Research Article

CSA: A Credibility Search Algorithm Based on Different Query in Unstructured Peer-to-Peer Networks

Hongyan Mei,^{1,2} Yujie Zhang,^{1,2} and Xiangwu Meng^{1,2}

¹ Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia,
Beijing University of Posts and Telecommunications, Beijing 100876, China

² School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China

Correspondence should be addressed to Hongyan Mei; liaoning_mhy@126.com

Received 27 February 2014; Revised 12 June 2014; Accepted 13 June 2014; Published 10 July 2014

Academic Editor: He Huang

Copyright © 2014 Hongyan Mei et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Efficient searching for resources has become a challenging task with less network bandwidth consumption in unstructured peer-to-peer (P2P) networks. Heuristic search mechanism is an effective method which depends on the previous searches to guide future ones. In the proposed methods, searching for high-repetition resources is more effective. However, the performances of the searches for nonrepetition or low-repetition or rare resources need to be improved. As for this problem, considering the similarity between social networks and unstructured P2P networks, we present a credibility search algorithm based on different queries according to the trust production principle in sociology and psychology. In this method, queries are divided into familiar queries and unfamiliar queries. For different queries, we adopt different ways to get the credibility of node to its each neighbor. And then queries should be forwarded by the neighbor nodes with higher credibility. Experimental results show that our method can improve query hit rate and reduce search delay with low bandwidth consumption in three different network topologies under static and dynamic network environments.

1. Introduction

In the past ten years, peer-to-peer (P2P) networks have gained full development and become an important part of the Internet. P2P networks are divided into structured P2P networks and unstructured P2P networks. Unstructured P2P networks are characterized with self-organization, distributed resource sharing, semantic queries, and so forth, which have been widely applied on the Internet, such as Gnutella [1], FastTrack [2] and KaZaA [3]. However, because of the dynamic characteristics of unstructured P2P networks, it is difficult to capture correctly global behavior [4, 5]. Each node in unstructured peer-to-peer networks does not have global information about the whole network topology and the location of queried resources. Thus, designing an efficient search algorithm has been a hot research issue in unstructured P2P networks.

There are mainly two kinds of search methods in unstructured P2P networks: blind search methods and informed

search methods. In the former, such as flooding [1], peers possess no knowledge to guide the search process, resulting in great blindness. When the size of network increases, the search time will be extended, a large number of redundant messages will be created, and large amounts of network bandwidth will be consumed. In order to reduce the bandwidth consumption, many improved methods [6–15] have been proposed on the basis of the blind search algorithms. And literatures [8–10] have improved flooding algorithm in network bandwidth consumption while preserving large coverage, response time, and flexibility of flooding in dynamic environment. The effective and optimizing search algorithms have been presented in literatures [11, 12], which achieve higher performance than random walks in terms of number of hits, network overhead, and response time by adopting stochastic process knowledge and estimating of the popularity of a resource, respectively. A hybrid search scheme [13] and light flood [14] are proposed by combining flooding and random walks and make full use of the both merits so as to minimize

redundant messages. In literature [15], RFSA limits effectively that the message be received and forwarded repeatedly in blind search methods by using the real-time search path information and the local messages index caching mechanism, thus reducing the production of a great number of redundant messages.

In contrast to the blind search methods, many informed search algorithms have been more extensively studied and proposed, such as intelligent search [16], APS [17], PQR [18] and SPUN [19]. In literature [16], an intelligent search is proposed where a query is forwarded to neighbors that have answered the most queries similar to the current query. APS [17] is a popular adaptive probability k random walks search algorithm and is also bandwidth efficient and easy to implement unstructured P2P search algorithm. APS utilizes the feedback information from previous searches to guide the future ones probabilistically. In APS, each node maintains an index table to record success rate of each neighbor for each requested resource in previous searches. APS probabilistically selects those neighbor nodes which get higher success rate for the requested resource from previous searches. Thus, the search will be successfully guided to the requested resource. At the same time, the success rate is updated dynamically based on whether a peer returns a hit or miss for a given query. PQR [18] is a novel query routing mechanism for improving query performance in unstructured P2P networks. In PQR, a data structure called traceable gain matrix (TGM) is designed and used to record gain value of every query at each peer along the query hit path. By TGM, PQR can optimize query routing decision effectively and achieve high query hit rate with low bandwidth consumption. TGM is an important component of PQR with a compound data structure and maintains query routing information. In these methods [16–18], peers update index values only based on the return type of the query message, success, or failure. And in APS and PQR, peers also have a tendency to use the first discovered neighbor node which reduces search performance in dynamic environments. As for these problems, SPUN is proposed in [19]. SPUN is an informed search algorithm that improves upon state-of-the-art APS. Each peer in SPUN maintains a vector of relative success rates (RSRV) along a query path for a given neighbor for a requested resource resulting in a more informed decision in SPUN. SPUN uses best path gradient (BPG) as neighbor node selection mechanism, which firstly calculates PG values of different paths through neighbor nodes and then discovers more successful query paths through the neighbors according to PG values. The purpose of SPUN is to select neighbor node with the most successful query path to forward query message. However, the most successful query path is often “traffic arteries” for the requested resources, which can cause search bottleneck problem. On the other hand, it only considers the success of a path with no consideration to the distance information of the path. The most successful path may be the longest and the most congested, thereby increasing the search time and network overhead.

The common characteristic of these methods above is to guide future searches through search information recorded previously. Therefore, in the search process, these algorithms

are effective for the repeated queries for the same objects (resources) or similar objects (resources). Because these algorithms can gain heuristic information from the relevant indexes such as TGM (in PQR) or RSRV (in SPUN). But for those nonrepetitive queries such as queries for rare resources, there is no heuristic information in the relevant indexes, so the queries will be forwarded to a random neighbor node (peer). At this moment, the search is inefficient in these methods and their overall search performance will be reduced.

The main motivation of our research is to solve the inefficient problem for nonrepetition or low-repetition or rare resources queries. So, a credible search algorithm (CSA) based on different query is proposed in this paper. The main purpose is to improve the search performance of the searching for nonrepetition or low-repetition or rare resources and repeated queries through the effective guidance and then achieve the higher overall search performance. The contributions of this paper are shown as follows. (1) It is the first in which queries are divided into familiar queries and unfamiliar queries and for different query adopting different calculation method to obtain credibility information and then selecting neighbor nodes with higher credibility as passers. (2) We give a credibility calculation method to calculate neighbor's credibility according to familiarity and similarity among queries and nodes based on the trust production principle in sociology and psychology. (3) Design a new data structure: query credible matrix (QCM), recording the credibility of each neighbor node for each specific object. (4) The proposed methods can achieve high query hit rate with low search delay and low bandwidth consumption and improve the search performance for nonrepetitive queries, such as rare resource queries.

2. Credibility Calculation Method

The studies of literature [20] show that trust among humans consists of two parts: the one generated by the familiarity and the other one by similarity. In social networks, the more familiar among humans, the more trust produced will be. The more similar people are in their interests and hobbies, the easier they will trust each other. So, through familiarity and similarity to calculate trust among humans can reflect the generating process of trust in social networks. Considering the similarity between unstructured P2P networks and social networks, the principle of trust generated among humans in social network can be used to the credibility calculation of node to its neighbors in unstructured P2P networks. The credibility of node to its neighbors is also divided into the trust generated by the familiarity and the trust generated by the similarity.

Suppose p_t stands for an arbitrary peer in unstructured P2P networks and p_j is an arbitrary neighbor peer of the peer p_t . The credibility of p_t to p_j is calculated as

$$\text{Cre}(p_t, p_j) = \text{fam}(p_t, p_j) + \text{sim}(p_t, p_j), \quad (1)$$

where $\text{Cre}(p_t, p_j)$ represents the credibility of the peer p_t to its neighbor peer p_j . $\text{fam}(p_t, p_j)$ refers to the trust generated

by the familiarity between p_t and p_j . In familiarity study, people often gain the familiarity through the number of contacts among humans. In our study, $\text{fam}(p_t, p_j)$ is defined as the success rate of the communications between p_t and p_j . The more the number of success communications is, the higher the success communication rate is and then the more familiar is between p_t and p_j . And the credibility of p_t to p_j is higher. $\text{sim}(p_t, p_j)$ denotes the trust generated by the similarity between p_t and p_j . In unstructured P2P networks, contents stored on a peer reflect the interests and hobbies of this peer and then $\text{sim}(p_t, p_j)$ will be obtained according to the similarity of contents stored on p_t and p_j .

2.1. Credibility Calculation Based on Familiarity. In social networks, familiarity among humans mainly derives from the mutual help and constant contacts, and so forth. Similar to social networks, in unstructured P2P networks, whether peer and its neighbor are familiar will be decided through the behavior of communications between the peer and its neighbor.

In general, the more the numbers of communications are between p_t and p_j , the more familiar they are. At the same time, this communication is bidirectional. If only p_t sends query messages to p_j and p_j cannot return successful messages to p_t , then, although the number of communications between them becomes more, the neighbor peer p_j is still not credible for the future search. So the number of communications between p_t and p_j cannot reflect better the familiarity of p_t to p_j . We can not only use the number of communications as the credibility of p_t to p_j generated by the familiarity.

Because query tends to be passed to the first peers found [17] or the peers with higher degree [21] in the informed methods, then these peers have more opportunities to be selected as passers. And thus, these peers can get higher number of success return messages than those peers that have less opportunity to be selected as passers, even though most query messages forwarded through the peers can be successfully returned. Meanwhile the first peers or peers with higher degrees will be selected repeatedly to forward messages. And thus search bottleneck problem on these peers will be produced and the broadness of search will be reduced.

For example, suppose the number of messages of p_t sending to p_1 is 7 and the number of messages of success return is also 7. The number of messages of p_t sending to p_2 is 20, and the number of messages of success return is 10. If we adopt the number of success communications as the credibility of p_t to p_j generated by the familiarity, then $\text{fam}(p_t, p_1) = 7$, $\text{fam}(p_t, p_2) = 10$. Although all messages of p_t sending to p_1 have been all returned successfully, because the number of query messages forwarded by p_1 is less, the number of messages of success return of p_1 is still less than that of p_2 . Thus p_2 gains more confidence than p_1 and will be more selected, which will cause bottleneck problem of the peer p_2 and ignores the selection for p_1 . So, we calculate the trust of p_t to p_j generated by the familiarity according to communication success rate in this paper. And

thus, $\text{fam}(p_t, p_1) = 1$ and $\text{fam}(p_t, p_2) = 0.5$; the peer p_1 can gain more trust and it will be more selected than p_2 , which reduces bottleneck problem of p_2 and improves performance of search in future search.

Therefore, the credibility of p_t to p_j generated by familiarity is calculated as

$$\text{fam}(p_t, p_j) = \frac{|\text{SR}_{p_t \leftarrow p_j}|}{|F_{p_t \rightarrow p_j}|}. \quad (2)$$

In formula (2), $F_{p_t \rightarrow p_j}$ is a set of messages from p_t forwarded to p_j . $\text{SR}_{p_t \leftarrow p_j}$ is also a set of messages from p_j success return to p_t . Where $\text{fam}(p_t, p_j) \in [0, 1]$, when $\text{fam}(p_t, p_j) = 0$, no success message is returned to p_t through p_j . When $\text{fam}(p_t, p_j) = 1$, the query messages forwarded to p_j are all returned successfully to p_t . Thereby the greater $\text{fam}(p_t, p_j)$ is, the more credible p_j is and selecting p_j to forward the query will get the higher probability of success hit.

2.2. Credibility Calculation Based on Similarity. In social networks, trust can be built on among humans who have similar family background, race, values, interests, hobbies, and so forth. Similar to social networks, in unstructured P2P networks, the different contents stored on peers contain preferences information of these peers. At the same time, studies [22, 23] show that a better search performance can be get though clustering similar peers into the same group based on the similarity of preferences of the peers, such as interests or hobbies, and then searching in these groups. This suggests peer can get more success hit from neighbor peers similar with the peer. So the more similar peer and its neighbor are, the more they will trust each other. In this section, we obtain the preferences information of peers from contents stored on the peers and then according to the similarity of the preferences information of peers to calculate the credibility between peer and its neighbor generated by similarity.

Given an object set of peer p_t : $O_t = \{o_1, o_2, \dots, o_j, \dots, o_m\}$ with m elements, because of advances in metadata retrieval technology [24–26], it is easier to obtain keywords information of every object. Let $K_{o_j} = \{k_{j1}, k_{j2}, \dots, k_{jn}, \dots\}$ be a set of keywords used to describe an object o_j . We denote the keywords set of all objects peer p_t held by $K_{p_t} = \bigcup_{j=1}^m K_{o_j}$ and the characteristics (preferences information) of peer p_t are defined as a vector of weights $\vec{p}_t = (\omega_{t,k1}, \omega_{t,k2}, \dots, \omega_{t,kn}, \dots)$, where the weight $\omega_{t,kn}$ denotes the preference of peer p_t for objects described by the keyword k_n as follows:

$$\omega_{t,kn} = \frac{|O_{t,kn}|}{|O_t|}, \quad (3)$$

where O_t is the set of objects held by peer p_t and $O_{t,kn}$ is a subset of O_t containing objects tagged by keyword k_n .

The similarity $\text{sim}(p_t, p_j)$ between p_t and p_j can be measured by comparing their preferences. There are several methods such as the correlation coefficient, the cosine similarity measure [27], and the Euclidean distance that can be used to compute the distance between two description vectors and return a quantitative value to represent the similarity between peers. In this context, we use the cosine similarity measure to quantify the similarity $\text{sim}(p_t, p_j)$ as follows:

$$\begin{aligned} \text{sim}(p_t, p_j) &= \cos(\vec{p}_t, \vec{p}_j) \\ &= \frac{\vec{p}_t \cdot \vec{p}_j}{\|\vec{p}_t\|_2 \times \|\vec{p}_j\|_2} \\ &= \frac{\sum_{k=1}^m \omega_{t,k} \omega_{j,k}}{\sqrt{\sum_{k=1}^m \omega_{t,k}^2} \sqrt{\sum_{k=1}^m \omega_{j,k}^2}}, \end{aligned} \quad (4)$$

where m is the total number of keywords. If p_t and p_j have similar interests in the contents they hold, then a bigger value $\text{sim}(p_t, p_j)$ will be obtained.

2.3. Credibility Calculation Based on Query. In a real world, when a person is asked to do a task, who tends to choose his (her) credible and capable friends to complete this matter? For example, in six degrees of segmentation principle, everyone who receives the letter usually passes the letter to his friends who have similar information with the letter.

In unstructured P2P networks, for a specific query q_h , we use $\text{sim}(q_h, p_t, p_j)$ to denote the credibility between p_t and its neighbor p_j based on the specific query q_h . The value of $\text{sim}(q_h, p_t, p_j)$ is calculated as

$$\text{sim}(q_h, p_t, p_j) = \frac{|K_{q_h} \cap K_{p_j}|}{|K_{q_h}|}, \quad (5)$$

where K_{q_h} is a set of keywords included in the query q_h , K_{p_j} denotes the keywords set of all objects peer p_j hold, and $\text{sim}(q_h, p_t, p_j) \in [0, 1]$. There are no similar keywords with the specific query q_h in peer p_j , if $\text{sim}(q_h, p_t, p_j) = 0$. All keywords of the specific q_h are contained in the keywords set of objects peer p_j hold, if $\text{sim}(q_h, p_t, p_j) = 1$. So, the more the number of similar keywords is between q_h and p_j , the bigger the value of $\text{sim}(q_h, p_t, p_j)$ is; then the neighbor peer p_j is more credible for the specific query q_h .

According to calculation method of the credibility $\text{fam}(p_t, p_j)$ generated by familiarity in Section 2.1, we denote the credibility $\text{fam}(q_h, p_t, p_j)$ generated by familiarity based on the specific query q_h as follows:

$$\text{fam}(q_h, p_t, p_j) = \frac{\left| \text{SR}_{p_t \xleftarrow{K_{q_h}} p_j} \right|}{\left| F_{p_t \xrightarrow{K_{q_h}} p_j} \right|}, \quad (6)$$

where $F_{p_t \xrightarrow{K_{q_h}} p_j}$ is a set of all messages that contain keywords K_{q_h} from p_t forwarded to p_j . $\text{SR}_{p_t \xleftarrow{K_{q_h}} p_j}$ is a set of all messages that contain K_{q_h} from p_j success return to p_t and $\text{fam}(q_h, p_t, p_j) \in [0, 1]$. If $\text{fam}(q_h, p_t, p_j) = 0$, no message containing keywords K_{q_h} is returned successfully to p_t from p_j ; then the credibility of p_t to p_j generated by familiarity for the specific query q_h is 0. If $\text{fam}(q_h, p_t, p_j) = 1$, all messages containing keywords K_{q_h} are returned successfully to p_t from p_j ; then the credibility of p_t to p_j generated by familiarity for the specific query q_h is 1. So, the bigger the value of $\text{fam}(q_h, p_t, p_j)$ is, the more credible the neighbor peer p_j is for the query q_h .

Above, we introduce the calculation method of credibility between peer and its neighbor and also describe the calculation method of credibility based on specific query between peers. In this method, we choose different credibility calculation methods to obtain trust for its neighbor peers according to different query types and then select neighbor peers with higher credibility to pass messages. Our algorithm will be described in detail in Section 4.

3. Data Structure and Update Mechanism in CSA

3.1. Message Structure in CSA

Query Message. We describe a query message *meg* initiated at peer p_s using a tuple of seven elements $\text{meg}(id, p_s, p_f, \text{state}, \text{hops}, K_{q_h}, \text{path})$. Where *id* denotes the unique identifier of a message in the unstructured P2P networks. p_s is the source peer that initiates the query message *meg*. p_f stands for the sender of the query message. *state* is type of the *meg* and contains three types: *Forward*, *Success_Return*, and *Failure_Return*. We define *hops* as the current hop value of the query message. K_{q_h} is a set of keywords included in the queried object. *path* describes search path of the query message and consists of peers that have traveled by the query, including the source peer p_s .

3.2. Index Structure in CSA. In CSA, credible neighbor list (short for CNL) and credible query matrix (short for CQM) are two main index structures that maintain peer local information. In CNL, each item describes the credibility of peer to each neighbor generated by similarity. The query information of each neighbor peer for each requested object is recorded in each item of CQM. We need two auxiliary data structures: query object list (short for QOL) and query peer list (short for QPL) before CQM is produced. The QOL of a peer is a set of objects that records all objects requested and forwarded from this peer. The QPL of a peer is a subset of its neighbors that records neighbors that one or more requested objects forwarded through the neighbors and returned successfully. In CQM, each row stands for each object in QOL and each column stands for each peer in QPL. Given O is a set of all objects in QOL with m elements, an arbitrary $o_i \in O$, $i \in [1, m]$. P is a set of all peers in QPL with

n elements, an arbitrary $p_j \in P$, $j \in [1, n]$. CQM is an $m \times n$ matrix like this:

$$\begin{matrix} & p_1 & p_2 & \cdots & p_j & \cdots & p_n \\ \begin{matrix} o_1 \\ \vdots \\ o_i \\ \vdots \\ o_m \end{matrix} & \begin{pmatrix} \text{Cre}_{11} & \text{Cre}_{12} & \cdots & \text{Cre}_{1j} & \cdots & \text{Cre}_{1n} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ \text{Cre}_{i1} & \text{Cre}_{i2} & \cdots & \text{Cre}_{ij} & \cdots & \text{Cre}_{in} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ \text{Cre}_{m1} & \text{Cre}_{m2} & \cdots & \text{Cre}_{mj} & \cdots & \text{Cre}_{mn} \end{pmatrix} \end{matrix}, \quad (7)$$

where each item Cre_{ij} has a compound data structure and is represented by a tuple of three elements: $\text{Cre}_{ij} = \langle F_{\text{num}_{ij}}, H_{\text{num}_{ij}}, \text{fam}_{\text{value}_{ij}} \rangle$. $F_{\text{num}_{ij}}$ is the number of query messages containing object o_i and forwarded by neighbor peer p_j . $H_{\text{num}_{ij}}$ is the number of query hit messages containing object o_i and forwarded by neighbor peer p_j . $\text{fam}_{\text{value}_{ij}}$ is calculated by formula (6) and denotes the credibility for neighbor p_j generated by familiarity for the query containing object o_i . $F_{\text{num}_{ij}}$ will be updated, if p_j is selected to forward the query containing object o_i . When the query is returned with hit by p_j , the value of $H_{\text{num}_{ij}}$ will be updated. At the same time, the value of $\text{fam}_{\text{value}_{ij}}$ will be updated according to formula (6). The dynamic update way in the search process adapts for dynamic characteristics of the unstructured P2P networks and avoids the bottleneck problem of the peer with higher credible. For example, when the value of $\text{fam}_{\text{value}_{ij}}$ is higher and there are multiple queries containing object o_i in peer, p_j will be multiselected. If the message is only to be forwarded, but not returned, then $F_{\text{num}_{ij}}$ will be increased and $H_{\text{num}_{ij}}$ not and thus the value of $\text{fam}_{\text{value}_{ij}}$ will be decreased. And then neighbor peers except p_j in CQM will have chances to forward the queries containing object o_i , thus reducing the burden of p_j and expanding the scope of the search. For the new query q_h containing object o_i , the credibility rate $\text{fam}(q_h, p_t, p_j)$ of p_t to its neighbor p_j by CQM is calculated as

$$\text{fam}(q_h, p_t, p_j) = \frac{|\text{Cre}_{ij}|}{\sum_{j=1}^n |\text{Cre}_{ij}|} = \frac{\text{fam}_{\text{value}_{ij}}}{\sum_{j=1}^n \text{fam}_{\text{value}_{ij}}}. \quad (8)$$

According to formula (2), on the basis of CQM, the credibility of p_t to its neighbor p_j generated by familiarity is shown as

$$\text{fam}(p_t, p_j) = \frac{\sum_{i=1}^m H_{\text{num}_{ij}}}{\sum_{i=1}^m F_{\text{num}_{ij}}}. \quad (9)$$

3.3. Update Mechanism. At the beginning of the search, both QOL and QPL are empty at each peer, as well as the corresponding CQM. Because there is no communication information between node and each of its neighbor nodes, the credibility of the familiarity to its each neighbor is 0 and the credibility of the similarity to its each neighbor is computed according to the formula (4). When a new query arrives,

peer firstly checks whether the requested object information is included in QOL; if not, the requested object information (keywords) will be joined in the QOL. When a neighbor node is selected to pass the query and the neighbor node is not in QPL, then it will be added in the QPL. At the same time, in CQM, the corresponding object row and node column information will be established and set the corresponding element $\text{Cre}_{ij} \cdot F_{\text{num}_{ij}} = 1$; if the requested object has been included in the QOL and the QPL contains the selected node, then the requested object position information $\text{position}(o_i)$ and the selected node position information $\text{position}(p_j)$ will be got from QOL and QPL, respectively, and then set the corresponding $\text{Cre}_{ij} \cdot F_{\text{num}_{ij}} = \text{Cre}_{ij} \cdot F_{\text{num}_{ij}} + 1$ in CQM. If the requested object has been included in QOL and the selected node is not in QPL, then the selected node will be added to QPL and the corresponding objects row and node column information will be established in CQM. After the query hit return happens, set $\text{Cre}_{ij} \cdot F_{\text{num}_{ij}} = 1$ and $\text{Cre}_{ij} \cdot H_{\text{num}_{ij}} = \text{Cre}_{ij} \cdot H_{\text{num}_{ij}} + 1$ in CQM and update $\text{Cre}_{ij} \cdot \text{fam}_{\text{value}_{ij}}$ according to formula (6). If the query returns failure, the corresponding information in QPL, QOL, and CQM will be deleted.

4. Search Algorithm CSA

4.1. Problem Analysis. In the existing heuristic search algorithm, such as APS, PQR, and SPUN, search mechanisms show that, when a node receives a query message, it firstly tests whether there is relevant information with the query in its index table. If there is, the query will be guided to forward according to the previous record information. And if there is no information, it will be forwarded based on random walks way. Thus, these methods are valid for the queries with high repetition rate. But for some queries with low repetition rate, such as a new query or query for rare resources, there is no or little heuristic guide information in these methods. At this moment, these methods are low efficient and their performances are equivalent to that of random walks. We simulate the three methods APS, PQR, and SPUN in experiments (experimental configuration given in Section 5.1); the simulation results are shown in Figure 1. In Figure 1, “total number” represents all number of selected neighbor nodes. “Random selection” means the number of neighbor nodes chosen randomly. “APS selection,” “PQR selection,” and “SPUN selection,” respectively, denote the number of neighbor nodes selected by the heuristic information given in the three algorithms. From Figure 1, we find that only about a third of neighbor nodes are selected according to the heuristic strategies given in the search processing and two-thirds of the neighbor node is still selected randomly in the three algorithms. Therefore, our motivation is to improve the search performance with high repetition rate as well as low repetition rate. In our method, queries firstly can be divided into familiar queries and strange (unfamiliar) ones and then using different neighbor node selection strategy to select neighbor peer forwarding different queries, respectively.

FIGURE 1: The comparison of the number of neighbor nodes selected.

4.2. Peer Selection Criteria. When a node receives a query message, if there is relevant information with the query in index table of this node, then the query is defined as familiar query. The familiar query will be processed according to previous search information. And if no, then the query is defined as strange query. Different from familiar query, there is no heuristic information for strange query in this node. Therefore, for strange query, our method firstly gets the credibility of the node to each neighbor node based on all the previous queries and the similarity between the requested object information and the contents stored on each neighbor node and then chooses the neighbor nodes with higher credibility to transfer the strange query. According to different query, familiar query, or strange query, the different credibility is described below. According to Sections 2 and 3, for a familiar query, the credibility of the node p_t to its neighbor node p_j is calculated as follows:

$$\begin{aligned} \text{Cre}(q_h, p_t, p_j) &= \text{fam}(q_h, p_t, p_j) \cdot r + \text{sim}(q_h, p_t, p_j) \\ &= \frac{\left| \text{SR}_{p_t \leftarrow p_j}^{K_{q_h}} \right|}{\left| F_{p_t \rightarrow p_j}^{K_{q_h}} \right|} + \frac{\left| K_{q_h} \cap K_{p_j} \right|}{\left| K_{q_h} \right|} \\ &= \frac{\text{fam}_{\text{value}_{ij}}}{\sum_{j=1}^n \text{fam}_{\text{value}_{ij}}} + \frac{\left| K_{q_h} \cap K_{p_j} \right|}{\left| K_{q_h} \right|}. \end{aligned} \quad (10)$$

For a strange query, the credibility of the node p_t to its neighbor node p_j is calculated as follows:

$$\begin{aligned} \text{Cre}(q_h, p_t, p_j) &= \text{fam}(p_t, p_j) + \text{sim}(q_h, p_t, p_j) \\ &= \frac{\left| \text{SR}_{p_t \leftarrow p_j} \right|}{\left| F_{p_t \rightarrow p_j} \right|} + \frac{\left| K_{q_h} \cap K_{p_j} \right|}{\left| K_{q_h} \right|} \\ &= \frac{\sum_{i=1}^m H_{\text{num}_{ij}}}{\sum_{i=1}^m F_{\text{num}_{ij}}} + \frac{\left| K_{q_h} \cap K_{p_j} \right|}{\left| K_{q_h} \right|}. \end{aligned} \quad (11)$$

In the search process, our method firstly determines a query whether it is a familiar query or a strange query. If it is a familiar query, we calculate the credibility of node to its neighbor nodes according to the formula (10). If it is a strange query, we compute the credibility of node to its neighbor nodes according to the formula (11) and then select the neighbor nodes with high credibility as passers. The proposed algorithm is described in Section 4.3.

4.3. Algorithm Description. The credibility search algorithm includes the process of the neighbor node selection and updating process, which is described in detail in Algorithm 1.

5. Experiments and Performance Evaluation

P2P networks are large-scale networks with millions of nodes, which join and leave frequently. The dynamic characteristics

TABLE 1: Simulation parameters.

Parameters	Value
Network size	10000
Kind of file (object)	100
Total number of files (object)	10000
Total number of queries	10000
Object distribution	Zipf (alpha = 0.8)
Beta in small-world network	0.5
walkers (k)	4

of P2P networks are challenging to deal with. It is not feasible to evaluate a new protocol in a real environment. To save time and improve efficiency, we use Peersim [28] as a simulation platform, which is a Java-based, open-source, large-scale P2P simulation platform and suitable for P2P dynamic characteristics. In this paper, we expand Peersim code for simulation and deploy four search strategies APS, PQR, SPUN, and CSA in the three most representative network topologies including random graph network structure (random graph), small world model network structure (small world), and scale-free network structure (scale-free). We evaluate the performance of the four search strategies in terms of network overhead, query hit rate, search delay, and the query hit rate for rare resources under static and dynamic network conditions.

5.1. Experimental Setting. Studies have shown that Gnutella, Napster, and Web users request tend to follow Zipf-like distributions [29]. In order to reveal the real network environment, the object popularity follows Zipf-like distribution in our experiments and is given by the formula [30]:

$$P_{\text{Zipf-like}}(x) = \frac{x^{-\alpha}}{\sum_{j=1}^C j^{-\alpha}}, \quad (12)$$

where C is the number of objects (resources) and α denotes the exponent characterizing of the distribution. Research [31] shows that α is usually between 0.6 and 0.8. x is the relative position of a resource or object. In experiments, we provide a total of 10,000 objects that are divided into 100 classes. Each class object is set up corresponding popularity based on the formula (12). We set the appropriate number of each type of object on the basis of its popularity and make objects or resources with high popularity to obtain a higher replication rate. Each object or resource is duplicated to a random node. As for queries, each node obtains query objects from 100 classes based on the popularity of objects. The objects with high popularity are more likely to be selected as the query objects. So the queries in our method can be more close to real networks. Table 1 shows the experimental parameters and their default values. And three kinds of network topologies, respectively, constructed, and node degree and its distribution are analyzed in our experiments. The results of the analysis are shown in Table 2.

Input: The relevant parameters, such as: query message number, network topology model, network size, TTL (Time-to-Live)

Output: Result (success or fail), the hops number of the first success return message, message information

Peer p_t gets a query message $meg \langle id, p_s, p_f, state, hops, K_{qh}, path \rangle$

If ($meg \cdot state = Forward$) {

If (K_{qh} hits at p_t) then { // the requested objects are included in p_t

$meg \cdot state = Success\_Return$;

Peer p_t sends successful meg to p_s along with the reverse $meg \cdot path$. }

Else if ($meg \cdot hops = TTL$) then { // the requested objects are not included in p_t

$meg \cdot state = Failure\_Return$;

Peer p_t sends failure meg to p_s along with the reverse $meg \cdot path$. }

Else if ($meg \cdot K_{qh} \in p_t \cdot QOL$) { // the query is familiar query

return $K_{qh} \cdot position i$ in $p_t \cdot QOL$;

calculates the value $fam(q_h, p_t, p_j)$ of each corresponding neighbor node of the i th row in $p_t \cdot CQM$ according to formula (6) and then gets the value $fam(q_h, p_t, p_j) \cdot r$ according to formula (10); selects the neighbor node p_j with higher credibility value and forwards meg to p_j ; updates

$Cre_{ij} \cdot F_{num_{ij}} = Cre_{ij} \cdot F_{num_{ij}} + 1$. }

Else { // $meg \cdot K_{qh} \notin p_t \cdot QOL$, the query is strange query

calculates the credibility value of each neighbor node according to formula (11) and selects the neighbor node p_j with higher credibility value and forwards meg to p_j ; adds the requested object and p_j to $p_t \cdot QOL$ and $p_t \cdot QPL$, respectively. }

End if }

Else if ($meg \cdot state = Success\_Return$) { // Updates the corresponding information in $p_t \cdot QPL$, $p_t \cdot QOL$ and $p_t \cdot CQM$

If ($p_t \cdot QOL \neq \emptyset$ and $K_{qh} \in p_t \cdot QOL$) then { returns $K_{qh} \cdot position i$ in $p_t \cdot QOL$. }

Else { $p_t \cdot QOL \cdot add(K_{qh})$; returns $K_{qh} \cdot position i$ in $p_t \cdot QOL$. }

End if

Peer $p_j \leftarrow meg \cdot p_f$

If ($p_t \cdot QPL \neq \emptyset$ and $p_j \in p_t \cdot QPL$) then { returns $p_j \cdot position j$ in $p_t \cdot QPL$. }

Else { $p_t \cdot QPL \cdot add(p_j)$; returns $p_j \cdot position j$ in $p_t \cdot QPL$. }

End if

If ($p_t \cdot CQM \cdot Cre_{ij} = \emptyset$) then { $p_t \cdot CQM \cdot add(Cre_{ij})$; $Cre_{ij} \cdot F_{num_{ij}} = Cre_{ij} \cdot H_{num_{ij}} = Cre_{ij} \cdot fim_{value_{ij}} = 1$. }

Else { $Cre_{ij} \cdot H_{num_{ij}} = Cre_{ij} \cdot H_{num_{ij}} + 1$; updates $Cre_{ij} \cdot fim_{value_{ij}}$ according to formula (9). }

End if

Peer p_t sends meg to p_s along with the reverse $meg \cdot path$. }

Else { // $meg \cdot state = Failure\_Return$, deletes the corresponding information in $p_t \cdot QPL$, $p_t \cdot QOL$ and $p_t \cdot CQM$

Peer $p_j \leftarrow meg \cdot p_f$

If ($p_j \in p_t \cdot QPL$ and $K_{qh} \in p_t \cdot QOL$) {

returns $K_{qh} \cdot position i$ in $p_t \cdot QOL$ and $p_j \cdot position j$ in $p_t \cdot QPL$;

updates $p_t \cdot CQM \cdot Cre_{ij} \cdot H_{num_{ij}} = p_t \cdot CQM \cdot Cre_{ij} \cdot H_{num_{ij}} - 1$;

If ($p_t \cdot CQM \cdot Cre_{ij} \cdot H_{num_{ij}} \leq 0$) then {

$p_t \cdot CQM \cdot remove(Cre_{ij})$ in CQM ;

For ($j = 1$ to n) do $p_t \cdot CQM \cdot sum(Cre_{ij} \cdot H_{num_{ij}})$ End for

If ($p_t \cdot CQM \cdot sum(Cre_{ij} \cdot H_{num_{ij}}) \leq 0$) then $p_t \cdot QOL \cdot remove(K_{qh})$ End if

For ($i = 1$ to m) do $p_t \cdot CQM \cdot sum(Cre_{ij} \cdot H_{num_{ij}})$ End for

If ($p_t \cdot CQM \cdot sum(Cre_{ij} \cdot H_{num_{ij}}) \leq 0$) then $p_t \cdot QPL \cdot remove(p_j)$ End if }

End if }

End if }

End if.

ALGORITHM 1: Credibility search algorithm (CSA).

5.2. Performance Evaluation

5.2.1. Performance Analysis in Static Network Environments.

In this section, we try to analyze the performance of four search strategies CSA, PQR, SPUN, and APS in static network environments where no peer or object departure or remove happens from the following several aspects.

(1) *Comparative Analysis of Query Hit Rate.* Figure 2 shows the query hit rates of the four methods: APS, PQR, SPUN, and CSA for different TTL value in three different network topologies. The four curves in Figure 2 show that the query hit rates are increasing with the increase of TTL value. The curve of PQR is similar to that of SPUN and slightly higher than SPUN. And the curves of PQR and SPUN are located in

TABLE 2: Analysis of node degree.

Topology	Min (degree)	Max (degree)	Avg. (degree)	Count min	Count max
Scale-free network	2	220	4	4993	1
Small world network	6	19	10	84	3
Random graph	6	26	12	19	1

FIGURE 2: Query hit rate versus TTL value in static network environments.

the middle of the curves of APS and CSA. The query hit rate of APS is the lowest in the four algorithms and that of CSA is highest. The query hit rate of CSA exceeds that of APS by about 22% and those of PQR and SPUN by about 9% and 10% when TTL value is greater than 5.

The rare resource query hit rates of the four algorithms are shown in Figure 3. The query hit rate of CSA for rare resources is significantly improved and better than those of the other three methods. Even compared to the PQR algorithm with better performance, the query hit rate for rare resources in CSA is still increased by about 20%. The main reason is that searches for rare resources are nonrepetitive search or less repetitive search; there is little or no historical experience used in the other three methods. So the other three methods can only adopt a random manner to forward query messages. However, in this paper, for nonrepetitive search or less repetitive search, we give full consideration for the overall success rate of the previous search experience and the similarity between the query itself and the contents stored in the neighbor node and obtain effective heuristic information to avoid blind random search, thus resulting in better performance in method CSA.

(2) *Comparative Analysis of Search Delay.* In our experiments, searching is based on the deployment of k walkers and k is set to 4. So each request may receive multiple query hits. In this paper, the search delay is defined as the *hops* value of hit message returned the first time successfully. Figure 4 displays the search delays of the four methods: APS, PQR, SPUN, and CSA for different TTL value in three different network topologies. The search delay of CSA is basically stable at 4 hops when TTL value is greater than 16. And those of the other three methods stepwise grow with the increase of TTL value. Meanwhile, Figure 4 shows the search delay of CSA is lower about 1 hop than those of the other three methods in the same range of TTL value when TTL value is less than 23. When TTL value is greater than 23, the advantage of CSA is more obvious.

(3) *Comparative Analysis of Average Number of Messages.* Figure 5 shows the average number of messages generated per query in different networks. The performance of CSA is also the best one in the four methods: PQR, SPUN, APS, and CSA. At the same time, the reduction rate of the average number of messages generated per query in CSA is indicated in Figure 5, in contrast to the best performance one in the other three methods PQR, SPUN, and APS. From Figure 5, we can see that the average number of messages per query of CSA is reduced 12.8%, 13.4%, and 12.9% in three different network topologies, respectively, when TTL value comes up to 25.

(4) *Comparative Analysis of Network Overhead.* In the search processing, network overhead is constituted by the number of messages generated per query and network bandwidth consumed by these messages. In this paper, we use the actual number of bytes of IP packets generated per message as network overhead.

By Section 3.1, the message structure in CSA is $meg\langle id, p_s, p_f, state, hops, K_{qh}, path \rangle$. In our experiments, id is made

up of 10 bytes. The information of node p_s and p_f is composed of "IP + port number," 6 bytes, a total of 48 bits. *hops* and *state* possess 2 bytes, respectively. 4 bytes are assigned to K_{qh} . The number of bytes of *path* is dynamic change in the search progress.

On the other hand, since the total number of bytes generated per message is small, it cannot produce fragmentation. And then message packet is a UDP packet. And thus, the total number of bytes generated by per message will contain a total of 28 bytes of the IP header and UDP header.

Figure 6 illustrates the comparison of network overhead among the four search strategies. Similar to Figure 5, we mark the reduction rate of the network overhead in CSA in contrast to the best performance one in the other three methods PQR, SPUN, and APS in Figure 6. Figure 6 shows that, when TTL value is 5, the network overhead of CSA is slightly higher than the best performance one of the other three methods PQR, SPUN, and APS by 0.2% and 1.0% in small world model network and scale-free network, respectively. But in the initial stage of the search, the entire bandwidth consumption is very low, so the small increase will not bring burden to the network. With the increase of TTL value, the network overhead of CSA is reduced continuously. When TTL value is 15 and 25, the reduction rate of the network overhead is up to the highest 10.3% and 17.0%, in scale-free network and random graph network, respectively. At this time, the performance of CSA is the best in the four algorithms.

5.2.2. *Performance Analysis in Dynamic Network Environments.* Here, we conduct a series of experiments similar to Section 5.2.1 to evaluate the performance of CSA under dynamic network environments. The total experimental runtime is divided into 100 time slices. At the end of each time slice, we add 10 new nodes and allocate 100 resources to these nodes according to the Zipf distribution. At the same time, 100 nodes from the network are selected randomly and each node deletes one resource from its resource list randomly. In this dynamic environment, we deploy four query strategies CSA, PQR, SPUN, and APS in three different network topologies; results shown in Figures 7, 8, and 9 and Tables 3 and 4.

Figure 7 shows the comparison of query hit rates of the four algorithms CSA, PQR, SPUN, and APS under dynamic network environments. From Figure 7, the query hit rate of CSA is the highest one in the four search strategies in three different network topologies. In the small world network, especially, the query hit rate of CSA exceeds PQR, SPUN, and APS by about 14%, 18%, and 26%, respectively.

Figure 8 illustrates the comparison of query hit rates of the four algorithms CSA, PQR, SPUN, and APS for rare resources. We can see the query hit rate of CSA algorithm is still the highest and it surpasses those of the other three methods by about 20% under dynamic network environments.

Figure 9 presents the comparison of the search delays of the four methods under dynamic network environments. The search delay of CSA is lower than the other three algorithms by about 1 hop in Figure 9. When TTL value is greater than 20, the search delay of CSA is stable at 6 hops while the other

FIGURE 3: Query hit rate of rare resources versus TTL value in static network environments.

three algorithms reflect continued stepwise growth trend with the increase of TTL value.

Table 3 shows the average number of messages generated per query in the three different kinds of network topologies. It consists of data of two-hop interval in the TTL value from

1 to 25. So the TTL value is a sequence of 1, 4, 7, ..., 25 in Table 3. As can be seen from Table 3, in the four methods, the average number of messages generated per query is basically the same, when the TTL value is less than or equal to 7. However, when the TTL value is greater than or equal to 10,

FIGURE 4: Search delay versus TTL value in static network environments.

the average number of messages generated per query in CSA is lower than the other three methods at most by about 9.6%.

Table 4 illustrates the network overhead generated by the four search strategies. Similar to Table 3, we also extract experimental data of two-hop interval among 1 to 25 hops

to form Table 4. From Table 4, we can see that the network overhead of CSA is less than or equal to those of the other three methods with the increase of TTL value and those of the other three methods are basically the same. At the same time, after the 10th hops, the network overhead of CSA is reduced

FIGURE 5: Average number of messages generated per query versus TTL value in static network environments.

by about 11.2% at most compared to the others in the three different network topologies.

In summary, we can see that the performances of the four algorithms are reduced to some extent in the dynamic

network environments. For example, in the dynamic environments, the query hit rates of the four search strategies CSA, PQR, SPUN, and APS for scarce resources are reduced on average by about 20%, compared with the static network

FIGURE 6: Network overhead versus TTL values in static network environments.

environments as shown in Figures 3 and 8. And, although the search delays of the four strategies are also stepwise growth similar to the static network environments, but its gradient is larger as shown in Figures 4 and 9. The main reason is that

the churn of network influences the performances of searches, but which has less effect on the CSA algorithm compared with the other three ones. For example, the query hit rate of CSA drops slightly in Figure 7 and less than 7%

FIGURE 7: Query hit rate versus TTL value in dynamic network environments.

compared to that in Figure 2. Moreover the query hit rates of the other three methods are reduced by more than 10%. At the same time, in the dynamic network environments, CSA method shows better performance than the other three

algorithms, compared with static environments. For example, in the static network environments, the query hit rate of CSA is improved by about 9%, 10%, and 22% compared to the algorithm PQR, SPUN, and APS in Figure 2 and in the dynamic

FIGURE 8: Query hit rate of rare resources versus TTL value in dynamic network environments.

FIGURE 9: Search delay versus TTL value in dynamic network environments.

TABLE 3: Average number of messages generated per query versus TTL value in dynamic network environments.

TTL	Small world				Random graph				Scale-free			
	CSA	PQR	SPUN	APS	CSA	PQR	SPUN	APS	CSA	PQR	SPUN	APS
1	8	8	8	8	8	8	8	8	8	8	8	8
4	20	20	20	20	20	20	20	20	20	20	20	20
7	31	32	31	31	30	32	31	31	31	32	31	31
10	41	43	42	42	40	43	42	42	41	43	42	42
13	50	53	52	53	49	53	52	53	50	52	53	53
16	58	63	62	63	58	63	62	63	58	62	62	63
19	66	72	71	72	67	72	72	72	65	71	72	72
22	73	81	80	82	74	80	81	82	72	79	80	82
25	80	88	89	90	80	88	89	91	78	86	88	90

TABLE 4: Network overhead versus TTL value in dynamic network environments (M).

TTL	Small world				Random graph				Scale-free			
	CSA	PQR	SPUN	APS	CSA	PQR	SPUN	APS	CSA	PQR	SPUN	APS
1	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7
4	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	2.0	1.9	1.9
7	3.3	3.4	3.3	3.3	3.3	3.4	3.3	3.3	3.3	3.4	3.3	3.3
10	4.8	5.1	4.9	4.9	4.8	5.1	4.9	4.9	4.8	5.0	4.9	4.9
13	6.4	6.8	6.7	6.7	6.3	6.9	6.7	6.8	6.4	6.8	6.7	6.7
16	8.0	8.8	8.6	8.7	8.1	8.8	8.7	8.7	8.0	8.6	8.6	8.7
19	9.8	10.9	10.6	10.8	9.9	10.8	10.7	10.8	9.5	10.6	10.7	10.8
22	11.5	13.0	12.8	13.1	11.7	12.9	12.9	13.1	11.3	12.6	12.8	13.1
25	13.4	15.1	15.1	15.3	13.4	15.0	15.1	15.5	12.9	14.6	15.0	15.4

environment increased by about 14%, 18%, and 26% in Figure 7. Therefore, whether it is under the dynamic or static environments, CSA can achieve high query hit rate with less network overhead and lower search delay. So, CSA can get the better performance than the other three algorithms PQR, SPUN, and APS.

5.3. Discussion. In CSA, the space cost is mainly composed of four indexes CNL, CQM, QPL, and QOL in each node. CNL, QPL, and QOL are linear lists. The number of elements in CNL is equal to the number of all neighbors of a node. The length of QPL does not exceed the length of CNL. At the same time, the length of QOL is also not large because it records only different query objects. CQM is a two-dimensional matrix. If the length of the QOL is m and the length of the QPL is n then the length of CQM is $m \times n$. So, the space cost of CNL, QPL, and QOL is negligibly small compared with that of CQM in each node because of their linear length. Thus the space complexity of each node in CSA is $o(m \times n)$ which is the same with the PQR and APS method, less SPUN ($o(m \times n \times \text{TTL})$) algorithm. Meanwhile, such space cost is not a burden for computing device in current P2P networks. However, if the storage capacity of node is very low, the node can choose a small size of CQM and update it according to first in first out (FIFO) policy or least recent used (LRU) policy.

In the search process, the worst case is that all the k walkers travel TTL hops and then return a hit or miss message along query path. Therefore the number of messages generated per query is $2 \times k \times \text{TTL}$ in the worst case and which is the same as the other three methods: PQR, SPUN, and APS. But simulation experiments in this paper have shown that CSA produced less number of messages and lower network consumption compared with the other methods, whether it is in a dynamic P2P environment or in a static P2P environment. The main reasons lie in two aspects. On the one hand, the credibility generated on the basis of familiarity and similarity as the heuristic information is very effective to guide the future searches, which improves the query hit rate, reduces the search delay, and shortens the length of the search path, resulting in reducing the number of messages generated and the network consumption. On the other hand, it is reasonable that queries are classified. At the same time, the credibility information provided in CSA is trustworthy, which guides effectively the strange queries to the requested objects, and superior to the random walks way in the other three methods. The improvement of scarce resources query hit rates in CSA may succeed to verify it in our experiments.

CSA can make full use of the advantages of the previous informed algorithms such as APS, PQR, and SPUN, using the previous search information to guide the future searches.

Thus, CSA is similar to these methods on the handling of the queries with high repetition rate. The difference is that, in the acquisition process of heuristic information, APS, PQR, and SPUN methods only consider hit information of successful path, while in CSA the similar information between the queried contents and contents stored on node is also considered as the heuristic information to guide the future searches. So, the query message is forwarded to nodes that are more likely to provide the necessary resources node in CSA.

The main difference between CSA and APS, PQR, and SPUN is that queries are classified as familiar queries and strange (unfamiliar) ones according to the similarity between the requested resources and those have been received in the query node in CSA. For the strange queries, there is no or little heuristic guide information in APS, PQR, and SPUN methods and the queries will be forwarded based on random walks way. However, in CSA, based on the trust production principle in sociology and psychology, CSA method firstly gets the credibility of the node to each neighbor node according to all the previous queries and the similarity between the requested object information and the contents stored on each neighbor node and then selects the nodes with higher credibility to forward the strange queries. In contrast to random manner in APS, PQR, and SPUN methods, the one in CSA greatly reduce the search blindness.

The main feature of CSA method is that it makes full use of the query information and resources information nodes themselves hold. Based on node local information, the searches are effectively forwarded. This method does not require too complicated structure and does not need to track search path information, so it has a good adaptability for dynamic characteristics of unstructured P2P networks.

6. Conclusion

In this paper, a credibility search algorithm (CSA) has been presented. The main feature of this method is that it can improve query performance in unstructured P2P networks. CSA can gain the effective heuristic information and credibility of node to its neighbor by combining with the trust production principle in sociology and psychology, so that the familiar query and the strange query can be guided successfully. Experimental results show that the proposed algorithm outperforms the other three methods: PQR, SPUN, and APS furthermore can achieve high query hit rate with less search delay and lower bandwidth consumption in three different types of network topologies under static and dynamic network conditions. At the same time, CSA is also very effective for the search of rare resources. In the scale-free network especially, the query hit of CSA for rare resources can reach up to about 85% when TTL value comes up to 24. Compared to PQR and APS, the query hit of CSA for rare resources is increased by about 20% and 40%, respectively, with the increase of TTL value in three different network topologies.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This study is supported by the National Natural Science Foundation of China (Grant no. 60872051) and the Program of the Co-Construction with Beijing Municipal Commission of Education of China.

References

- [1] Open Source Community, "Gnutella," 2001, <http://eg.wego.com/>.
- [2] FastTrack Peer-to-Peer Technology Company, "Fast track," 2001, <http://fasttrack.nu/>.
- [3] KaZaA lesharing network, "KaZaA," 2002, <http://www.kazaa.com/>.
- [4] D. Stutzbach, R. Rejaie, N. Duffield, S. Sen, and W. Willinger, "Sampling techniques for large, dynamic graphs," in *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM '06)*, pp. 1–6, April 2006.
- [5] A. H. Rasti, D. Stutzbach, and R. Rejaie, "On the long-term evolution of the two-tier Gnutella overlay," in *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM '06)*, pp. 1–6, April 2006.
- [6] B. Yang and H. Garcia-Molina, "Improving search in peer-to-peer networks," in *Proceedings of the 22nd IEEE International Conference on Distributed Computing (ICDCS '02)*, pp. 5–14, IEEE, July 2002.
- [7] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, "Search and replication in unstructured peer-to-peer networks," in *Proceedings of the International Conference on Supercomputing*, pp. 84–95, June 2002.
- [8] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker, "Making gnutell-like p2p systems scalable," in *Proceeding of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM '03)*, pp. 407–418, August 2003.
- [9] M. Andreolini and R. Lancellotti, "A flexible and robust lookup algorithm for P2P systems," in *Proceedings of the 23rd IEEE International Parallel and Distributed Processing Symposium (IPDPS '09)*, pp. 1–8, May 2009.
- [10] R. Gaeta and M. Sereno, "Generalized probabilistic flooding in unstructured peer-to-peer networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 12, pp. 2055–2062, 2011.
- [11] C. Gkantsidis, M. Mihail, and A. Saberi, "Random walks in peer-to-peer networks," in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '04)*, vol. 1, pp. 7–11, March 2004.
- [12] N. Bisnik and A. A. Abouzeid, "Optimizing random walk search algorithms in P2P networks," *Computer Networks*, vol. 51, no. 6, pp. 1499–1514, 2007.
- [13] C. Gkantsidis, M. Mihail, and A. Saberi, "Hybrid search schemes for unstructured peer-to-peer networks," in *Proceedings of 24th Annual Joint Conference of the IEEE Computer and*

- Communications Societies (INFOCOM '05)*, vol. 3, pp. 1526–1537, Miami, Fla, USA, March 2005.
- [14] S. Jiang, L. Guo, X. Zhang, and H. Wang, “LightFlood: minimizing redundant messages and maximizing scope of peer-to-peer search,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 5, pp. 601–614, 2008.
  - [15] H. Y. Mei, Y. J. Zhang, X. W. Meng, and W. M. Ma, “Limited search mechanism for unstructured peer-to-peer network,” *Journal of Software*, vol. 24, no. 9, pp. 2132–2150, 2013.
  - [16] V. Kalogeraki, D. Gunopulos, and D. Zeinalipour-Yazti, “A local search mechanism for peer-to-peer networks,” in *Proceedings of the 11th International Conference on Information and Knowledge Management (CIKM '02)*, pp. 300–307, McLean, Va, USA, November 2002.
  - [17] D. Tsoumakos and N. Roussopoulos, “Adaptive probabilistic search in peer-to-peer networks,” in *Proceedings of the 3rd International Conference on Peer-to-Peer Computing (P2P '03)*, pp. 102–109, September 2003.
  - [18] M. Xu, S. Zhou, J. Guan, and X. Hu, “A path-traceable query routing mechanism for search in unstructured peer-to-peer networks,” *Journal of Network and Computer Applications*, vol. 33, no. 2, pp. 115–127, 2010.
  - [19] D. M. R. Himali and S. K. Prasad, “SPUN: a P2P probabilistic search algorithm based on successful paths in unstructured networks,” in *Proceedings of the 25th IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW '11)*, pp. 1610–1617, May 2011.
  - [20] B. Zuo and Q. Gao, “The effects of familiarity and similarity on the interpersonal attraction,” *Chinese Journal of Clinical Psychology*, vol. 16, no. 6, pp. 634–636, 2008.
  - [21] W. Ma, X. Meng, and Y. Zhang, “Bidirectional random walk search mechanism for unstructured P2P network,” *Journal of Software*, vol. 23, no. 4, pp. 894–911, 2012.
  - [22] G. Chen, C. P. Low, and Z. Yang, “Enhancing search performance in unstructured P2P networks based on users’ common interest,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 6, pp. 821–836, 2008.
  - [23] K. C. Lin, C. Wang, C. Chou, and L. Golubchik, “SocioNet: A social-based multimedia access system for unstructured P2P networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 7, pp. 1027–1041, 2010.
  - [24] C. G. M. Snoek, M. Worring, J. C. van Gemert, J.-M. Geusebroek, and A. W. M. Smeulders, “The challenge problem for automated detection of 101 semantic concepts in multimedia,” in *Proceedings of the 14th Annual ACM International Conference on Multimedia (MULTIMEDIA '06)*, pp. 421–430, October 2006.
  - [25] M. S. Lew, N. Sebe, C. Djeraba, and R. Jain, “Content-based multimedia information retrieval: State of the art and challenges,” *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 2, no. 1, pp. 1–19, 2006.
  - [26] C. J. Lin, S. C. Tsai, Y. T. Chang, and C. F. Chou, “Enabling keyword search and similarity search in small-world-based P2P systems,” in *Proceedings of the 16th International Conference on Computer Communications and Networks (ICCCN '07)*, pp. 115–120, August 2007.
  - [27] G. Salton, *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*, Addison-Wesley, 1988.
  - [28] “The peersim simulator,” 2007, <http://peersim.sourceforge.net/>.
  - [29] K. Sripanidkulchai, “The popularity of Gnutella queries and its implications on scalability,” 2001, <http://www.cs.cmu.edu/~kunwadee/research/p2p/paper.html>.
  - [30] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, “Web caching and Zipf-like distributions: evidence and implications,” in *Proceedings of the 18th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '99)*, pp. 126–134, March 1999.
  - [31] P. Backx, “A comparison of peer-to-peer architectures,” in *Proceedings of the Eurescom 2002 Powerful Networks for Profitable Services*, pp. 1–8, 2002.