particular peer. This process is described in finer detail in section 3.

The main contribution of this paper is therefore the proposal of the Minimum Expected Hop Count (MEHC) technique for query routing. MEHC allows a taxonomy-based query, requesting a given numbers of documents, to traverse the peer-to-peer network effectively. Compared to a baseline technique where all neighbour peers are searched not using any information about the peers' contents, MEHC can process a query using up to 10 times less messages. Another interesting result is that limiting a query TTL is not necessarily a good idea since it prevents the query from going more directly to a good source of information.

2 Background

In the taxonomy-based routing scheme, each node in the P2P network describes the information accessible through each of its peers with a hierarchical document classification system. As

discussed earlier, collections that are closer to a peer are described in detail by the lower levels of the hierarchy. On the other hand, collections that are further away are described in a coarser way by the higher levels of the hierarchy. To describe the collections accessible via a connection, the hierarchy stores in each node n of the taxonomy the number of documents in the corresponding category or topic that are exactly $height(n)$ hops away. One instance of this structure has to be stored for each directly-connected peer. The workings of this data structure are illustrated in Figures 1 and 2.

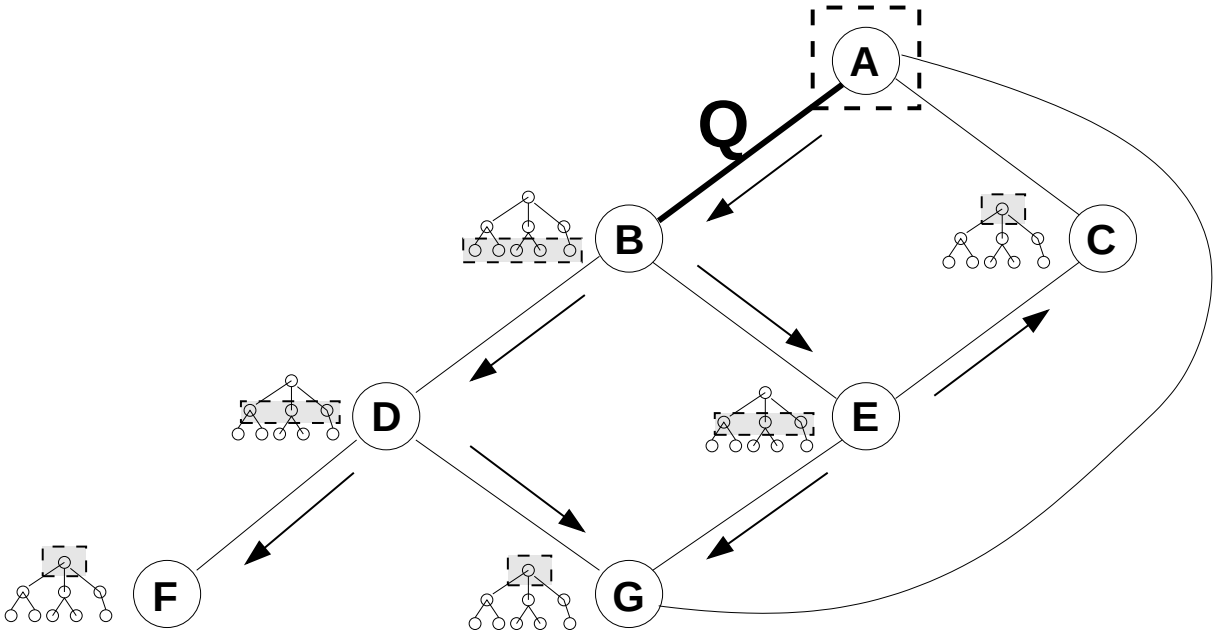

Figure 1: Peer A's view of the network through peer B. The level of the taxonomy describing A's view of each peer is highlighted.

Figure 1 shows how peer A sees the network through peer B, and which level of the taxonomy is describing each peer node. We can see that B is only one hop from A. Therefore, it is described by the bottom, and most detailed, level of the taxonomy. Peers D and E are two hops away from A via B. They are thus described by the second level of the taxonomy. Finally, peers C, F, and G are three hops away from A via B, and are described by the third level of the taxonomy.

On the other hand, Figure 2 illustrates how the same node A sees the network through its peer node C. In this case, node C is only one hop away from A and so it is described by the bottom level of the hierarchy. Node E is two hops away from A via C and it is described by the second level of the taxonomy. Peers B, and G are three hops away and are therefore summarized by the top level of the hierarchy. Finally, since peers D and F are more than three hops away from A via C, A has no knowledge about their collections.

In such a scenario it is important to have a scoring function such that given a query at a peer node, this peer could decide to which neighbour to forward the query in order to minimize the number of messages required to answer the query. We focus on this point throughout this paper.

Peers communicate via messages. An `Update` message is used to exchange information about the collections. On the other hand, queries are communicated via a `Query` message. All messages

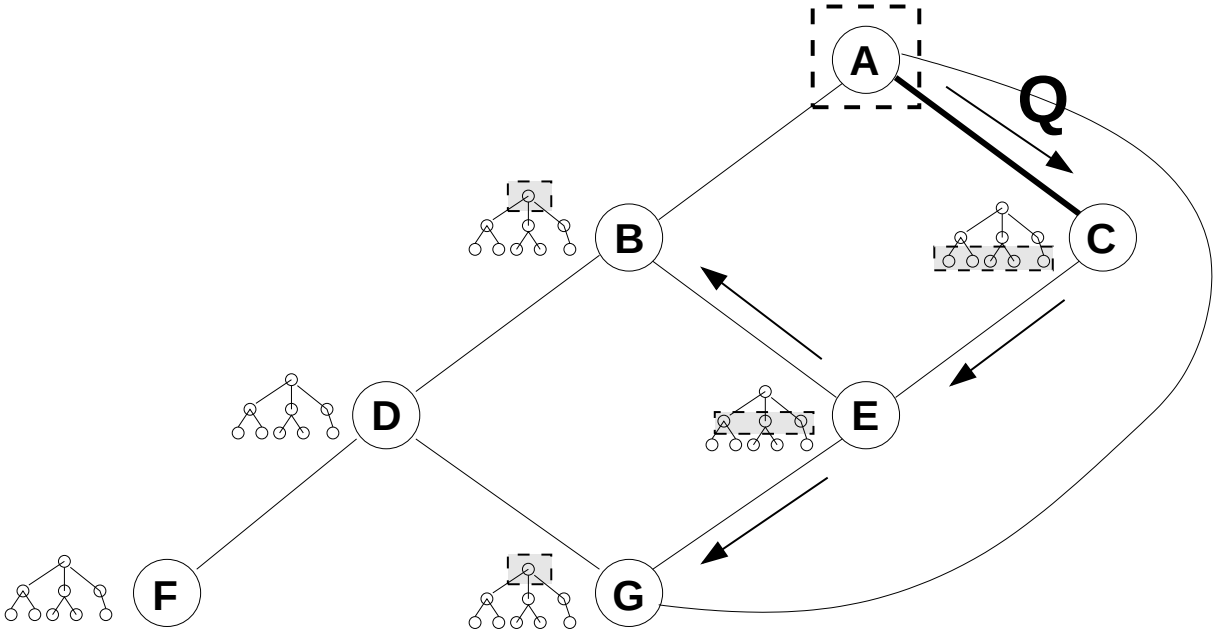

Figure 2: Node A's view of the network through its peer node C. The level of the taxonomy describing each peer is highlighted.

carry a unique conversation identifier that is used to identify cycles in the network and prevent updates and queries from circulating uncontrollably.

The `Update` message provides the means for the participants of the network to share information about the collections they can access. This information is then used to make informed routing decisions. The data sent with an update message contains information about all the collections that are accessible by the node. However, the update data must not take into consideration what is accessible via the recipient of the update. By choosing an appropriate height and bushiness of the taxonomy, the size of the updates and the distance that they are propagated can be controlled.

It is important to note that every peer sends updates that represent its own view of the network. Because of this, a peer receiving the information can easily make a routing decision because it "knows" what documents are available through each of its directly-connected peers.

A `Query` message is composed of a query string, a stop condition indicating the number of documents desired, and possibly a TTL field that limits the query's search radius. The node that initiates the query sends a `Query` message sequentially to its directly-connected peers until the desired result set size is reached or there are no more peers to query. Every node that receives a query message must reply with a message that identifies the query and reports the number of hits generated by that node.

If a peer cannot satisfy a query with the contents of its own collection, it will forward the message sequentially to its directly-connected peers in the same manner as the query initiator, until the desired number of results has been reached or there are no more peers to query. The query is not forwarded back to the peer who sent it to the node. Choosing an appropriate order in which to forward the query to the peers, the main goal of our research, is of utmost importance.

Cycles in the network create complications for the search and propagation of information. Updates in particular suffer because of cycles in the network. There is no risk of an update circling the network indefinitely because the update is propagated only once per level of the taxonomy. Also, messages are labelled with a unique conversation identifier allowing cycles to be detected and broken if they include the originator of the message. However, there exists a problem with over counting the documents available on the network if they are accessible via more than one path. Fortunately, this problem only affects cycles that are shorter than the height of the hierarchy.

Queries can easily deal with cycles by using the unique conversation identifier. Each peer keeps track of recently-evaluated queries by using the identifier, and thus avoids processing them more than once. From a practical perspective the conversation identifier does not need to be kept for long since queries are to be executed within a short lifespan. Hence this should not pose a significant overhead for a peer node.

3 The Minimum Expected Hop Count (MEHC) Score

As discussed earlier peer nodes query their neighbours sequentially. In the following we describe how to determine the sequence in which those neighbours are queried in order to minimize the number of messages exchanged, by using the taxonomical description of those neighbours.

Each peer has an attributed hierarchy for each connection that it has established with other nodes in the network. As previously mentioned, the hierarchy informs the peer about how many documents about a topic are accessible through the connection, and how far those documents are. However, the information becomes less precise as its distance increases.

We assume that documents in a peer node are distributed uniformly among the document topics, i.e., leaf nodes of the taxonomy. More specifically, given the taxonomy tree T of a given network node and a node n in such a tree, we define:

- *docs_in(n)* = the number of documents at node n (sub-nodes not included).
- *num_topics_under(n)* = number of topics (leaf nodes) under n. If n is a leaf node the result is defined as 1.

Therefore, given a taxonomy node (category) n the expected number of documents in any of its sub-topics (leaf nodes) is $\frac{docs\_in(n)}{num\_topics\_under(n)}$.

Recall that a query is a assumed to be a single leaf node of the taxonomy. Therefore, the score computation begins by finding the path $[n_1, n_2, n_3, \ldots, n_l]$ in the taxonomy tree from the queried leaf to the tree's root (note that l is the taxonomy's height). This path is used by the two parts of the scoring function described in the following paragraphs.

The first part of the scoring function consists of estimating the number of hops required by a neighbouring node to fulfill the query. This estimate is calculated by walking along the path in the taxonomy from the query to the root node, estimating the number of related documents available at each level, and stopping when the stop condition has been met. The number of hops required is then inverted to yield a score to be maximized. An algorithm reflecting this strategy is presented below.

While the routing methods introduced in [2] try to maximize the ratio of documents to query messages, we feel that this strategy may not always be best. Specifically, this routing method does not take into consideration the number of documents sought by the user (the stop

Algorithm 1 Estimating the number of hops required to fulfill a query.

$hopsToFulfill \leftarrow 0$
while $stopcondition > 0$ and $hopsToFulfill < taxonomy's\ height$ **do**
  $num\_related\_docs \leftarrow$
    $round(docs\_in(path[hopsToFulfill])$
     $/\ num\_topics\_under(path[hopsToFulfill])$
  $stopcondition \leftarrow stopcondition - num\_related\_docs$
  $hopsToFulFill \leftarrow hopsToFulFill + 1$
end while
return hScore = 1.0 / $hopsToFulFill$

condition). It could therefore route a query to a far node with more documents, while a node with *enough* documents is available near by. To avoid this issue, the Minimum Hop Count function tries to minimize the number of hops required to find only the number of documents required to satisfy the query. Further, while Crespo et al's heuristic strategy assumes that the network has the topology of a regular tree, the Minimum Expected Hop Count scoring strategy presented herein makes no assumptions about the network's structure.

The second component of the scoring algorithm incorporates into the score the estimated number of relevant documents available through a peer. The idea is that with more documents available it is more likely that there will be documents relevant to our query topic. However, it is not necessarily good to have this component of the score overwhelm the other since more documents is not always better; in fact it sometimes makes no difference. A good function will prioritize larger collections but should be bound by a an upper value. Likewise small collections should not be ignored completely. To take the number of documents available into account, the following score function was defined:

$$dscore = \frac{1}{2} \cdot tanh(\frac{n - s}{k * s})$$

where n is the estimated number of relevant documents
s is the stop condition (number of documents requested)
k is a scaling constant

The *tanh* function is asymptotical to 1 and -1. This trait has the advantage that an regardless of how big a collection may be, the amount of bonus it receives is still bounded. In essence, the functions rewards collections with some extra documents, but after a while stops paying attention. Similarly, it penalizes peers that have fewer relevant documents than are required to fulfill the query, but only up to a maximum. The scaling constant k is present to slow the transition of the function. Finally, the result is multiplied by $\frac{1}{2}$ to reduce the range to 1 and make it the same as the range for the minimum expected hop count function.

The two score functions described in the preceding sections are combined into one score in the following way:

$$score = \alpha \times hScore + (1 - \alpha) \times dScore$$

where *hScore* is the estimated hop count score
dScore is the document bonus or penalty
α is a bias parameter.

Using this score, a peer node can rank its neighbouring peers and sequentially query them for documents.

4 Experiments

The experiments conducted to evaluate this taxonomy-based routing scheme assume a static network. All connections between peers are established before the experiments start, and may not be dropped in the midst of the experiment. Also, new peers cannot join the network while an experiment is running. We deem this assumption acceptable since our goal is only to test the effectiveness of this query routing technique.

The performance of the routing scheme is evaluated with respect to the number of messages generated by a query. Since the experiments run on a static network, update messages are not exchanged throughout their course. Instead, each peer node sends update messages to its neighbours as part of the initialization procedure, thereby communicating the contents of its collection to them. The taxonomy hierarchy we assume within each peer node is that of the *ACM Computing Classification System* [9]. Despite it breadth, 739 topics, i.e., leaf nodes, it is only 5 levels high and 1487 nodes in total.

The topologies generated in our experiments fit the out-degree power law [4], and the nodes are assigned random collections. To evaluate the routing strategies, 3000 random queries are posed to the network from random nodes, and the number of query messages generated as well as the number of queries that the system was unable to fulfill are recorded. All experiments are repeated 10 times on different combinations of topologies and collections and their results are averaged. The baseline for comparison is obtained by having a peer node to forward the query sequentially to its neighbours in a random order. That is, the difference between the baseline and our method is simply how the query is routed.

The cost required to return the result set is not considered since the routing decision that minimizes the generated number of query messages will also minimize the cost of returning the result set (they are both a function of the number of hops between the originator of the query and the node with the results). The number of query reply messages is not counted for the same reason. Nevertheless, the performance of the baseline is measured in the exact same way as the performance of the routed query network and therefore the comparison is valid.

As a simplification, throughout the tests described in this document the queries are limited to the strings making up the leaf nodes of the hierarchy. This restriction implies that no two leaves in the hierarchy be labeled with the same string. To accommodate this restriction only the first instance of any given label was kept in ACM's hierarchy. Nevertheless, the system could be easily extended to support any node of the taxonomy as a query.

4.1 Simulation

There are several facets of the simulation that affect the evaluation of this query routing technique. To begin, the network has to be modelled. The simulation used for these experiments generated random, connected, network topologies that fit the out-degree power law [4]. It says that for an out-degree d, its frequency f_d, and a constant exponent o, the following proportion holds: $f_d \propto d^o$. Experiments were run with the out-degree exponent value $o = -1.4$. This value was reported by Jovanović et al. to be the exponent of a snapshot of the Gnutella P2P network taken in December 2000 [5].

The collections are modelled as follows. Each document is assumed to belong to exactly one of the leaf-level topics defined in the taxonomy. To generate a collection, a collection size is sampled from a distribution, and then the sampled number of documents are randomly distributed among the topics. The collection size is formed by the adding a minimum value m to the absolute value of a random number $r \sim N(0, \sigma)$. This approach was chosen because it is thought to resemble the way real collection sizes are distributed.

The simulator used for these experiments consists of a modified version of the PeerSim simulator [7]. PeerSim is aimed at simulating high-level communication protocols, so it does not concern itself with the low-level details of the communication network–e.g., TCP/IP stack, latencies, etc. The parameters of the simulation are summarized in Table 1.

Category	**Name**	**Value**
Query	Time to live	7, ∞
	stop condition	10
Topology	Out-degree exponent	-1.4
	number of nodes	2000
Collections	min size	70
	std dev σ	300
Scoring	alpha	0, 0.25, 0.5, 0.75, 1
	scaling const k	100

Table 1: Summary of simulation parameters.

4.2 Results

The first experiments were run to determine the best value for the parameter α that combines the Minimum Expected Hop Count and Document Bonus scoring functions. The experiments vary the value of α from 0.0 to 1.0 in increments of 0.25. We also vary the queries' TTL value between the value seven, which is the maximum value recommended for the Gnutella protocol [8], and infinity. The graph in Figure 3 summarizes the results of these tests, as well as the performance of our baseline sequential routing algorithm.

Note that by varying the parameter α we are in effect changing the influence of each of the two scoring functions on the routing decision. Recall that $\alpha = 0.0$ causes the scoring function to only use the Document Bonus component, while $\alpha = 1.0$ uses only the Minimum Expected Hop Count function.

Our first observation is that the routing scheme works better as α goes to 1.0, with a best score obtained when the routing decisions are made solely by the Minimum Expected Hop Count function. We therefore conclude that Document Bonus scoring function is ineffective. On the other hand, we find the Minimum Expected Hop Count function to be very effective. When compared to our baseline, on average the Minimum Expected Hop Count function performed 10 times better.

A second observation that the graph clearly shows is that using a TTL to limit the radius of the search penalizes the search performance. While such a limit is a practical necessity in a P2P system using the flood search scheme, artificially limiting the radius of a better informed search forces it to query network nodes with unrelated collections that that would have otherwise been ignored. It therefore generates more query messages per hit. Further, limiting the search

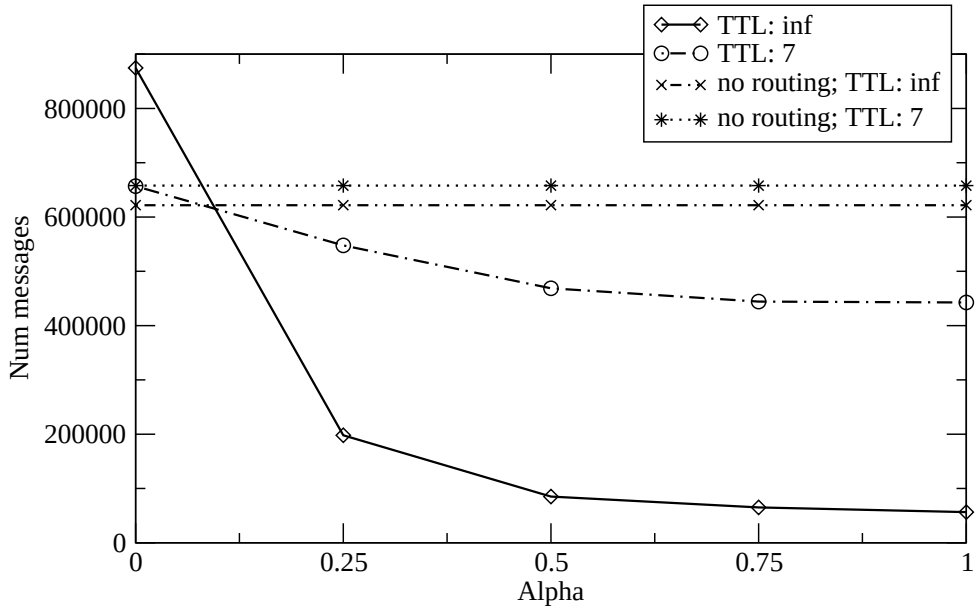

Figure 3: Effect of α in routing performance.

radius artificially limits the number of accessible collections. We found this factor to prevent the system from fulfilling a number of queries (about 5% on average), while still producing more query messages than an identical setup not using a TTL. In addition, all queries were completely satisfied by the routing scheme without a TTL. Therefore, at this point we set $\alpha = 1$, effectively dropping the *dScore* component of the MEHC score.

We also investigated whether the observation above would remain valid when varying the size of the collections at a peer node (Figure 4). As if can be seen the performance improves as the average size of the peers' collection increases, since it becomes easier to find the desired documents. More importantly, decreasing the collection sizes does not severely penalize performance. In fact, the figure suggests that for larger values of α the loss is linearly proportional to the reduction in the average collection size.

In addition, we observed an interesting phenomenon arising from the interaction between the attempt of the system to deal with cycles and the TTL. Let us explain it with the example in figure 5. Suppose there are two paths of different length from a query initiator node A to a node D. Now suppose that node A first routes its query along the longer path to D. The TTL of the query expires after the query passes D, but before the query is fulfilled. Therefore, A re-issues the query along the shorter path to D. However, when D receives the query it does not forward it because it has already seen it. Therefore, the contents of the network accessible through D are not explored to the fullest levels allowed by the TTL. We find evidence of this phenomenon while running multiple experiments while only varying the routing scheme. Intuitively, one would expect the same number of unfulfilled queries to be reported by each experiment. However, we found that the number of unfulfilled queries fluctuates depending on the particular routing decisions made by the algorithm. Although further research is necessary we believe this issue could be resolved by having the peers examine the TTL of a query in addition to its message identifier before deciding not to forward it.

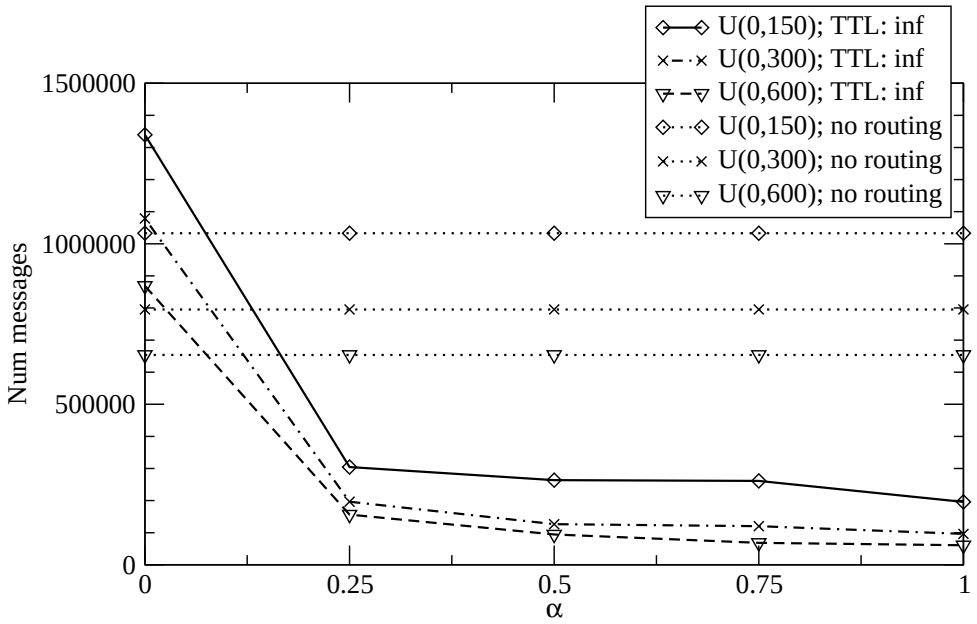

Figure 4: Effect of varying the collections sizes.

5 Conclusions

Peer-to-Peer networks have the potential to make use of their popularity to become effective information sharing tools, rather than only file sharing systems. However, to reach that level of functionality the systems need to adopt some type of content-based search, as well as improve their scalability and network usage by incorporating efficient network search methods. In this paper, we propose the use of a taxonomy data structure as part of an effective and efficient search method. In addition, we present the Minimum Expected Hop Count function that uses the taxonomy to estimate the number of hops required to fulfill a query. Our tests show that this function is an effective query routing heuristic, and can be used as part of a scalable information sharing P2P network. Finally, we show that while using a TTL value to limit the radius of a flood search is a practical necessity, artificially limiting the radius of an informed search method increases network usage and reduces the fraction of queries that the system is able to fulfill.

There are plenty of opportunities to extend this research in the future. The effects of varying the topology out-degree on the performance of this routing scheme should be studied. Also, a problem of interest consists of investigating the volume of update messages generated by this proposed routing scheme. Further, a comparison of Crespo et al.'s routing indices to the taxonomy-based routing scheme would be instructive. As well, it would be important to extend the techniques presented here by considering a dynamic environment where peer nodes can join and leave the network.

Acknowledgements

This research was partially supported by the National Science and Engineering Research Council (NSERC) and the Informatics Circle of Research Excellence (iCORE).

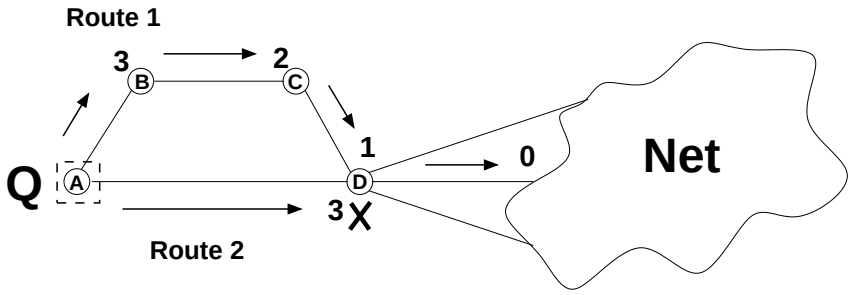

Figure 5: How TTL and cycle avoidance interact. The numbers show the query's TTL at each stage. The query routed through (A,D) is stopped because it has already been at D through node C.

References

[1] Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, pages 46–66, July 2000.

[2] Arturo Crespo and Hector Garcia-Molina. Routing indices for peer-to-peer systems. In *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS)*, pages 23–34, July 2002.

[3] Francisco Matias Cuenca-Acuna and Thu D. Nguyen. Text-based content search and retrieval in ad hoc P2P communities. Technical Report DCS-TR-483, Department of Computer Science, Rutgers University, 2002.

[4] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. In *Proceedings of ACM Special Interest Group on Data Communications (SIGCOMM)*, pages 251–262, 1999.

[5] Mihalo A. Jovanović. Modeling peer-to-peer network topologies through "small-world" models and power laws. In *TELFOR*, 2001.

[6] Jie Lu and Jamie Callan. Content-based retrieval in hybrid peer-to-peer networks. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 199–206, 2003.

[7] Peersim peer-to-peer simulator. `http://peersim.sourceforge.net/`.

[8] Rfc-gnutella 0.6. `http://rfc-gnutella.sourceforge.net/developer/testing/`, February 2004.

[9] The acm computing classification system [1998 version]. `http://www.acm.org/class/1998/`, 1998.