A New Search Mechanism for Unstructured Peer-to-Peer Networks

Chia-Hung Lin and Sun-Yuan Hsieh

Abstract—In a traditional file search mechanism, such as "flooding," a peer broadcasts a query to its neighbors through an unstructured Peer-to-Peer (P2P) network until the Time-To-Live (TTL) decreases to zero. A major disadvantage of flooding is that, in a large-scale network, this blind-choice strategy usually incurs an enormous traffic overhead. In this paper, we propose a method, called the Statistical Matrix Form (SMF), which improves the flooding mechanism by selecting neighbors according to their capabilities. The SMF measures the following peer characteristics: the number of shared files, the content quality, the query service, and the transmission distance between neighbors. Based on these measurements, appropriate peers can be selected and thereby reduce the traffic overhead significantly. Our experimental results demonstrate that the SMF is effective and efficient. For example, compared with the flooding search mechanism in dynamic unstructured P2P networks, the SMF reduces the traffic overhead by more than 80 percent. Moreover, it achieves a good success rate and shorter response times.

Keywords—Unstructured peer-to-peer networks, flooding search mechanism, traffic overhead, statistical matrix form.

I. INTRODUCTION

Generally, Peer-to-Peer (P2P) networks can be classified as: structured P2P networks, which are based on centralized management (e.g., Chord [10]), and unstructured P2P networks, which are built on a distributed search mechanism (e.g., Gnutella [1]). Although both types allow users to participate in a fully distributed cooperative network, unstructured P2P networks give participants more freedom to exchange resources and services. The major disadvantage of unstructured P2P networks is that their basic search mechanism, "flooding," incurs an enormous traffic overhead. To resolve this issue, numerous search mechanisms have been proposed to replace or improve the flooding mechanism [3], [6], [7]. In this paper, our object is to improve the flooding mechanism by exploiting the scalability of unstructured P2P networks.

In a real network environment, peers differ from each other in a number of respects, such as the number of shared files, the content quality, the query service and transmission distance between neighbors. These characteristics are crucial because they can be utilized to optimize the search performance effectively. We propose a method that statistically analyzes query messages in terms of the following four characteristics: Processing Ability (PA), Effective Sharing (ES), Index Power (IP), and Transmission Efficiency (TE). The PA of peers is analyzed to determine which peers leech the most resources without giving feedback [2], [4]. The ES refers to the number of files that a peer shares, and can be used to classify a peer's sharing capability. It has been shown that, in a network, very few peers share a large number files, so that the quality of the files influences the sharing capability [9], [14]. The IP measures the number of files that a peer records in the index cache, and can also be used to analyze the number of responses in the cache content. Finally, the TE is utilized to measure the distance between peers in order to prevent inefficient routing. Xiao et al. [15] observe that only a small number of links connect peers in the same Autonomous System (AS). Most generated connections stay past AS borders to produce more distant links.

We represent the four characteristics in a matrix form called the Statistical Matrix Form (SMF). To adjust the values of the matrix, we utilize a standard deviation technique to determine an overall ranking of a query peer's neighbors. As a result, the performance of the flooding search mechanism can be improved by only sending query messages to the top-k ranked neighbors of a query peer for some k determined by careful analysis, instead of sending messages to all the peer's neighbors. The performance evaluation demonstrates that the response time and traffic overhead can be reduced significantly, while the computation overhead is acceptable.

II. THE SEARCH MECHANISM BASED ON THE SMF

The SMF of query peer u is comprised of two matrices: the left-hand matrix and the right-hand matrix. The left-hand matrix, called the feature matrix (FM), is an \( n \times 4 \) matrix, where \( n \) is the number of neighbors of \( u \). To derive the entries for the four columns of the FM, we compute the PA, ES, IP, and TE scores for \( n \) neighbors of \( u \) by the method described in Subsections A-E; then, we record the computed scores in the first, second, third, and fourth columns respectively. The right-hand matrix, called the weight matrix (WM), is a \( 4 \times 1 \) matrix in which each peer can set the proper weights according to the derivation degree of each feature. The method is described in Subsection F. Finally, each query peer \( u \) computes a scoring matrix (SM), which is an \( n \times 1 \) matrix obtained by the...
matrix multiplication $FM \times WM$. To deliver queries for $u$, we obtain the score of each of $v$'s neighbors in the $SM$ and then select the neighbors with the top-$k$ scores to send query messages. Fig. 1 illustrates the structure of SMF. Since the query peer $u$ has five neighbors, $v$, $w$, $x$, $y$, and $z$, its feature matrix is a $5 \times 4$ matrix; the weight matrix is a $4 \times 1$ matrix; and the score matrix is computed by the formula $SM = FM \times WM$, which is a $5 \times 1$ matrix.

A. Feature Collection Scope

We define the $d$-collected scope of a query peer $u$ as the set of peers that are at most $d$ hop(s) away from $u$. In the construction of the $FM$ presented in the following subsections, we collect relative information about the $d$-collected scope of a query peer. For example, the information about the 2-collected scope of a query peer $u$ contains the information about each neighbor $v$ of $u$, as well as that of $v$'s neighbors. We could improve the search performance by increasing the value of $d$; however, it may increase the computational overhead because of the extra cost of collecting information. Therefore, the problem is how to choose appropriate values of $d$ to improve the performance by determining the acceptable extra-overhead incurred by the collection and exchange of information. We evaluated that $d = 2$ or $d = 3$ are the optimal values; hence, we adopt $d = 2$ in the remainder of this paper.

B. Processing Ability (PA)

In P2P networks, there are usually free loaders [2] who download files without sharing any of their resources, which impacts the search performance of coadjutant communities. To prevent free loaders, we utilize the $PA$ to differentiate between leeching and enthusiastic peers. The $PA$ score is computed in terms of the peers’ query frequency and response frequency.

In a P2P network, a query peer that generates a lot of queries may be a free loader. Let $N(u)$ be the neighbors of a query peer $u$; that is, $N(u)$ are peers that are one hop away from $u$. In addition, let $NQ(v)$ be the number of queries sent by $v$. Each query peer $u$ computes $SQ(u)$, which is the total number of queries ($SQ$) sent from the peers that are one hop away from $u$. Formally,

$$SQ(u) = \sum_{v \in N(u)} NQ(v).$$

(1)

The Query-Minus-Score ($QMS$) of a neighbor $v$ of $u$ is defined as

$$QMS(u, v) = SQ(u) - NQ(v).$$

(2)

When $NQ(v)$ increases, the possibility of $v$ being regarded as a free loader also increases and peer $v$ will be assigned a lower score. Next, each query peer $u$ computes $SQMS_1(u)$ (resp. $SQMS_2(u)$), which is the sum of the query-minus-scores ($QMS$) of all peers that are one (resp. two) hop(s) away from $u$:

$$SQMS_1(u) = \sum_{v \in N(u)} QMS(u, v)$$

(3)

and

$$SQMS_2(u) = \sum_{v \in N(v)} SQMS_1(v).$$

(4)

Then, the query frequency of a neighbor $v$ of $u$ is defined as

$$QF(u,v) = w_1 \cdot \frac{QMS(u,v)}{SQMS_1(u)} + w_2 \cdot \frac{SQMS_2(u)}{SQMS_3(u)},$$

(5)

where $w_1$ and $w_2$ are two parameters used to adjust the influence of peers that are one hop away and two hops away from $u$ respectively. Based on (5), a peer can determine the amount of resources that their neighbors leech from the network.

If a peer responds to a large number of queries, we define it as an “eager” peer. The term “response frequency” refers to a peer’s ability to respond to queries. Each peer $u$ computes $SR_1(u)$ (resp. $SR_2(u)$), which is the sum of the response times ($SR$) of peers that are one (resp. two) hop(s) away from $u$. Formally,

$$SR_1(u) = \sum_{v \in N(u)} NR(v),$$

(6)

where $NR(v)$ is the number of responses sent by peer $v$, and

$$SR_2(u) = \sum_{v \in N(v)} SR_1(v).$$

(7)

Based on the above two equations, the response frequency of a neighbor $v$ of $u$, denoted by $RF(u,v)$, is computed as follows:

$$RF(u, v) = w_1 \cdot \frac{NR(v)}{SR_1(u)} + w_2 \cdot \frac{SR_1(v)}{SR_2(u)}.$$  

(8)

From $RF(u,v)$, we can determine the response ability of a neighbor $v$.

We represent the processing ability of a neighbor $v$ of $u$, denoted by $PA(u,v)$, in terms of the query frequency and the response frequency as

$$PA(u,v) = QF(u,v) + RF(u,v).$$

(9)

C. Effective Sharing (ES)

This feature is based on the observation in [9] and [14] that the file-sharing among peers is extremely unbalanced. For example, it has been shown that 7 percent of peers in a P2P network share more files than all the other peers can provide, and the top-1 percent of peers respond to 47 percent of the queries. Instead of all peers participating in file-sharing, only a small number of volunteer peers provide most of the resource sharing services in a P2P network. Moreover, peers’ query response capabilities vary because of the heterogeneity of their file-sharing resources. The trace analysis in [11] showed that a small number of peers share a large number of files. Because query answering involves matching keywords with the names of all shared files, we posit that, as the number of shared files increases, the probability of successful matching should also increase. Based on the above observations, we propose the concept of effective sharing (ES), which is used to determine the number of files shared among peers in a P2P network. The ES is comprised of two sub-features: the sharing count (SC) and the quality of sharing (QS).
When choosing influential neighbors to send queries from a query peer, it is necessary to consider the number of files shared by the peers. In a real environment, if a peer shares a large number of files, it should have a higher probability of matching queries than a peer that only shares a few files. Each query peer $u$ computes $SF_1(u)$ (resp. $SF_2(u)$) which is the total number of shared files (SF) by peers that are one (resp. two) hop(s) away from $u$. Formally,

$$SF_1(u) = \sum_{v \in N(u)} NF(v), \quad (10)$$

where $NF(v)$ is the number of shared files, and

$$SF_2(u) = \sum_{v \in N(u)} SF(v). \quad (11)$$

The **sharing count of a neighbor** $v$ of $u$ is defined as

$$SC(u,v) = w_1 \cdot \frac{NF(v)}{SF_1(u)} + w_2 \cdot \frac{SF(v)}{SF_2(u)}. \quad (12)$$

Based on (12), if the $SC$ score of some neighbor $v$ of a query peer $u$ is the largest among all the neighbors, then it is regarded as the most influential neighbor. The reason is that $v$ shares more files than the other neighbors, so there is a much higher probability that $u$ will get responses from $v$.

It has been shown that some shared files are never used to answer queries [2]. If we only consider the number of files used to answer queries, the number of files shared with query peers has a strong correlation with the responding peers. Motivated by this fact, we utilize the second $ES$ sub-feature, the “quality of sharing,” to distinguish useful files from useless files. Let $NFH(v)$ be the number of $v$'s shared files that match queries. Each query peer $u$ computes $SFH_h(u)$ (resp. $SFH_2(u)$), which is the effectiveness of the neighbors of a query peer $u$, by summing the $NFH$ values of the peers that are one (resp. two) hop(s) away from $u$. Formally,

$$SFH_1(u) = \sum_{v \in N(u)} NFH(v), \quad (13)$$

and

$$SFH_2(u) = \sum_{v \in N(u)} SFH_1(v). \quad (14)$$

The **quality of sharing of a neighbor** $v$ of $u$ is defined as

$$QS(u,v) = w_1 \cdot \frac{NFH(v)}{SFH_1(u)} + w_2 \cdot \frac{SFH_1(v)}{SFH_2(u)}. \quad (15)$$

Based on (15), if the $QS$ score of some neighbor $v$ of a query peer $u$ is the largest among all the neighbors, then $v$ is regarded as the most influential neighbor because it has the most opportunities to share files with the query peer $u$.

We then represent the effective sharing of a neighbor $v$ of $u$, denoted by $ES(u,v)$, in terms of the sharing count and the quality of sharing as follows:

$$ES(u,v) = SC(u,v) + QS(u,v). \quad (16)$$

### D. Index Power (IP)

In a P2P network, volunteers have different-sized indexes to record historical information. However, if a peer records a large number of file sharing messages in its index, many of the messages may never be used by the mechanism. The **Index Power (IP)** feature determines the amount of content in a queried peer's index and assesses its quality. The IP comprises two sub-features: **Index Counting (IC)** and **Quality of the Index (QI)**.

The index count feature records the number of messages in a peer's index. We assume that if a peer records a large amount of information in its index, it will have a higher probability of matching queries. Each query peer $u$ computes $SI_1(u)$ (resp. $SI_2(u)$) which is the number of indices of the peers that are one (resp. two) hop(s) away from $u$.

$$SI_1(u) = \sum_{v \in N(u)} NI(v), \quad (17)$$

where $NI(v)$ is the number of index records in peer $v$, and

$$SI_2(u) = \sum_{v \in N(u)} SI_1(v). \quad (18)$$

Then, the index count of a neighbor $v$ of $u$ is calculated as follows:

$$IC(u,v) = w_1 \cdot \frac{NI(v)}{SI_1(u)} + w_2 \cdot \frac{SI_1(v)}{SI_2(u)}. \quad (19)$$

Based on (19), if the IC score of some neighbor $v$ of a query peer $u$ is the largest among all the other neighbors, then $v$ can be regarded as the most influential neighbor because it has the largest possibility to reply the query sent from the query peer $u$ via $v$'s index.

The second sub-feature analyzes the quality of an index's content and the characteristics of the files. Since the probability of index hits may be influenced by the index counts as well as the quality of the index's content, we count the number of index hits to analyze the quality of the information in the index. Each query peer $u$ computes $SIH_1(u)$ (resp. $SIH_2(u)$) which is the total number of index hits of peers that are one (resp. two) hop(s) away from $u$.

$$SIH_1(u) = \sum_{v \in N(u)} NIH(v), \quad (20)$$

where $NIH(v)$ is the number of index hits of a neighbor $v$ of $u$, and

$$SIH_2(u) = \sum_{v \in N(u)} SIH_1(v). \quad (21)$$

Based on (20) and (21), QI($u,v$) can be computed as follows:

$$QI(u,v) = w_1 \cdot \frac{NIH(v)}{SIH_1(u)} + w_2 \cdot \frac{SIH_1(v)}{SIH_2(u)}. \quad (22)$$

Based on (22), if the QI score of some neighbor $v$ of a query peer $u$ is the largest score, then $v$ is regarded as the most influential neighbor for index hits, i.e., it has the highest probability of responding to the query sent by $u$.

We then represent the index power of a neighbor $v$ of $u$, denoted by IP($u,v$), in terms of the index count and the quality of the index as follows:

$$IP(u,v) = IC(u,v) + QI(u,v). \quad (23)$$

### E. Transmission Efficiency (TE)

This feature considers the transmission distances in a P2P network, as shown in Fig. 2. In the figure, peer $S$ must choose a neighbor to dispatch a query to neighbor $A$ or neighbor $C$. Peers $A$ and $S$ are in the same autonomous system, but peer $C$ belongs to another system. We assume that the transmission path...
between $S$ and $C$ is much longer than that between $S$ and $A$. Hence, $S$ will choose the path to $A$.

The $TE$ feature calculates the distances between a peer and its neighbors so that the peer can choose the closest neighbors to deliver a message. Each query peer $u$ computes $SLD_1(u)$, which is the sum of the link-distances of peers that are one hop away from $u$. Formally,

$$SLD_1(u) = \sum_{v \in N(u)} LD(u, v),$$

where $LD(u, v)$ is the link-distance between $u$ and $v$. The Link-Minus-Score (LMS) of a neighbor $v$ of $u$ is defined as

$$LMS(u, v) = SLD_1(u) - LD(u, v).$$

According to the above equality, when $LD(u, v)$ increases, the distance between peers $u$ and $v$ also increases; thus, peer $v$ will be assigned a lower score. Next, we compute $SLMS_1(u)$ (resp. $SLMS_2(u)$), which is the sum of the link-minus-scores of peers that are one (resp. two) hop(s) away from peer $u$. Formally,

$$SLMS_1(u) = \sum_{v \in N(u)} LMS(u, v)$$

and

$$SLMS_2(u) = \sum_{v \in N(u)} SLMS_1(v).$$

The transmission efficiency of a neighbor $v$ of peer $u$ is defined as

$$TE(u, v) = w_1 \times \frac{LMS(u, v)}{SLMS_1(u)} + w_2 \times \frac{SLMS_1(v)}{SLMS_2(u)}.$$  

Based on (26) and (27), the scores of peers that are at most two hops away from $u$ are computed. Then, the transmission efficiency of the neighbors of $u$ can be measured and we can determine the quality of each link $(u, v)$.

**F. Weight Matrix**

In this section, we introduce the weight matrix, which is incorporated into the feature matrix described in the previous section. It is used to normalize the values of the feature matrix. In the distribution of the data set in the feature matrix, if the values of a feature are widely dispersed, we can set a higher value in the weight matrix to represent the diversity of the feature. Conversely, if the values of a feature are close to each other, we set a lower value in the weight matrix to reduce the influence of the feature. We adopt the mean and standard deviation techniques to achieve this goal.

Let $\{x_1, x_2, x_3, \ldots, x_n\}$ be a data set. First we define the terms.

**Definition 1.** The mean, which is the arithmetic average of a set of values or distribution, is given by

$$\bar{x} = \frac{1}{N} \sum_{i=1}^{n} x_i.$$  

**Definition 2.** The standard deviation is a quantity that describes the spread of the data set from the mean. It is given by

$$S = \sqrt{\frac{1}{N-1} \sum_{i=1}^{n} (x_i - \bar{x})^2}.$$  

Before computing the standard deviation of each column in the feature matrix, we compute the mean value of each column as follows:

$$\bar{C}_m = \frac{1}{n} \sum_{j=1}^{m} FM(j, m),$$

where $n$ is the number of neighbors of peer $u$; and $FM(j, m)$ is the value of the entry in the $j^{th}$ row and $m^{th}$ column, i.e., $(j, m)$-entry, of the feature matrix $FM$. We compute the standard deviation $S_m$ for $1 \leq m \leq 4$ as follows:

$$S_m = \sqrt{\frac{1}{n-1} \sum_{j=1}^{m} (FM(j, m) - \bar{C}_m)^2}, \ n \geq 2.$$  

Then, we define the $4 \times 1$ weight matrix $WM$ as

$$WM(m, l) = \frac{S_m}{\sum_{j=1}^{4} S_j},$$

where $WM(m, l)$ is the weight of the $(m, l)$-entry.

**G. Scoring Matrix**

Each query peer $u$ constructs an $n \times 1$ scoring matrix ($SM$) by the multiplication of the feature matrix and the weight matrix as follows:

$$SM(m, l) = \sum_{k=1}^{4} (FM(m, k) \ast WM(k, l)),$$

where $SM(m, l)$ for $1 \leq m \leq n$ is the $(m, l)$-entry of $SM$ and $n$ is the number of neighbors of $u$.

Based on the scoring matrix, each query peer $u$ can assess the capability of each of its neighbors. To deliver query messages, peer $u$ selects $k$ influential neighbors according to the top-$k$ values in the scoring matrix. The proper value of $k$ is determined by the parameters of a P2P network e.g., the scale of the network, the number of neighbors of a query peer, and the requirement of a query peer. Moreover, it has been shown that, in a real P2P network, the number of neighbors of a query peer is usually 6 at most [8], which implies that the scoring matrix should not be too large. As a result, the performance of the proposed search mechanism can be optimized because the memory requirement is small.
III. PERFORMANCE EVALUATION

A. Simulation Environment and Performance Metrics

In this section, we present the experimental results, which demonstrate the effectiveness of the proposed search mechanism SMF. We construct two types of network topology, a physical topology and a logical topology. The physical topology is comprised of the real connections in a large-scale network; and the logical topology is the P2P transmission layer, which is built on top of the physical topology connections. All the peers in the logical network are chosen from the physical network. We compare the search efficiency of SMF with that of other search mechanisms, namely Flooding (FL), Random Walk (RW), and Multiple Random Walk (MRW) in terms of the following four criteria:

- **Traffic Cost**: the average number of messages sent per query; it is used to measure the search efficiency.
- **Response Time**: the query response time is the interval between the time a query is initiated and the time the result is received when each query explores the matching response in the P2P network.
- **Query Success Rate**: the percentage of queries that receive at least one response during the search process.
- **Searching Hop in Query Hit**: the average number of steps required to send a query if the query is responded to in our experiments.

Because of the restriction on page limit, we only present and discuss the evaluation result for the experiments in a dynamic environment.

B. Experiments in a Dynamic Environment

As shown in Fig. 3(a) and 3(b), SMF is more effective than FL, which incurs an enormous traffic overhead in order to increase the number of opportunities to match files during the search process. Meanwhile, Fig. 3(b) shows that SMF also has a higher probability of obtaining responses. Thus, SMF has the advantages of FL, but its traffic overhead is much lower. Comparison of MRW and SMF shows that SMF's search cost is lower, even though it selects the same number of neighbors to send a query. Finally, although RW has the lowest traffic overhead, its success rate is the lowest. Thus, overall, SMF achieves an effective search performance.

In Fig. 3(c), the traditional search mechanism FL has the shortest response time. This is because it delivers queries to all of a peer's neighbors. SMF's response time is a little higher than that of FL, but it reduces the traffic overhead significantly. Although the number of neighbors selected by SMF to send a query is the same as that of MRW in the experiment, SMF performs better than MRW. Hence, SMF can achieve a shorter query response time in dynamic environments.

Fig. 3(d) shows that SMF's performance is a little worse than that of FL, but it outperforms the other two search mechanisms. Although SMF does not send messages to all the neighbors of a query peer, it can still get query responses within an acceptable search scope.

In addition to the features compared above, SMF achieves a better performance through the warm-up phase avoiding. As shown in Fig. 3(a) – 3(d), SMF's performance is not satisfactory...
in the initial stage; however, as the number of queries increases, the search performance improves. The reason is that the matrix form yields more precise computations as the number of queries increases. Since communications in a real P2P network are continuous, we can obtain accurate information about each peer via the matrix.

IV. CONCLUDING REMARKS

We have performed simulations of the proposed SMF search mechanism in static and dynamic environments. The results demonstrate that SMF can reduce the traffic overhead significantly, achieve shorter query responded times, and maintain a high success rate. Specifically, SMF performs more than 80 percent better than the flooding approach in terms of the traffic overhead. Compared to the multiple random walk approach, SMF’s response times and success rate are 40 percent and 20 percent better respectively. The experimental results also show that each peer can determine the capability of each neighbor peer and send messages to the appropriate number of its neighbors to avoid redundant messages. Therefore, SMF is an effective search mechanism for P2P networks.

SMF can also be utilized in different applications, e.g., to adjust the connections for clustering influential peers, such as the clustering approach proposed in [12], or to reconstruct a topology, like the minimal spanning tree technique used in [5] and [13]. By utilizing the statistical matrix form, we can obtain more precise information and further improve the search performance in P2P networks.

REFERENCES


Chia-Hung Lin received the B.S. degree in the Department of Information Management from National Taiwan University of Science and Technology, Taiwan, in 2001. He received the M.S. degree in the Department of Computer Science and Information Engineering from National Cheng Kung University, Taiwan, in 2004. He is currently working toward the PhD degree in computer science and information engineering at National Cheng Kung University, Taiwan. His research interests include parallel and distributed computing, peer-to-peer computing, and cloud computing.

Sun-Yuan Hsieh received the PhD degree in computer science from National Taiwan University, Taipei, Taiwan, in June 1998. He then served the compulsory two-year military service. From August 2000 to January 2002, he was an assistant professor at the Department of Computer Science and Information Engineering, National Chi Nan University. In February 2002, he joined the Department of Computer Science and Information Engineering, National Cheng Kung University, and now he is a full professor. He received the 2007 K. T. Lee Research Award, President’s Citation Award (American Biographical Institute) in 2007, the Engineering Professor Award of Chinese Institute of Engineers (Kaohsiung Branch) in 2008, the National Science Council’s Outstanding Research Award in 2009, and Distinguished Professor Award at National Cheng Kung University in 2010. His current research interests include design and analysis of algorithms, fault-tolerant computing, bioinformatics, parallel and distributed computing, and algorithmic graph theory.