Efficient and Scalable Search on Scale-free P2P Networks

Lu Liu, Jie Xu, Duncan Russell, Paul Townend, David Webster

School of Computing, University of Leeds, Leeds,
West Yorkshire, LS2 9JT, United Kingdom

Corresponding Author: Lu Liu
School of Computing, University of Leeds
Leeds, West Yorkshire, LS2 9JT
United Kingdom
Email: luliu@comp.leeds.ac.uk
Tel: +44 (0) 113 343 5876
Fax: +44 (0) 113 343 5468

Abstract

Unstructured peer-to-peer (P2P) systems (e.g. Gnutella) are characterized by uneven distributions of node connectivity and file sharing. The existence of "hub" nodes that have a large number of connections and "generous" nodes that share many files significantly influences performance of information search over P2P file-sharing networks. In this paper, we present a novel Scalable Peer-to-Peer Search (SP2PS) method with low maintenance overhead for resource discovery in scale-free P2P networks. Different from existing search methods which employ one heuristic to direct searches, SP2PS achieves better performance by considering both of the number of shared files and the connectivity of each neighbouring node. SP2PS enables peer nodes to forward queries to the neighbours that are more likely to have the requested files and also can help in finding the requested files in the future hops. The proposed method has been simulated in different power-law networks with different forwarding degrees and distances. From our analytic and simulation results, SP2PS achieves better performance when compared to other related methods.

Keywords

Peer-to-peer, Search, Scale-free networks, Simulation

1. Introduction

Peer-to-peer (P2P) networking attracts attentions worldwide with its great success in file sharing networks (such as Napster, Gnutella, Freenet, BitTorrent, Kazaa, and JXTA). As a major design pattern for future systems opposite to the traditional client-server paradigm, research on P2P networks is extremely important and could possibly change the way of use of computer systems. In the past few years, great achievements have been made on P2P resource sharing and data transfer. However, we will not reap all the benefits of utilizing these resources in P2P networks unless we have an efficient way to discover them. Efficient resource discovery remains a fundamental challenge for large-scale P2P networks.

Numerous studies have been performed to address the problem of resource discovery in P2P networks. Adamic et al. [1] showed that many peer nodes can be reached by forwarding queries to the neighbouring node with the highest connection degree (the number of connections of a peer node) and thus it can get many results back. Yang and Garcia-Molina [2] tested this model by forwarding queries to the peer node which had the largest number of neighbours. Several other heuristics were examined in [2] to help in selecting the best peer node to send a query, for example, the peer node that returned the highest number of results from previous searches or the peer node that had the shortest latency. The study in [3] evaluated a similar heuristic that the peer nodes sharing more files are more likely to be able to answer the query. A search method was implemented, called Most File Shared on Neighbourhood (MFSN), which enabled peer nodes to forward queries to the neighbours sharing the largest number of files. In accordance with the results shown on the previous studies above, these heuristic-based search methods can generally achieve better performance than random walkers [1, 4].

In this paper, we present a new Scalable Peer-to-Peer Search (SP2PS) method in scale-free (power-law) networks, which enables peer nodes to send queries to the neighbouring nodes that are more likely to have the requested files and can help in finding the requested files in the future hops. In contrast to the above methods which employ one heuristic parameter, SP2PS incorporates two heuristics to guide searches by preferentially selecting the neighbours with a large number of connections and the neighbours sharing a large number of files. SP2PS utilizes high-degree nodes to discover a wider range of accessible peer nodes and locate the requested files in a high likelihood from the neighbours that share a large number of files.

The remainder of this paper is organized as follows: Section 2 discusses background and related work. Section 3 presents the SP2PS algorithm. Simulation results are analysed in Section 4. In Section 5, conclusions are given and future work is outlined.

2. Related work

In this section, existing P2P search systems are analysed by classifying them into two categories: structured and unstructured P2P systems.

2.1 Structured P2P systems

Structured P2P systems have a dedicated network structure on the overlay network which establishes a link between stored content and node addresses. Distributed Hash Tables (DHTs) are widely used for resource discovery in structured P2P systems like Chord [5], ROME [6, 7], Pastry [8], CAN [9], Accordion [10] and Kademlia [11].

In DHT-based P2P systems, each file is associated with a key generated by hashing the file name or content. Each peer node in such systems is responsible for maintaining a certain range of keys. The network structure is sorted by routing tables (or finger tables) stored on individual peer nodes. Each peer node only needs a small amount of "routing" information about other peer nodes (e.g. nodes' addresses and the range of keys the peer node is responsible for). With routing tables and uniform hash functions, peer nodes can conveniently put and get files to and from other peer nodes according to the keys of files.

In structured P2P systems, queries are propagated by a small number of peer nodes to the target node according to distributed routing references stored in each peer node. However, some recent studies (e.g. [2, 12, 13]) argued that the cost of maintaining a consistent distributed index is very high in the dynamic and unpredictable Internet. Some structured P2P systems (e.g. [11, 14, 15]) are attempting to reduce the cost of maintaining a consistent index. Moreover, structured P2P cannot afford to strictly control data placement and the topology of the network, especially when system users come from non-cooperating organisations [13]. Another problem observed in structured P2P systems is that these systems can only support search-by-identifiers and lack the flexibility of keyword searching [16].

2.2 Unstructured P2P systems

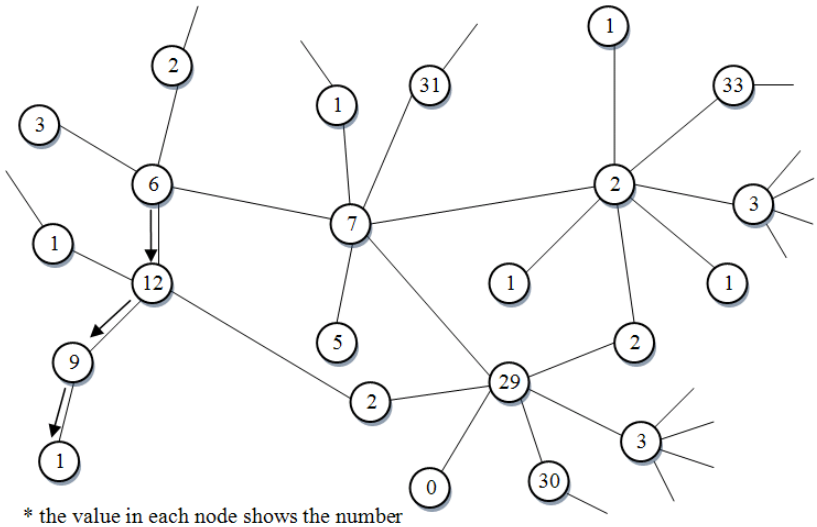
In contrast to structured P2P systems, unstructured P2P systems do not have to maintain network structure. Although existing search methods in unstructured P2P systems are incompatible, most of them are dedicated to solving observed issues with flooding mechanisms which can be classified into two broad categories: blind search and informed search, according to whether the algorithm needs additional indices about the locations of resources. Using a blind search method (e.g. [1, 2, 4]), peer nodes do not need to maintain additional information about resource locations. In contrast, informed search methods enable peer nodes to maintain additional information about other peer nodes in the network, e.g. network topology and resource locations. Existing informed

search solutions can be further classified in two groups: mechanisms with specialised index nodes and mechanisms with indices at each node. The search mechanisms with specialised index nodes have already been adopted in the design of current P2P applications. For example, Napster employs a centralised server to maintain such additional information. Kazaa and JXTA utilize a set of semi-centralised supernodes to maintain the extra information about their leaves and other supernodes.

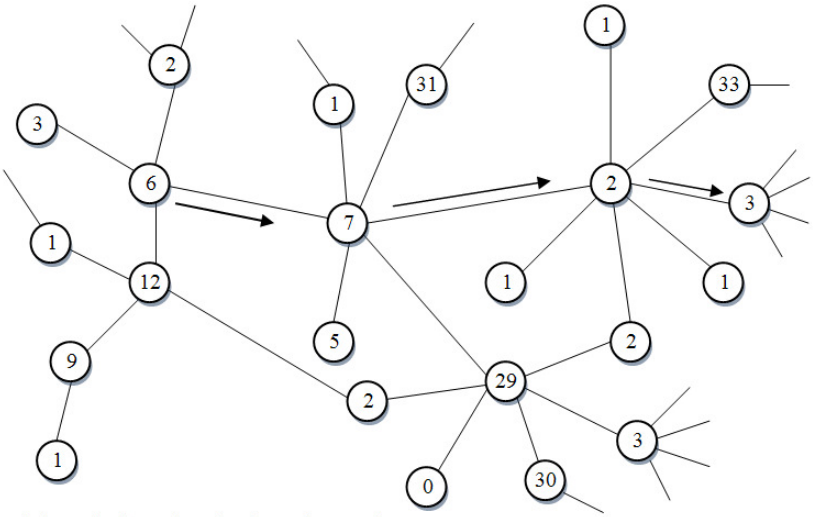
However, centralised indices at a small subset of peer nodes may make these nodes attractive to attackers. New methods have emerged in recent years by distributing indices to each individual peer node in the network. Such methods can be further classified into the following three approaches according to their design principles.

In many P2P networks, topology significantly affects the performance of resource discovery. The first approach (e.g. [17-19]) intends to reduce search cost by adapting and optimising the overlay topology of network. Each peer node is expected to keep topology information about its neighbours or neighbours' neighbours. The second approach (e.g. [20-23]) enables each peer node to keep a routing index which contains detailed semantic information about content of shared files. This information can be collected by exchanging indices regularly with other peer nodes or caching the historic record regarding the results of previous queries. However, maintenance overhead will increase when keeping more concrete information about shared resources. Furthermore, it is difficult for each peer node to maintain and update a large amount of heterogeneous routing indices, especially in the networks with high churn rates [12].

In contrast to the second approach of maintaining detailed semantic information with high overhead, the third approach enables peer nodes to route queries to the neighbours that are likely to have the requested files in accordance with the cached statistical information about neighbours, which is closest related to the methods we are presenting in this paper. The previous studies in [3, 24] discussed that the peer nodes with more files are more likely to be able to answer the query. MFSN was presented in [3]. MFSN enables peer nodes to forward queries to the neighbouring node sharing the largest number of files (see Figure 1(a)) which could have the requested files with a higher probability in the current hop regardless of the probability of finding the requested files in the future hops. The study in [1] indicated that many peer nodes can be reached by forwarding queries to the peer nodes with the highest degree (number of connections) in the neighbourhood (see Figure 1(b)). The corresponding method named DEG was experimented in [2]. DEG utilizes high-degree nodes to discover a wider range of accessible peer nodes which are helpful to find the requested files in the future hops, regardless of the results in the current hop. Different from the above methods which only use one heuristic, SP2PS guides searches regarding both of the parameters: the number of shared files and the connectivity of each neighbour, which is able to overcome the disadvantages of both methods while retaining their advantages.



(a)



(b)

Figure 1 Comparison of search methods (a) MFSN (b) DEG

3. Scalable Search over Scale-free Networks

3.1 Scale-free networks

Barabasi and Albert [25] mapped a part of the World Wide Web (WWW) in 1999 using a Web crawler and displayed the result as a graph. The surprising result is that the WWW did not have an even distribution of connectivity. In contrast, some nodes had many more connections than others. In this graph, the probability $p(x)$ that a node connected to x other nodes was proportional to x to the power of a constant γ: $p(x) \propto x^{-\gamma}$. The similar phenomena have also been observed in some other networks, including P2P networks, social and biological networks. These networks with the same characteristic are called scale-free networks (or power-law networks).

The factors leading to scale-free distributions are complex. For example, a new node joining the network has a preference to early entrants; the longer a node has been in the network, the larger number of connections to it; nodes with a larger number of connections are easier to find; nodes with a higher capability (such as with large bandwidth and sharing a great deal of interesting content) also attract more connections, which causes that the greater the capacity of the node, the faster its growth [26]. Barabasi and Albert [25] presented a "rich get richer" generative model called preferential attachment, where each new Web page creates links to existent Web pages with a probability distribution which is not uniform, but proportional to the current in-degree of Web pages. Thus, a page with many links can attract more links than a regular page, which generates a power-law.

3.2 SP2PS

Previous studies (e.g. [27, 28]) have observed that the connectivity of peer nodes in large-scale P2P networks (e.g. Gnutella, Freenet) follows a power-law distribution. The majority of peer nodes have only a few connections. Instead, a small number of peer nodes are far more connected than others. Like the connectivity of peer nodes, the distribution of file sharing is also unbalanced in P2P networks. Some observed peer nodes in existing P2P networks only tend to download a large amount of files, but share few files or none at all [29]. On the contrary, a small percentage of peer nodes share a large number of files. The distribution of file sharing can also be estimated approximately as a power-law [27, 30].

The existence of such well-connected "hub" nodes and "generous" nodes sharing many files significantly influences the performance of information search in P2P file-sharing networks [31]. In SP2PS, two types of neighbours' information are maintained in each peer node: the number of shared files and the number of connections of each one-hop neighbour (the directly connected neighbouring node in one-hop distance). The node selection is made according to these two types of information.

In a power-law graph, the expected maximum connection degree of the richest neighbour is usually higher than the connection degree of the peer node itself in the networks with a certain range of power-law exponents [1]. The peer nodes with a large number of connections have a high probability to find a neighbouring node with an even larger number of connections to more accessible nodes which may have the requested files. Using SP2PS, a query is forwarded to the well-connected neighbours which can help discover a wider range of accessible peer nodes and locate the requested files in the future hops. SP2PS uses a TTL (Time-to-Live) to prevent infinite propagation. TTL represents the number of hops a message can be forwarded before it is discarded. A query is propagated via the well-connected peer nodes with a TTL of a given value. These high-degree peer nodes can find the peer nodes with an even larger number of connections in the future hops, which could in turn find the peer nodes sharing a larger number files.

The peer nodes with more files are more likely to have the requested file [3]. The requested files could be found in a high probability from these "generous" peer nodes that share a large number of files. Using SP2PS, a query is preferentially sent to the neighbouring nodes that share a large number of files, which can find the requested files with a relatively high probability in the current hop. Because these neighbouring nodes that share a large number of files can only benefit search in the current hop, the query is forwarded to them with a TTL of 1. These peer nodes sharing a large number of files will only process the received queries in hope of finding a matched file for the query, but will not forward the queries further.

In SP2PS, the number of peer nodes to be forwarded in each hop, called forwarding degree, is defined by two parameters: D_{link} and D_{file}. For each search, a query is propagated to top D_{link} highest-degree neighbours with a TTL of a given value. In the meantime, the query is sent to D_{file} neighbours with a TTL of 1 preferring to the neighbours sharing more files. The search originators can also enlarge the forwarding degree (either $D_{link} > 1$ or $D_{file} > 1$, or both) to achieve a higher success rate of searches by forwarding queries to some of neighbours with lower connection degrees or with fewer files. Moreover, the ID of each visited node is attached to the query and SP2PS avoids forwarding the query message to the peer nodes which have already received it.

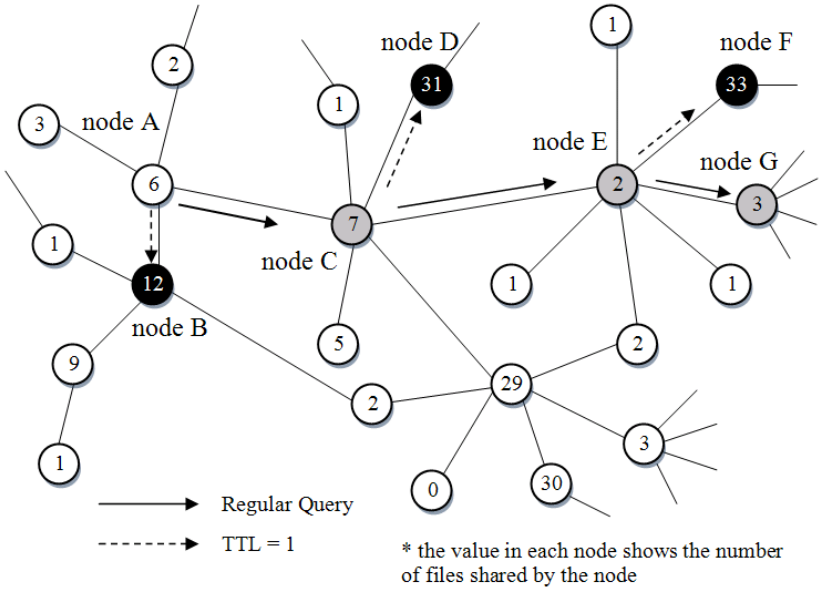


Figure 2 Example of SP2PS

Figure 2 illustrates SP2PS with $D_{link} = D_{file} = 1$, where the grey nodes C, E and G are the peer nodes keeping the largest number of connections in the neighbourhoods of the nodes A, C and E, which received the query. The query is propagated via these grey nodes with a decreased TTL. In the meanwhile, the query with a TTL of 1 is sent to the black nodes B, D, and F, which have the largest number of files in the neighbourhoods of the nodes A, C, and E. These peer nodes with the largest number of files only handle the query by searching shared files locally, but will not forward the query further. In this example, we can see that a total of 94 (6+12+7+31+2+33+3) files are searched by visiting 7 peer nodes. As shown in Figure 1, MFSN searches 28 (6+12+9+1) files and

DEG searches only 18 (6+7+2+3) files from 4 nodes. As shown in this example, SP2PS searches more files than DEG and MFSN.

3.3 Performance evaluation

In this section, the performance of SP2PS is analysed by evaluating the expected quantity of files we could search in each hop over a scale-free network where node connections and file sharing follow power-law distributions with exponents γ_1 and γ_2, respectively. The cut-offs of the both power-law distributions are defined as one, which means that each peer node needs to connect to at least one other node and share at least one file to the network. The power-law distribution of file sharing is given by:

$$p_{files}(x) = \alpha_2 \cdot x^{-\gamma_2},$$

The coefficient of the power-law α_2 is given by:

$$\alpha_2 = \left(\sum_{1}^{m_2} x^{-\gamma_2} \right)^{-1},$$

where m_2 is the maximum number of files in a node. The cumulative distribution function (CDF) is:

$$P_{files}(x) = \int_1^x p_{files}(x)dx.$$

The CDF of the maximum number of files in a node among n neighbouring nodes is:

$$P_{\max\_files}(x,n) = P_{files}(x)^n.$$

Therefore, the distribution of the maximum number of files in a node among n neighbouring nodes is:

$$p_{\max\_files}(x,n) \approx n \cdot (\gamma_2 - 1) \cdot x^{-\gamma_2} \cdot \left(1 - x^{1-\gamma_2}\right)^{n-1} \cdot \left(1 - m_2^{1-\gamma_2}\right)^{-n}. \quad (1)$$

Therefore, the expected maximum number of shared files in a node among n neighbouring nodes is estimated as follows, where $p_{\max\_files}(x,n)$ is given by Equation (1):

$$E\left[x_{\max\_files}(n)\right] = \sum_{1}^{m_2} x \cdot p_{\max\_files}(x,n). \quad (2)$$

The distribution of node connections follows a power-law with exponent γ_1. The distribution of the number of connections from a random node is different from the distribution of node connections from a node following a random connection. The probability of a random connection arriving at a peer node is proportional to the connection degree of the peer node. Therefore, the distribution of outgoing connections [1] from a node reached by a random connection is:

$$p_{conns}(y) = \left(\sum_{0}^{m_1-1} (y+1)^{(1-\gamma_1)} \right)^{-1} (y+1)^{(1-\gamma_1)},$$

where m_1 is the maximum connection degree of peer nodes. The CDF is:

$$P_{conns}(y) = \int_0^x p_{conns}(y)dx$$

The CDF of the maximum number of outgoing connections of node among n neighbouring nodes is:

$$P_{\max\_conns}(y,n) = P_{conns}(y)^n$$

So

$$p_{\max\_conns}(y,n) \approx n \cdot (1+y)^{1-\gamma_1}(\gamma_1 - 2)\left[1-(y+1)^{2-\gamma_1}\right]^{n-1}\left(1-m_1^{\,2-\gamma_1}\right)^{-n} \qquad (3)$$

The expected maximum number of connections of node among n neighbouring nodes is estimated as follows, where $p_{\max\_conns}(y,n)$ is given by Equation (3).

$$E\left[x_{\max\_conns}(n)\right] = \sum_{0}^{m_1-1} y \cdot p_{\max\_conns}(y,n), \qquad (4)$$

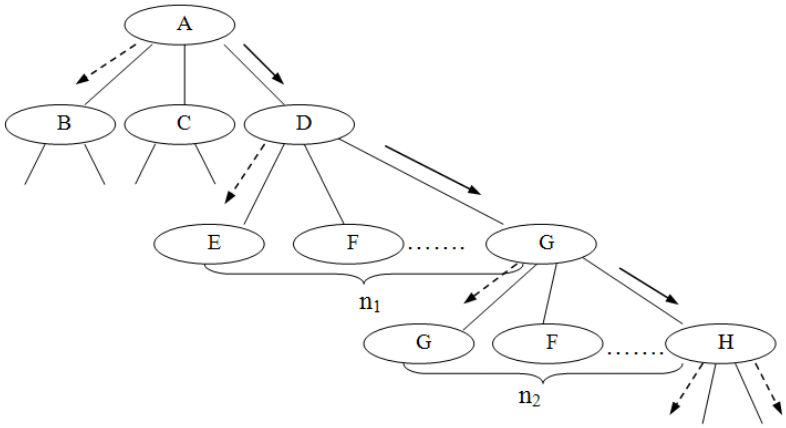


Figure 3 SP2PS query propagation

In case of $D_{link} = D_{file} = 1$, the one-hop neighbour with the largest number of outgoing connections n_1 is utilized to forward queries to the highest-degree neighbour among n_1 two-hop neighbours as illustrated in Figure 3. The expected maximum number of files in a two-hop neighbour is:

$$E[f_2] = E[x_{\max\_files}(n_1)].$$

The expected maximum number of outgoing edges of a two-hop neighbour is:

$$E[n_2] = E[x_{\max\_conns}(n_1)] - 1.$$

The expected maximum number of files in a three-hop neighbour via the two-hop neighbour with the maximum number of outgoing connections n_2 can be estimated approximately as:

$$E[f_3] = E[\mathrm{x}_{\max\_files}(E[n_2])].$$

Because no correlation between the distribution of file sharing and the distribution of node connections are assumed, the number of files in a neighbour with the richest connections is estimated as:

$$E_{random} = \sum_{1}^{m_2} \alpha_2 \cdot x^{1-\gamma_2} \quad . \qquad (5)$$

The expected number of files we can access in the $(i+1)^{th}$ hop is calculated as follows, where $E[x_{\max\_files}(E[n_i])]$, $E[x_{\max\_conns}(E[n_{i-1}])]$ and E_{random} are given by Equation (2), (4) and (5), respectively:

$$E[f_{i+1}] = E[x_{\max\_files}(E[n_i])] + E_{random}, \quad (6)$$

$$E[n_i] = E[x_{\max\_conns}(E[n_{i-1}])] - 1 \ (1 < i < TTL). \quad (7)$$

3.4 RSP2PS

In order to be more efficient, queries need to be forwarded to the "good" nodes that are more likely to have the requested files and can simultaneously help in finding the requested files with a high probability in the future hops. In order to achieve higher efficiency by searching more files via fewer nodes in accordance with neighbours' information maintained in SP2PS, two parameters are considered to determine the "goodness" of a neighbour: the number of accessible files from the neighbour in the current hop and the expected number of accessible files via the neighbour in the future hops. In SP2PS, we utilize high-degree nodes to forward queries and can potentially find the peer nodes with a large number of files in the next hop, regardless of the number of files that are currently shared by these high-degree nodes. In contrast, the strategy MFSN that forwards queries to the peer nodes with a large number of shared files only considers the number of files that can be searched in the current hop, regardless of the number of files that are expected to be searched in the future hops.

In this section, we present a Revised SP2PS (RSP2PS) method in case of D_{link} =D_{file}=1, which forwards queries to the neighbours with the largest sum of the number of shared files on the neighbour and the expected number of accessible files in the future hops via the neighbour. In RSP2PS, a query is propagated by the neighbours with the largest goodness value *G* instead of the neighbours having the largest number of connections. The query is preferentially forwarded to the neighbours that have a large number of connections and simultaneously share a large number of files. In RSP2PS, the goodness value *G* of a neighbouring node is given by the following equation:

$$G = f_1 + (\mathrm{E}[f_2] + \mathrm{E}[f_3] + .. + \mathrm{E}[f_{TTL}]), \quad (8)$$

where f_1 is the number of files shared by the one-hop neighbour and $\mathrm{E}[f_i]$ is the expected number of files that potentially can be accessed in the i^{th} hop via the neighbour which can be estimated using Equations (6) and (7).

The queries are propagated by the neighbours with the largest *G* with a TTL of a given value in each hop. Similarly to SP2PS, the query is also sent to the neighbour with the largest number of files with a TTL of 1. In the last hop, the query is only forwarded to the neighbour with the largest number of files.

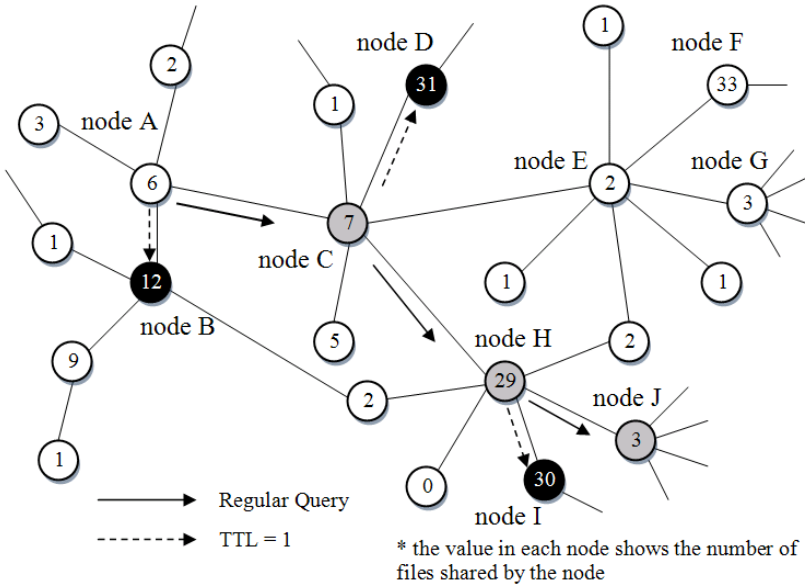


Figure 4 Example of RSP2PS

In the previous example shown in Figure 2, SP2PS forwards the query to node E which has the largest number of connections (7 connections) in the neighbourhood of node C, but shares a few files (2 files only). By using RSP2PS, the query can be preferentially propagated to node H (instead of node E) which has a large number of connections (6 connections) and simultaneously shares a large number of files (29 files) in the neighbourhood of node C, as illustrated in Figure 4. In this example, RSP2PS achieves improved performance by searching a total of 118 (6+12+7+31+29+30+3) files from 7 peer nodes, when compared to SP2PS which searches 91 (6+12+7+13+2+23) files from the same number of peer nodes, as shown in Figure 2.

This goodness value *G* generated by Equation (8) is a better (or equivalent) reference parameter to direct searches rather than connection degree of peer nodes. In order to prove this hypothesis, we assume *t* search methods: S_0, S_1,..., S_{t-1}, with $TTL = t$, where the neighbours with the largest goodness value *G* of each hop are used to forward queries in the first *0, 1,..., t-1* hop(s), respectively. Then a query is propagated among the neighbours with the highest connection degree of neighbourhood in the rest *t-1, t-2, ..., 0* hop(s), respectively. In the last hop, the query is sent to the neighbour with the largest number of files. Hence, S_0 is SP2PS and S_{t-1} is RSP2PS. The expected number of files that can be searched by S_1 can be estimated using Equation (8). Because $(f_1 + E[f_2] + E[f_3] + .. + E[f_{TTL}]])_{\max} \geq E_{random} + (E[f_2] + E[f_3] + .. + E[f_{TTL}]])_{\max}$, the performance of S_1 is better than (or equal to) that of S_0 (SP2PS). In the same way, the performance of S_2 is better (or equal to) than that of S_1 and the rest may be deduced by analogy. We can deduce from the inequalities: $S_0 \leq S_1 \leq S_2 .... \leq S_{t-1}$ that RSP2PS (S_{t-1}) can theoretically achieve better performance than (or equal to) SP2PS (S_0) if the query is propagated until $TTL = 0$.

4. Simulation results

Due to the decentralised nature of P2P networks, it is prohibitively expensive to test a P2P algorithm by deploying it on real-world networks and obtaining data on its performance. Instead, we evaluated the performance of SP2PS and RSP2PS by simulations in different power-law networks with a comparison to other relevant methods. The simulator of SP2PS for this purpose is programmed in the Java language. The main components of the SP2PS simulator are illustrated in Figure 5.

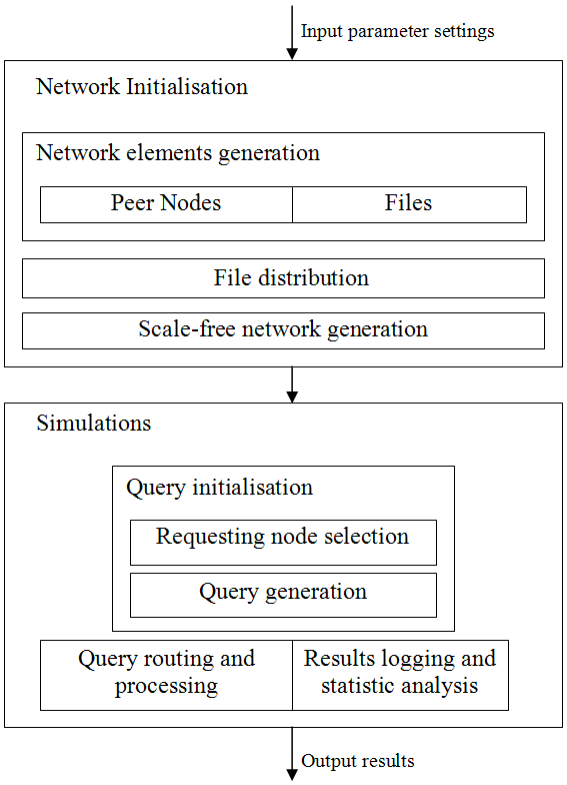


Figure 5 Structure of SP2PS simulator

Performance is evaluated against the following criteria:

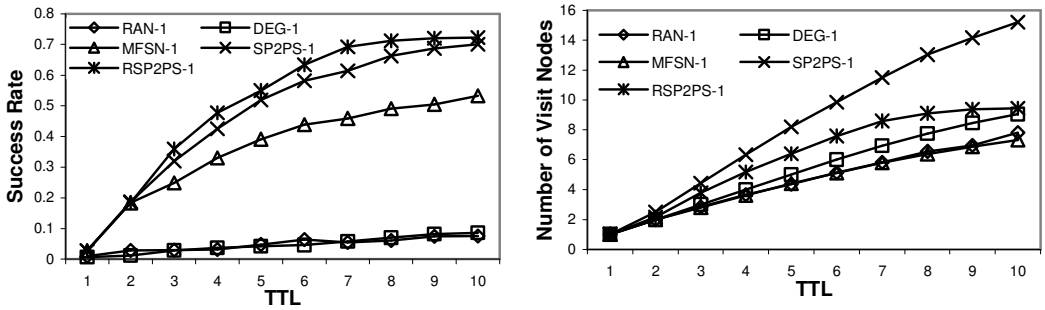
- Success rate: the ratio of successful to the total requests. (A search is successful if it can find at least one requested file.)
- Average number of visited nodes: the average number of peer nodes that have been visited per request.
- Efficiency: the ratio of the average number of files found to the average number of visited nodes.

The simulation network contains 3000 peer nodes. The files randomly selected from a pool of 1000 unique files are distributed to peer nodes following a power-law distribution with the exponent of γ_2=1.5. Each peer node only keeps two types of information about its neighbouring nodes: the number of connections and the number of shared files of each directly connected node.

In each time step of simulations, a peer node is randomly selected as the query originator to start a lookup for a file. The average results are generated from the experimental results of 1000 searches (1000 time steps). SP2PS and RSP2PS with D_{link} = D_{file} =1 (SP2PS-1 and RSP2PS-1) have been examined with a comparison to the following methods:

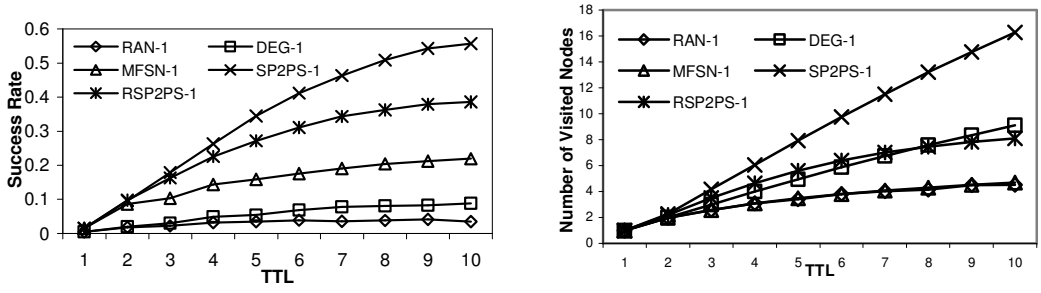
- MFSN-1: forward queries to the neighbour with the largest number of files in each hop;
- DEG-1: forward queries to the highest connection degree neighbour in each hop;
- RAN-1: randomly forward queries to a neighbour in each hop.

In the Gnutella network, the power-law exponent of connectivity distribution γ_1 is estimated as low as 1.4 and as high as 2.3 [32]. Figure 6 and Figure 7 show the respective performance results in the networks with power-law exponents γ_1 = 1.5 and γ_1 = 2.0. Search efficiencies in the networks with γ_1 = 1.5 and γ_1 = 2.0 are shown in Figure 8(a) and 8(b), respectively.



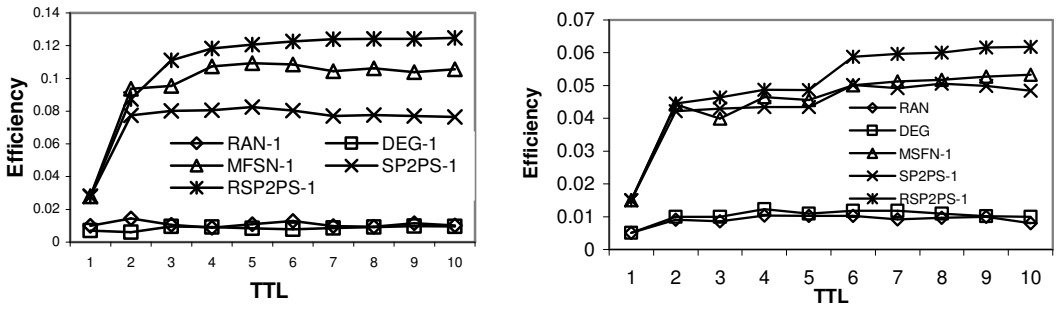
(a) (b)

Figure 6 Performance parameters in a power-law network with exponent γ_1 =1.5 (a) Success rate (b) Average number of visited nodes



(a) (b)

Figure 7 Performance parameters in a power-law network with exponent γ_1 =2.0 (a) Success rate (b) Average number of visited nodes



(a) (b)

Figure 8 (a) Efficiency in a power-law network with exponent γ_1 = 1.5 (b) Efficiency in a power-law network with exponent γ_1 = 2.0

As shown in Figure 6(a) and Figure 8(a), RSP2PS has achieved the highest success rate and efficiency by visiting a small number of peer nodes in the network with γ_1 = 1.5. However, the success rate of RSP2PS is lower than that of SP2PS in the network with γ_1 = 2.0 as shown in Figure 7(a), but the efficiency of RSP2PS is still the highest among all simulated methods as shown in Figure 8(b).

Because queries are not allowed to send back to previously visited peer nodes, the queries cannot be propagated further by those "edge nodes" with only one connection. However, a large percentage of peer nodes only keep one connection in the network with a large exponent: γ_1=2.0. Many queries are discarded before expiring ($TTL = 0$). By propagating queries via the neighbours with highest connection degree, queries are passed to the peer nodes with a large number of connections by avoiding sending to these "edge nodes". Therefore, in the network with exponent γ_1=2.0, the number of visited nodes in RSP2PS is fewer than that of SP2PS, which makes its success rate lower.

MFSN only considers the number of files of each neighbour regardless of the number of connections of each neighbour, therefore queries cannot be forwarded to a wider range of peer nodes and the number of visited nodes is very limited, similarly to RAN, as shown in Figure 6(b) and Figure 7(b). The efficiency of MFSN is higher by visiting peer nodes with a large number of files, but the success rate is relatively low with a small number of visited nodes. As shown in Figure 6(a), DEG achieves a similar success rate as RAN in the network of γ_1 = 1.5, but obtains an obviously better success rate in the network of γ_1 = 2.0 as shown in Figure 7(a), since DEG directs searches avoiding propagating queries to those "edge nodes" with one connection which can not propagate queries further.

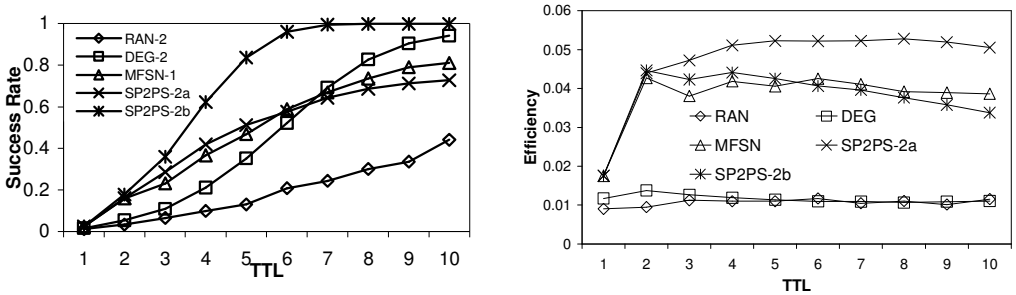
The performance of SP2PS is further evaluated with a higher forwarding degree:

- SP2PS-2a: $D_{link} = 1$, $D_{file} = 2$. A query is propagated among the highest-degree neighbours and simultaneously sent to the two neighbours that share the first and second largest numbers of files with a TTL of 1 in each hop;

- DSearch-2: $D_{link} = 2, D_{file} = 1$. A query is propagated to the neighbours with the first and second highest connection degrees and simultaneously sent to the neighbour that shares the largest number of files with a TTL of 1 in each hop.

And compare them to the following methods:

- MFSN-2: forward queries to the two neighbours with the first and second largest numbers of files in each hop;
- DEG-2: forward queries to the neighbours with the first and second highest connection degrees in each hop;
- RAN-2: forward queries to two randomly selected neighbours in each hop.



(a) (b)

Figure 9 Performance parameters with the forwarding degree of 2 (a) Success rate (b) Efficiency

As shown in Figure 9, SP2PS-2b performs the highest success rate, while SP2PS-2a achieves the highest search efficiency. Therefore, SP2PS-2b is recommended for the network with abundant bandwidth since it can achieve more than a 96% success rate within 6 hops. On the contrary, SP2PS-2b is recommended for the bandwidth-limited networks since it achieves the highest efficiency. In Figure 9(b), the efficiency of MFSN and SP2PS-2a drops when TTL increases. The expected number of links of the richest neighbour is higher than the number of files of a random node itself [1]. However, as a query moves to the nodes with higher and higher connection degrees, the probability of discovering a neighbour with an even higher connection degree [1] and finding a neighbour with an even larger number of files will begin decreasing. Therefore, the efficiency starts falling with increasing TTLs, as shown in Figure 9(b).

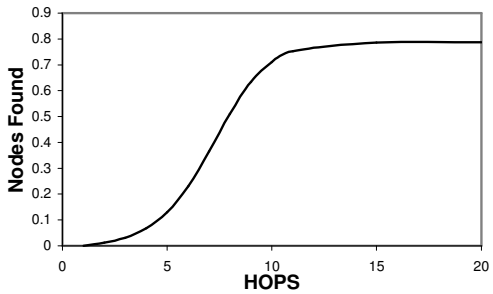


Figure 10 Percentage of peer nodes discovered

Using SP2PS, the high-degree nodes with a large number of connections are utilized to discover a wider range of accessible peer nodes. SP2PS is simulated to see the portion of peer nodes can be discovered and whether the search will get stuck in a small number of peer nodes. We set $D_{link} = 2$ and $D_{file}=\infty$ in the simulation to get a total of peer nodes we can potentially access. The queries are propagated via the neighbours with the first and second highest degrees with a TTL of a given value from one to twenty and all connected neighbours are accessed in each hop with a TTL of 1. As shown in Figure 10, a large portion of the network can be discovered within ten hops.

5. Conclusions and future work

Unstructured P2P systems (e.g. Gnutella) are characterized by uneven distributions of both node connectivity and file sharing. In this paper, we presented new scalable search methods with low maintenance overhead in scale-free (power-law) networks where both node connections and file sharing follow power-law distributions. Different from existing search methods, like DEG [2] and MFSN [3], which employ one heuristic to direct searches, SP2PS and RP2PS achieve better performance by considering both of the number of shared files and the connectivity of each neighbour. The SP2PS and RSP2PS search methods enable peer nodes to forward queries to the neighbouring nodes that are more likely to have the requested files in the current hop and can help in finding the requested files with a high probability in the future hops. These methods have been simulated in different power-law networks with different forwarding degrees and forwarding distances. From our simulation results, SP2PS and RSP2PS can achieve better performance when compared to other related methods.

SP2PS and RP2PS are able to achieve good search performance under the assumption that we only have the minimal information about each directly connected node: the number of shared files and the number of connections. With the ever-increasing power of computer hardware and communication bandwidth, it is possible for each peer node to cache more detailed semantic information about resources that are shared by the neighbouring nodes in the future. Maintaining a relatively higher overhead in return could achieve better search performance. The next step will be to extend SP2PS to work with other P2P search algorithms (e.g. [14, 33]) to increase the fault-tolerance of these previous algorithms.

References

1. Adamic, L.A., Lukose, R.M., Puniyani, A.R., Huberman, B.A. (2001) Search in Power Law Networks. Physical Review 64 046135-046131 - 046135-046138

2. Yang, B., Garcia-Molina, H. (2002) Efficient Search in Peer-to-Peer Networks. International Conference on Distributed Computing Systems, Vienna, Austria

3. Li, X., Wu, J. (2005) A Hybrid Searching Scheme in Unstructured P2P Networks. International Conference on Parallel Processing, Oslo, Norway

4. Lv, Q., Cao, P., Cohen, E., Li, K., Shenker, S. (2002) Search and Replication in Unstructured Peer-to-Peer Networks. ACM SIGMETRICS, Marina Del Rey, CA

5. Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H. (2001) Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications. ACM SIGCOMM, San Diego, CA

6. Antonopoulos, N., Exarchakos, G. (2007) G-ROME: A Semantic Driven Model for Capacity Sharing Among P2P Networks. Journal of Internet Research 17 7-20

7. Salter, J., Antonopoulos, N. (2007) An Optimised 2-Tier P2P Architecture for Contextualised Keyword Searches. Journal of Future Generation Computer Systems 23 241-251

8. Rowstron, A., Druschel, P. (2001) Pastry: Scalable, Distributed Object Location and Routing for Large-scale Peer-to-Peer Systems. IFIP/ACM International Conference on Distributed Systems Platforms, Heidelberg, Germany

9. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Shenker, S. (2001) A Scalable Content-Addressable Network. ACM SIGCOMM, San Diego, CA

10. Li, J., Stribling, J., Morris, R., Kaashoek, M.F. (2005.) Bandwidth-Efficient Management of DHT Routing Tables. 2nd Symposium on Networked Systems Design and Implementation, Boston, MA

11. Maymounkov, P., Mazi`eres, D. (2002) Kademlia: A Peer to Peer Information System Based on the XOR Metric. Internation Workshop on Peer-to-Peer Systems, Cambridge, MA

12. Rhea, S., Geels, D., Roscoe, T., Kubiatowicz, J. (2004) Handling Churn in a DHT. USENIX Annual Technical Conference, Boston, MA

13. Vuong, S., Li, J. (2003) Efa: an Efficient Content Routing Algorithm in Large Peer-to-Peer Overlay Networks. International Conference on Peer-to-Peer Computing, Linköping, Sweden

14. Liu, L., Antonopoulos, N., Mackin, S. (2007) Fault-tolerant Peer-to-Peer Search on Small-World Networks. Journal of Future Generation Computer Systems 23 921-931

15. Liu, L., Antonopoulos, N., Mackin, S. (2007) Small World Peer-to-peer for Resource Discovery. International Conference on Information Networking, Lecture Notes in Computer Science, Estoril, Portugal

16. Li, J., Vuong, S. (2004) An Efficient Clustered Architecture for P2P Networks. 18th International Conference on Advanced Information Networking and Application, Fukuoka, Japan

17. Chawathe, Y., Ratnasamy, S., Breslau, L., Lanham, N., Shenker, S. (2003) Making Gnutella-Like P2P System Scalable. ACM SIGCOMM, Karlsruhe, Germany

18. Xiao, L., Liu, Y., Ni, L.M. (2005) Improving Unstructured Peer-to-Peer Systems by Adaptive Connection Establishment. IEEE Transactions on Computers 54 176-184

19. Liu, Y., Xiao, L., Liu, X., Ni, L.M., Zhang, X. (2005) Location Awareness in Unstructured Peer-to-Peer Systems. IEEE Transactions on Parallel and Distributed Systems 16 163-174

20. Crespo, A., Garcia-Molina, H. (2002) Routing Indices for Peer-to-Peer Systems. International Conference on Distributed Computing Systems, Vienna, Austria

21. Sripanidkulchai, K., Maggs, B., Zhang, H. (2003) Efficient Content Location Using Interest-Based Locality in Peer-to-Peer Systems. IEEE Infocom, San Francisco

22. Liu, L., Antonopoulos, N., Mackin, S., Xu, J., Russell, D. (2008) Efficient Resource Discovery in Self-organized Unstructured Peer-to-Peer Networks, in press. Concurrency and Computation: Practice and Experience

23. Liu, L., Antonopoulos, N., Mackin, S. (2007) Social Peer-to-Peer for Resource Discovery. 15th Euromicro International Conference on Parallel, Distributed and Network-based Processing, Naples, Italy

24. Marti, S., Garcia-Molina, H. (2004) Limited Reputation Sharing in P2P Systems. ACM Conference on Electronic Commerce, New York, USA

25. Barabasi, A.L., Albert, R. (1999) Emergence of Scaling in Random Networks. Science 286 509-512

26. Robb, J. (2004) Scale-free Networks. Global Guerrillas.
http://globalguerrillas.typepad.com/globalguerrillas/2004/05/scalefree_terro.html

27. Saroiu, S., Gummadi, P.K., Gribble, S.D. (2002) A Measurement Study of Peer-to-Peer File Sharing Systems. International Conference on Multimedia Networking and Computing, Santa Barbara, CA

28. Gummadi, K.P., Dunn, R.J., Saroiu, S., Gribble, S.D., Levy, H.M., Zahorjan, J. (2003) Measurement, Modelling and Analysis of a P2P File-sharing Workload. ACM Symposium on Operating Systems Principles, Bolton Landing, New York

29. Pauli, C., Shepperd, M. (2005) An Empirical Investigation into P2P File-Sharing User Behaviour. Americas Conference on Information Systems, Omaha, Nebraska

30. Perera, G., Christensen, K., Roginsky, A. (2005) Targeted Search: Reducing the Time and Cost for Searching for Objects in Multiple-Server Networks. International Performance Computing and Communications Conference, Phoenix, USA

31. Liu, L., Antonopoulos, N., Mackin, S. (2006) Directed Information Search and Retrieval over Unstructured Peer-to-Peer Networks. the International Computer Engineering Conference, Cairo, Egypt

32. Banaei-Kashani, F., Shahabi, C. (2003) Criticality-based Analysis and Design of Unstructured Peer-to-Peer Networks as Complex Systems. IEEE/ACM International Symposium on Cluster Computing and Grid, Tokyo, Japan

33. Liu, L., Antonopoulos, N., Mackin, S. (2008) Managing Peer-to-Peer Networks with Human Tactics in Social Interactions, in press. Journal of Supercomputing