QPM: Phonetic Aware P2P Search

Md. Rakibul Haque and Reaz Ahmed
Dept. of Computer Science & Engg.
Bangladesh University of Engg. & Tech.
Dhaka-1000, Bangladesh
rakibul_haq@yahoo.com, reaz@buet.ac.bd

Raouf Boutaba
School of Computer Science
University of Waterloo
Ontario, Canada, N2L 3G1
rboutaba@bbcr.uwaterloo.ca

Abstract

Efficient discovery of information based on partially specified and misspelled query keywords is a challenging problem in large scale Peer-to-Peer (P2P) networks. This paper presents QPM, a P2P search mechanism for efficient information retrieval with misspelled and partial keywords. QPM uses the Double Metaphone algorithm to phonetically match misspelled query keywords with advertised keywords. For achieving bandwidth efficiency and similarity matching, QPM incorporates second order Reed-Muller code within the Plexus protocol having a logarithmic routing efficiency on overlay network size. QPM supports large scale networks and achieves better resilience to peer failure by maintaining redundant routing paths and by systematically placing index replicas. The concept presented in this paper is supported by necessary experimental evaluation.

1. Introduction

The query success rate in currently deployed P2P networks is very low, as pointed out in [6]. The reason behind this low success rate is not only the lack of knowledge about the advertised objects, but is also the rate of misspelled query keywords. According to the study in [6] about 25% queries in the Gnutella network contain at least one misspelled keyword. There are many reasons for which spelling mistakes may occur in query keywords, such as typing mistakes, inconsistencies between spelling and pronunciation in the English language, difference of spelling in different dialects of the English language etc.

Our proposed routing protocol *QPM* (Query with Partial and Misspelled keywords) can search expected files efficiently even if the query keywords are partially specified and misspelled. In *QPM*, advertised keywords are phonetically encoded and injected into separate Bloom filters [3] to create advertisement patterns (bit vectors). Query patterns are constructed in a similar manner from query keywords. During the query resolution process, all advertised patterns within a given Hamming distance of the query pattern are discovered. A second order Reed-Muller code is used to route messages to the rendezvous peers for resolving queries and storing advertised signatures. *QPM* reduces the impact of spelling mistakes in two ways: (a) phonetic encoding of keywords and (b) list decoding of advertisement(or search) patterns. The simulation results reveal that the proposed routing mechanism scales logarithmically with network size, provides high success rates even in presence of spelling mistakes and does not collapse due to frequent peer failures.

The rest of this paper is organized as follows. Section 2 presents related works. Section 3 presents the model of *QPM*. In Section 4, we present the performance evaluation of *QPM*. Finally, we conclude and outline our future research goals in Section 5.

2. Related Works

Structured search techniques, such as Chord [9], CAN [7] *etc*., map file identifiers to network locations using Distributed Hash Table (DHT)-based techniques. DHT-based techniques are not generally suitable for partial matching. A number of proposals, including [2], [8], and [10], have suggested additional layer on top of DHT routing to achieve partial matching capability. Squid [8] adopts space-filling-curves to map similar keywords to numerically close keys and uses Chord [9] for routing. pSearch [10] adopts CAN for routing and distributes document indices based on document semantics generated by Latent Semantic Indexing(LSI). In general, these extensions to the DHT routing support prefix matching only and require multiple DHT-lookups for query resolution. Some signature based techniques (like Plexus [1], NSS [5]) construct signatures (bit patterns) using keywords extracted from file descriptions or queries for routing. Although signature based search techniques support search with partial information with high performance, they fail to answer queries involving

978-1-4244-5067-1/09/$26.00 ©2009 IEEE

misspelled keywords. *QPM* addresses the issue of spelling mistakes in query keywords while securing high levels of bandwidth efficiency and query success rate.

3. Model of *QPM*

3.1. Core Concept

The basic concept of *QPM* is to advertise a file using associated keywords to a set of peers, say *A(P)* , and to search a query in another set of peers, say *S(Q)* , in such a way that at least one peer will be common in *A(P)* and *S(Q)* for serving the query if

$$A(P) \cap S(Q) \neq \emptyset \quad (1)$$

QPM has purely decentralized architecture. We assign each peer in the network one or more codewords from the second order Reed-Muller code RM2,7 [4]. Now the problem of finding *A(P)* and *S(Q)* is reduced to generating nearby codeword sets for the advertisement pattern P or the query pattern Q, respectively. We choose second order RM code because of its ability to support very large networks and simplicity of implementation.

QPM uses double metaphone as standard phonetic algorithm. Edit distance between the double metaphone codes of the advertisement and query keywords is less than the edit distance between the advertisement and query keyword. Moreover, list decoding of second order Reed-Muller code in computing *A(P)* and *S(Q)* helps to match the *A(P)* and *S(Q)* . In *QPM*, we satisfy the condition in Equation (1) by combining phonetic encoding technique with list decoding of RM(2, 7) code. We use both phonetic equity and edit distance as similarity measures during query resolution.

3.2. The Framework

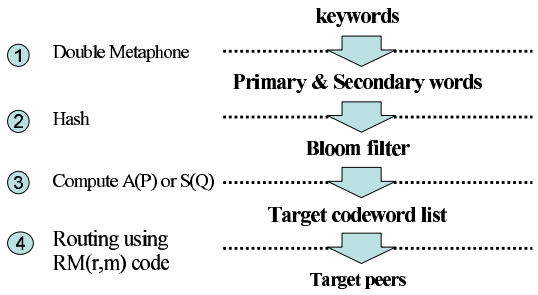


Figure 1. Framework

Advertisement and query routing mechanism in *QPM* consists of the following four steps as depicted in Figure 1.

1. Phonetic encoding: *QPM* extracts keywords from the file description or query and filter out the popular keywords to avoid hot-spot in the network. Then phonetically encoded words are generated from the selected keywords using the double metaphone algorithm.

2. Bloom filter construction: The double metaphone words are used to create Bloom filter P_{bf} (or Q_{bf}). Good advertisement and search patterns requires 20% to 50% bit to be 1 (see Figure 3(a)).

3. List decoding : In this step, the advertisement Bloom filter P_{bf} or the query Bloom filter Q_{bf} is mapped to target codewords *A(P)* or *S(Q)* for advertisement or query, respectively. The pseudocode for list decoding is presented in Algorithm 1. We compute *A(P)* by exact decoding of all the patterns within the Hamming sphere of radius d centered at the advertisement pattern $P_(bf)$ that have lower weights (i.e., number of 1-bits) than P. We compute *S(Q)* through exact decoding of all the patterns within the Hamming sphere of radius d centered at the query pattern Q_{bf} that have higher weights than Q_{bf}.

4. Routing : Routing in *QPM* is done by RM(2,7) code as explained in the next section.

Algorithm 1 Find $List(P, depth, type)$

Inputs:
  P : a 2^m bit pattern to be list decoded for advertising
  $depth$: current decoding distance form the original pattern
  $type$: advertisement or query
Externals:
  $maxDepth$: max. decoding distance form the original pattern
  $Decode(p)$: exact decoding algorithm for 2^{nd} order RM-code
$result \leftarrow Decode(P)$
if $depth < maxDepth$ **then**
  for each bit x_i in P **do**
    if x_i = 1 and $type$=advertisement **then**
      $P' \leftarrow P$ with i^{th}-bit reset to 0
    else if x_i = 0 and $type$=search **then**
      $P' \leftarrow P$ with i^{th}-bit set to 1
    end if
    $result \leftarrow result \cup FindList(P', depth + 1)$
  end for
end if
return $result$

3.3. Routing

The routing mechanism in *QPM* requires that each peer maintains $k + 1$ routing entries in its routing table, where k is the dimension of the RM-code in use. These $k + 1$ routing entries contain links to the peers responsible for the codewords $X_1, X_2, \ldots, X_{k+1}$ computed as follows:

$$X_i = \begin{cases} X \oplus g_i & 1 \leq i \leq k \\ X \oplus g_1 \oplus g_2 \oplus \ldots \oplus g_k & i = k+1 \end{cases} \quad (2)$$

Since we are interested in routing a message to the peers responsible for the list decoded codewords in *A(P)* or

S(Q) we require a overlay multicast routing mechanism. We adopt the multicast routing algorithm from [1].

RM(r,m) code has exactly 2^k codewords, where k is the dimension of the code and the generator matrix G of the code has k rows. Now, we define a k-bit ID to identify the 2^k codewords in RM(r, m). The i^{th} bit of the ID for a codeword will be 1 if the i^{th} row of G (*i.e.*, g_i) is required to construct that codeword. We use this ID to partition the codewords into a logical binary tree of height at most k as in Plexus [1] to map the codewords to the peers. The join and recovery processes in *QPM* are similar to those processes in the Plexus[1] network.

4. Simulation Result and Evaluation

In this section, we present the experimental results obtained by simulating the *QPM* framework. In this experiment we have varied network size, failure rate and the edit distance between advertised and queried keywords to measure the scalability, robustness and spelling mistake correction ability of *QPM*.

We have developed a simulator using Java to evaluate *QPM* on various aspects. We have simulated a growing network, which is grown from an initial set of nodes. The overlay network has a growing phase and a steady state phase to achieve different network sizes. During the growing phase, arrival rate is kept higher than departure rate (up to five times) until the network reaches a target size and enters the steady state phase. During this phase, peer departure and arrival rates remain almost equal. Advertisements and queries are performed during the steady state phase to measure the performance metrics as mentioned above.

Our major contribution in this paper is the search in presence of spelling mistakes in keywords. From Figure 2(a), we see that *QPM* has greater than 90% success rate for even up to edit distance three. From the figure, it is observed that search completeness is very close to 100% for exact match keywords for all network sizes. Search completeness remains almost constant at around 98% as the network is grown from 2000 peers to 20000 peers which is for edit distance one to three. This implies that *QPM* is scalable and successful for queries with misspelled keywords. Here, search completeness slightly decreases with the increase of edit distance, because Hamming distance increases with the increase of edit distance. By double metaphone encoding we reduce the impact of edit distance on Hamming distance. We have estimated the search completeness with the following function: $Searchcompleteness = \frac{100}{N} \sum_{i=0}^{N} \frac{d_f(i)}{d_a(i)}$

Here, N is the total number of keywords used for searching. $d_a(i)$ is the actual number of documents with the i^{th} keyword and $d_f(i)$ is the number of documents from $d_a(i)$ as returned by the search mechanism.

QPM requires at most $\frac{k}{2}$ hops for routing a message from any source peer to any destination peer, where k is the dimension of the code (here $k = 29$). Figures 3(d) and 2(b) depict that *QPM* requires to visit small percentage of peers for the search and advertisement process. It is also evident that the percentage of visited peers declines as the network size increases. Moreover, the impact of increasing edit distance is not significant on search efficiency.

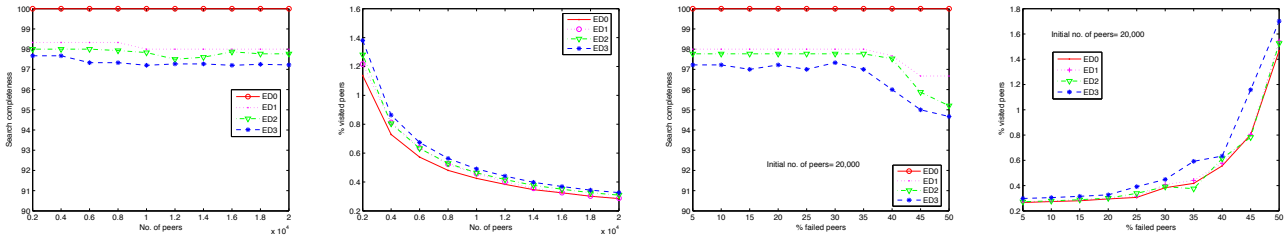
We have measured load of the peers in the simulated network. Each file is advertised to multiple peers along with their replicas. As depicted in Figure 3(c), variation in average load on each peer is very low and decreases with increasing network size, which implies uniform distribution of load over the participating peers in the network.

To measure the join overhead in the growing network simulated in this experiment, we recorded the number of messages exchanged for finding local minima, failed neighbors and routing table entries during the join process at different network sizes. Figure 3(b) presents the average percentage of visited peers during a join operation for different network sizes. Scalability of the join process is evident form Figure 3(b), since the percentage of visited peers decreases significantly as the network size increases.

In this experiment, we measure the robustness of *QPM* in presence of simultaneous peer failure in a simulated network of 20,000 peers. Figure 2(c) presents search completeness and figure 2(d) provides query routing traffic in case of various levels of peer failures. From Figure 2(c), we find that search completeness remains at almost constant levels of 100%, 98%, 97.8% and 97.2% for edit distances zero, one, two and three, respectively for up to 35% peer failures. From Figure 2(d), it can be seen that the percentage of visited peers does not increase significantly below 35% failure levels. This indicates that the network can successfully and efficiently route queries below 35% failure levels. As the failure level increases beyond 35%, the percentage of visited peers increases and search completeness decreases. The reason behind this behavior is that many hops are wasted due to the failure of intermediate peers on a routing path and that of the target peers. However, for exact queries (*i.e.*, with edit distance 0) search completeness does not decline even if the failure level rises above 35%. In case of exact matching, the Hamming spheres for the query pattern and advertisement pattern are identical and all the codewords in *A(P)* and *S(Q)* coincide. Hence, an exact match query will fail only if all the peers responsible for the codewords in *A(P)* (or *S(Q)*) fail, which will happen with a very low probability.

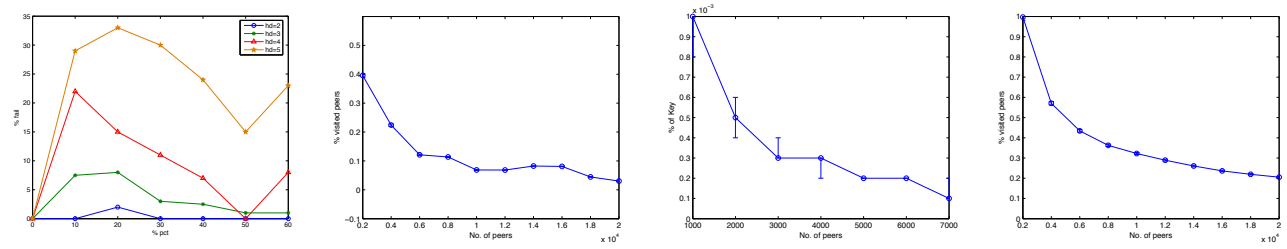
5. Conclusion and Future work

In this paper we have presented *QPM*, which is an efficient P2P search mechanism capable of handling misspelled



(a) Search Completeness (b) Search Efficiency (c) Search Completeness (Fault Resilience) (d) Search Efficiency (fault Resilience)

Figure 2. Search Completeness and Search Efficiency



(a) Good Advertisement/Search Pattern (b) Join Overhead (c) Load Distribution (d) Advertisement Efficiency

Figure 3. Join Overhead, Load Distribution and Advertisement Efficiency

keywords in query. *QPM* is a purely decentralized, structured search technique based on RM(2, 7) code and double Metaphone encoding. Routing in *QPM* scales logarithmically with network size and remains stable in the presence of peer failures. Search completeness in *QPM* is high. Experimental results show that *QPM* achieves search completeness levels of 100% for zero edit distance and 97% for an edit distance of 3 by visiting around 0.4% peers in a network of 20000 peers. Below 35% peer failure rate, *QPM* retains these search completeness levels though visited peers rises to 0.6%.

We intend to extended *QPM* to a multi-level routing framework by combining 1^{st} and 2^{nd} order RM codes in order to exploit the heterogeneity in peers' capability. We also intend to deploy the protocol on a distributed testbed, like PlanetLab, to gain indepth knowledge of the real life performance of *QPM*.

References

[1] R. Ahmed and R. Boutaba. Plexus: A scalable peer-to-peer protocol enabling efficient subset search. *IEEE/ACM TON*, 17(1):130–143, February 2009.

[2] M. Bawa, T. Condie, and P. Ganesan. LSH forest: Self-tuning indexes for similarity search. In *in Proc. of WWW Conference*, May 2005.

[3] B. H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Comm. of ACM*, 13(7):422–426, 1970.

[4] C. L. K. P. C. R. D.G. Hoffman, D.A. Leonard and J. Wall. Coding Theory;The Essentials.

[5] M. Li, W. Lee, and A. Sivasubramaniam. Neighborhood signatures for searching P2P networks. In *Proc. of IDEAS*, pages 149–159, 2003.

[6] S. S. M. A. Zaharia, A. Chandel and S. Keshav. Finding content in file-sharing Networks When you Can't Even Spell. In *Int. workshop on Peer-To-Peer Systems*, Feb. 2007.

[7] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A scalable content-addressable network. In *ACM SIGCOMM*, pages 161–172, 2001.

[8] C. Schmidt and M. Parashar. Enabling flexible queries with guarantees in P2P systems. *IEEE Internet Computing*, 8(3):19–26, June 2004.

[9] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: a scalable Peer-to-Peer lookup protocol for Internet applications. *IEEE/ACM TON*, 11(1):17–32, 2003.

[10] C. Tang, Z. Xu, and M. Mahalingam. pSearch: information retrieval in structured overlays. *ACM SIGCOMM Computer Communication Review*, 33(1):89–94, 2003.