2011 Sixth Annual ChinaGrid Conference

Cooperation based P2P Topology Adaptation Scheme towards Improving Search Performance

Aibo Song, Jingya Zhou, Junzhou Luo
School of Computer Science and Engineering
Southeast University
Nanjing, P.R. China
{absong, jyz, jluo}@seu.edu.cn

***Abstract*—Topology adaptation as a widely used approach to enhance search performance has been studied from many ways (e.g. based on interests, lifecycle and so on). This paper presents a cooperation based peer-to-peer topology adaptation scheme (CPTA) which is from contributors point of view. All nodes are classified as providers or guiders according to their contributions to success ratio. Algorithms for constructing cooperative alliance (CA) are designed to enhance cooperation between roles so as to maximize their contributions. We present an analytical model on contributions and expected search performance. Simulation results further demonstrate the efficiency of our scheme.**

Keywords- peer-to-peer; topology adaptation; cooperative alliance; search Performance

I. INTRODUCTION

Peer-to-Peer (P2P) provides a kind of effective approach for data sharing in a large-scale heterogeneous environment. In P2P networks, all participating nodes form a P2P topology over a physical network. P2P topology is an abstract, logical network called overlay, in which each node only keeps in touch with a limited number of neighbors, and there is no node act as central server storing index structure, which makes searching data in an efficient and scalable way become one of the most challenging issues in P2P research.

From architectural perspective P2P network is mainly classified into structured P2P and unstructured P2P. Structured P2P like Chord [1], Tapestry [2] organizes nodes according to some strict rules. The rules define relation between the data and the node position, through DHT users can route to the node that possesses the required data within O(logN) hops. However, DHT only supports exact-match query rather than complicated search, furthermore, frequent node churn in P2P network leads to high overhead for the maintenance of structured topology. Distinguished from structured P2P, nodes in unstructured P2P connect with each other without any strict rules on topology. Decentralized unstructured P2P is especially a good candidate for serving large numbers of Internet users because it is resilient to nodes' entering and departure. For example, Gnutella [3] as the most popular file sharing network on Internet is a fully decentralized unstructured P2P network. We concern unstructured P2P topology in this paper.

When a query is issued, it is transmitted among the nodes until being satisfied by one node. In this process, there are three kinds of roles that a node plays: requestor, provider and guider. Requestor is the sponsor of a query, provider affords data that are queried by requestor, and guider helps requestor to route its query to the provider on which the target data are possessed. From the system point of view, the roles of provider and guider are the contributors to search performance. The key issue in the study of search is how to make these roles work cooperatively. Different from the previous studies which are usually from the perspective of requestor, we propose to optimize topology by combining providers and guiders together as a cooperative alliance (CA) so as to strengthen collaboration between them and improve search performance further.

There have been multiple studies on how to enhance search performance by changing P2P topology. Cooper [4] put forward the square-root topology to improve routing performance. However, assumptions that each data type has only one replica and there is no limit on hop count are not realistic. We have considered the case of multiple replicas and *TTL* (time to live) limit. Adamic *et al.* [5] suggest that search mechanism should bias its forwarding toward the high degree nodes at each step. Thus the high connectivity nodes will receive more queries, but more queries do not indicate higher hit ratio. If matching failure the high connectivity nodes will propagate more query messages. In fact, the hit ratio lies on the data provided by node and data query frequency. For improving search performance, it is recommended to draw lessons from structured P2P and establish some relation between data and node topological attributes (e.g. node degree). Unlike from strict rules in structured P2P, the relation is loose and indeterminate. G. Chen *et al.* [12] seek to construct a small-world network [13] through exploiting users' common interest patterns, so as to enhance search performance, but it cannot adapt well to the frequent changes of users' interests. We propose to calculate nodes' contributions according to their capacities and data possessed so as to distinguish their roles and modify the topology. Feng *et al.* [6] study the relation among the distribution of the degree, the query frequency and the success ratio, and propose an optimal distribution model of degrees to guide the topology construction. However, it primarily considers the case that each node only possesses one kind of data. The calculation of optimal degree for case of multiple data is simply the addition of degree of single data, and it neglects superposition of multi-data node degree.

Furthermore, it does not discuss how to construct overlay topology based on the model (e.g. neighbor selection). We propose the self-organization algorithms for constructing CA. Chawathe and Lv *et al.* [7, 8] propose a self-organization topology. Each node evaluates its neighbors based on the capability of dealing with queries then connects with those one with high values. The self-organization mechanism improves performance from the view of enhancement of forwarding efficiency. Nevertheless, it merely regards every node as a selfish one, and each one wants to maximize its own success ratio from requestor's view, the optimal topology will not always be achieved. Instead, from contributor's view, we suggest to maximize nodes' contributions by construct CA and give a probabilistic model to estimate the expected search performance.

The paper is organized as follows. Sec.II presents a topology adaptation scheme to improve performance. Sec.III gives the detailed theoretical analysis for the performance. In Sec.IV simulations are performed to verify the validity of our scheme. Finally, we conclude this paper in Sec.V.

II. Cooperation Based P2P Topology Adaption

A. Cooperative Alliance (CA)

As we have mentioned, each node in a P2P network takes the role of contributor. When a node acts as a provider by satisfying a query it makes direct contribution (C_d) to the success ratio which is represented by the probability of satisfying a query during a search, and it makes indirect contribution (C_{id}) which is represented by the probability of forwarding a query successfully when acts as a guider by routing query to the hit node. During a successful search, each node along the successful path except the query node and hit node will gain the indirect contribution, so in order to make two kinds of contributions comparable, indirect contribution should be divided by the number of forwarding nodes along the path. We use contribution values to identify node's role, for example, if the direct contribution of a node is larger than its indirect contribution, its primary role is provider and secondary role is guider, here we call it provider instead, and vice versa.

Which role a node chooses to play depends on three factors: the data possessed by it, its position over the topology and its capacity which is used to describe the ability of a node to maintain links with its neighbors and deal with queries. If each node in a P2P network can find a suitable location according to its data popularity and capacity to maximize its contributions, which implies providers satisfy more queries issued by requestors and guiders lead more queries to the right providers, we think that the P2P network forms a CA in which providers and guiders mutually cooperate to enhance search performance.

B. Topology Adaptation Algorithms

Before presenting details of our adaptation algorithms, we first explore how a node manages to maximize its contributions. When node acts as a provider, we focus on its direct contribution which lies on the hit ratio and its capability of receiving query during a search, and the hit ratio is tightly related to the query frequency of data possessed by the node, while its capability of receiving query is related to its position. In order to improve its direct contribution, we need to enhance its capability of receiving query by change its position (equals to change its neighbors). It reminds us of the responsibility of guider, so the provider is always inclined to select high indirect contribution node as its neighbor. When node acts as a guider, we focus on its indirect contribution which lies on its capability of receiving query and the probability that the query is satisfied by the following nodes arrived. On the one hand, like a provider the guider is also inclined to choose neighbors with high indirect contribution so as to enhance the capability of receiving query. On the other hand, the guider is inclined to choose neighbors with high direct contribution in order to enhance the probability that the query be satisfied.

Algorithm 1 Update_neighbors (node *i*) // *i* ∈ Path()

```
1    if i is a provider  //Path() is the set of all nodes along the path
2      for each node j ∈ {N(i)+Path()} //N(i) is i's neighbor set
         select j as i's new neighbor with the probability of
         $[C_{id}(j)/\overline{C_{id}}]$[1]  // $\overline{C_{id}}$ is the average indirect contribution
3      end for
4      if $d_i > num_i$   //overload  $d_i$ is i's degree
5        overload_handle(i)
6      end if
7    else        //i is a guider
8      for each node k∈ {N(i)+Path()}
9        if k is a provider
10         select k as i's new neighbor with the probability of
           $[C_d(k)/\overline{C_d}]$  // $\overline{C_d}$ is the average direct contribution
11       else        //k is a guider
12         select k as i's new neighbor with the probability
           of $[C_{id}(k)/\overline{C_{id}}]$
13       end if
14     end for
15     if $d_i > num_i$  //overload
16       overload_handle(i)
17     end if
18   end if
```

In terms of the aforesaid principles, we propose a probabilistic adaptation approach to implement the contribution-oriented choice preference of node. Whenever a query is finished successfully, all nodes along the path from the requestor to the hit node update their contributions and set of neighbors, which is described by algorithm 1. Before updating the set of neighbors, node *i* estimates its role according to the latest contribution values. If node *i* is a provider, it is inclined to choose neighbors with high indirect contribution, so we let node *i* choose new neighbors with probability of $[C_{id}(j)/\overline{C_{id}}]$ from the set of its previous neighbors and nodes along the path. Here $C_{id}(j)$ is node *j*'s indirect contribution and $\overline{C_{id}}$ is the average indirect

1 $$[A/B]=\begin{cases}1 & ,\ A/B \ge 1\\ A/B & ,\ A/B < 1\end{cases}$$

contribution, and the calculation of contributions is described in Sec.III. Otherwise, node i is a guider, and then its choosing strategy depends on the opposite node's role. If the opposite node k is a provider, its probability of being chosen as one of i's new neighbors should be proportional to its direct contribution which is set to $[C_d(k)/\overline{C_d}]$, if not the probability should be proportional to k's indirect contribution which is set to $[C_{id}(k)/\overline{C_{id}}]$.

In addition, we also take node capacity heterogeneity into account, which is assumed to be tightly related to node's ability to maintain links with its neighbors and deal with queries, for simplicity, node i's capacity is defined as the maximal number of links that it could maintain (num_i). According to our algorithm providers with high C_d will get more links, however, some of them may not have enough capacity to maintain these links. We use algorithm 2 to deal with the situation of overloading. When provider i's links exceed its capacity value, it should drop some links to neighbors with relative low indirect contributions. As we know high C_d derives partly from high hit ratio of data on the node, it is suggested to find a competent node j and exchange the most used data with j's least used data in order to reduce the probability of overloading again. Like searching process, the process of finding a competent node uses flooding algorithm with *TTL* limit. It should also be noted that guider's C_{id} has nothing to do with its data, so there is no need to exchange data but drop some links directly whenever a guider overloads.

Algorithm 2 overload_handle(node *i*) //*i* is an overloaded node

```
1  if i is a provider
2    sort descending nodes belong to N(i) according to $C_{id}$, disconnect
       links with the last $d_i - num_i$ neighbors
3    find a node j satisfy $num_j - d_j > d_i - num_i$
4    if j exists
5      exchange i's most used data with j's least used data
6    end if
7  else        //i is a guider
8    sort descending nodes belong to N(i) according to $C_d$,
     disconnect links with the last $d_i - num_i$ neighbors
9  end if
10 return new N(i)
```

C. Accounting for Hot Spots

As we know, in a P2P network data access frequency usually exhibits considerable diversity. There exists a small part of data that are required with very high frequency, these data is known as hot data and the node possessing hot data is called hot spot. Like Guntella [10] we use Zipf's law [9] to describe data popularity. In terms of Zipf's law, the majority of queries aim at hot data, if these queries can be route to hot spots, hot spots will achieve higher C_d and make greater contribution to success ratio. Therefore, hot spots should be apt to undertake the responsibility of provider.

Proposition 1: Hot spot should be a provider rather than a guider.

proof: Assume that there exists a hot spot i whose role is a guider, according to analysis above, i's C_{id} lies not on its data but its position and capacity, so we can find a non-hot spot j such that j's capacity is equal to i's to exchange position with i, and make j be a guider like i. Let's take a look at j's previous role, if j is a guider, then i can undertake the responsibility of guider instead of j, and there is no performance loss. Otherwise, j is a provider whose C_d lies on its position, capacity and data, now i has the same position and capacity with the non-exchanged j, the only difference is data. Assume that the hit ratio of non-exchange j and i are x and y respectively ($y>x$, for i is hot spot), and forwarding success ratio is p $(0 \le p \le 1)$, then the success ratio achieved from j before exchange is $x+(1-x)p$, while the success ratio achieved from i after exchange is $y+(1-y)p$, make a subtraction we get $(x-y)(1-p)<0$. The search performance is improved after exchange, so hot spot should be a provider rather than a guider. □

When two hot spots with different hit ratios connect, the lower one will be apt to be a guider because it may guide more queries to the higher one, which is contrary to proposition 1. So we revise adaptation algorithm by adding judgment when hot spot selects neighbors. Only the guider will hot spot connect to with the probability of $[C_{id}(k)/\overline{C_{id}}]$.

III. CPTA Analytical Model

Suppose that the P2P topology consists of N nodes and L links, and the average node degree $\bar{d}$ should be $2L/N$. Node possesses m_i different kinds of data ($m_i \ll N$), and the total number of data types is r. To avoid the impact of replica on search, we assume that each data type has the same number of replicas, and is distributed uniformly in the network. Nodes are randomly chosen with the same probability to issue queries, and the search method is flooding with *TTL* limit.

A. Contributions Calculation

As we have mentioned, each node has two kinds of contributions no matter what its primary role is, direct contribution when act as a provider and indirect contribution when act as a guider. Both roles vary in the probability of receiving queries during a search. When node s issues a query, all nodes as provider within s's valid area ($V_{TTL}(s)$) can receive the query and nodes as guider only within s's valid forwarding area ($V_{TTL-1}(s)$) can receive and forward the query. Eq. 1 is the expression of the probability (p_u) that node u receives queries during a search as a provider.

$$p_u = d_u\left(\frac{\bar{d}}{2L}w(s) + \sum_{k\in V_{TTL-1}(s)} \frac{\bar{d}-1}{2L}w(k)\right) \tag{1}$$

$$w(k) = \begin{cases} [C_{id}(k)/\overline{C_{id}}] & k \text{ is a provider} \\ [C_d(k)/\overline{C_d}] & k \text{ is a guider} \end{cases}$$

where $\frac{\bar{d}}{2L}w(s)$ represents the probability that u connects with s through one of its links, and $\sum_{k\in V_{TTL-1}(s)} \frac{\bar{d}-1}{2L}w(k)$ represents the probability that u connects with nodes that be visited during a search through one of its links. It is noted that node u may receive multiple queries during a search but only the first one is valid. Assuming any node will not forward query to a certain node for more than one time during a search, then the probability that u has been visited n times is defined as

$$p_u(n) = \binom{d_u}{n} \left(\frac{\bar{d}}{2L} w(s) + \sum_{k \in V_{TTL-1}(s)} \frac{\bar{d}-1}{2L} w(k) \right)^n \tag{2}$$

u has been visited n times through its n links, $\binom{d_u}{n}$ represents all possible combinations of n links. Then the effective probability that node u receives queries during a search as a provider is

$$Ep_u = \sum_{n=1}^{d_u} (p_u(n)/n) \tag{3}$$

Assume that q_i is the access frequency of data type i, the hit ratio on u should be $Q_u = \sum_{i \in u}^{m_u} q_i$, multiply it by Eq. 3 we get u's direct contribution

$$C_d(u) = Ep_u\, Q_u \tag{4}$$

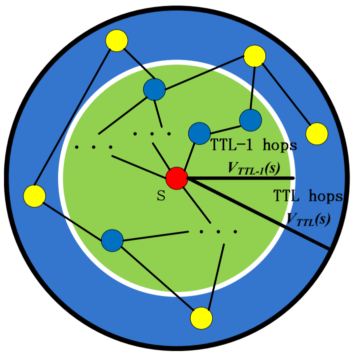


Figure 1. Valid area($V_{TTL}(s)$) and valid forwarding area($V_{TTL-1}(s)$)

In the same way, the probability (pi_u) that node u receives queries during a search as a guider is described by Eq. 5.

$$pi_u = d_u \left(\frac{\bar{d}}{2L} w(s)(1-Q_u) + \sum_{k \in V_{TTL-2}(s)} \frac{\bar{d}-1}{2L} w(k)(1-Q_u) \right) \tag{5}$$

where the expression in brackets represents the probability that u receives query from one of its links and forwards it. Then the effective probability that node u receives queries during a search as a guider is

$$Epi_u = \sum_{n=1}^{d_u} (pi_u(n)/n) \tag{6}$$

$$pi_u(n) = \binom{d_u}{n} \left(\frac{\bar{d}}{2L} w(s)(1-Q_u) + \sum_{k \in V_{TTL-2}(s)} \frac{\bar{d}-1}{2L} w(k)(1-Q_u) \right)^n$$

where $pi_u(n)$ *is* the probability that u is visited n times. After visiting u there are a portion of nodes within $V_{TTL}(s)$ have not been visited, $V_{TTL}(s\text{-}u)$ is used to represent the set of these nodes. So the probability that the query is matched after forwarded by u is $\sum_{j \in V_{TTL}(s-u)} Q_j$, the average hops taken to find the data is T which is discussed in the next part. Eq. 7 gives u's indirect contribution.

$$C_{id}(u) = \frac{Epi_u}{T} \sum_{j \in V_{TTL}(s-u)} Q_j \tag{7}$$

Thus, the average contributions are calculated by

$$\overline{C_d} = \sum_{i=1}^{N} \frac{C_d(i)}{N} \qquad \overline{C_{id}} = \sum_{i=1}^{N} \frac{C_{id}(i)}{N} \tag{8}$$

In terms of Eq. 8 each node needs the public information $\{C_d(i), C_{id}(i) | i \in 1\ldots N\}$ to calculate the average contributions. We propose a distributed approach to send and receive the information. According to our approach, each node has two tables: table online and table offline shown in Figure 2. Table online is used to record all online nodes' contributions and node i's online table is initially filled out by its own contributions, while table offline is used under the condition of dynamics to record the departed nodes. Assume that each node maintains connections to its neighbors through sending and receiving heartbeat packets periodically. For any neighbor of node i, say node j, if node i has not received any packets from it for a given period, node i will think node j has departed and add it into table offline. In order to collect more information to update the tables, the tables are sent out accompanied by queries. We use Bloom Filters [14] of tables (BF_{on}, BF_{off}) instead of raw tables to reduce the traffic overhead. The process of tables updating is described below:

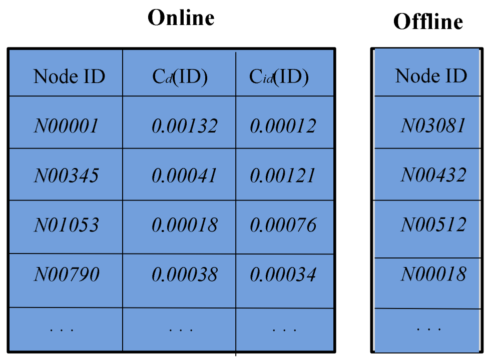
1) Each node uses BF_{off} to update its BF_{on} by eliminating the items appeared in BF_{off}, denoted by BF_{on}- BF_{off}.

2) When node j receives node i's BFi_{on}, BFi_{off}, then merge both nodes' Bloom Filters,

$$BFi_{on} \cup BFj_{on}, \quad BFi_{off} = BFj_{off} = BFi_{off} \cup BFj_{off}$$

and eliminate the offline items

$$BFi_{on} = BFj_{on} = BFi_{on} \cup BFj_{on} - BFi_{off}$$



Online

Node ID	C_d(ID)	C_{id}(ID)
N00001	*0.00132*	*0.00012*
N00345	*0.00041*	*0.00121*
N01053	*0.00018*	*0.00076*
N00790	*0.00038*	*0.00034*
. . .	. . .	. . .

Offline

Node ID
N03081
N00432
N00512
N00018
. . .

Figure 2. An example of two tables

In P2P networks, each node is aware of local information achieved from its neighbors. Through repeating the above updating process our approach utilizes the propagation and merge of local information to obtain the global information approximately. Consider the dynamic situations, node churn including node's joining and departure can be perceived by its neighbors, then they will update their own Bloom Filters of tables and send the information out accompanied by queries. Our approach uses two ways to update tables, one way is active mode, a node send out Bloom Filters of tables with queries for updating; another one is passive mode, which implies a node passively receives Bloom Filters of tables accompanied by queries from other nodes and merges together. Hence the refresh rate is fast, which also guarantee the rapidity of convergence and consistency of public information.

B. Performance Calculations

The main performance metrics concerned are search success ratio and the time taken to find the data. In the absence of detailed information on the underlying physical topology, average hops taken to find the data can be considered to reflect the search time. As we have mentioned, direct contribution represents the contribution of a single node to the total success ratio during a search. Node *i* will satisfy queries for $C_d(i)R$ times during *R* times search. Thus, the total success ratio will be

$$S_{UC}(TTL) = \frac{\sum_{i=1}^{N}(C_d(i)R)}{R} = \sum_{i=1}^{N} C_d(i) \qquad (9)$$

Success ratio has a close relation with *TTL* but is independent of *R*. Eq. 1 and 9 indicate that success ratio initially increases quickly as *TTL* increases, then as success ratio increases the increasing amplitude slow down gradually. Let S_{UCj} denotes the probability of search success at exactly the *j*th hop, then $S_{UC}(TTL)$ is given by

$$S_{UC}(TTL) = \sum_{j=1}^{TTL} S_{UC\,j}$$

Now consider the probability of searching success at exactly *j*th hop under the condition of search success within *TTL* hops, it should be $S_{UCj} / S_{UC}(TTL)$, so the average hops taken to find the data should be

$$T(TTL) = \sum_{j=1}^{TTL} j \frac{S_{UC\,j}}{S_{UC}(TTL)} = \frac{1}{S_{UC}(TTL)} \left(\sum_{j=1}^{TTL-1} jS_{UC\,j} + TTL\ S_{UCTTL} \right) \qquad (10)$$

Substitute $S_{UCTTL}=S_{UC}(TTL)-S_{UC}(TTL-1)$ to above equation, we get

$$T(TTL)S_{UC}(TTL) = T(TTL-1)S_{UC}(TTL-1) + TTLS_{UC}(TTL) - TTLS_{UC}(TTL-1)$$

$$T(TTL) = TTL - \frac{1}{S_{UC}(TTL)} \sum_{j=1}^{TTL-1} S_{UC}(j) \qquad (11)$$

By analyzing Eq. 11, we conclude that both *TTL* and $S_{UC}(TTL)$ have relationship with $T(TTL)$, meanwhile, $S_{UC}(TTL)$ is also relevant to *TTL*. Increasing *TTL* exclusively may not always lead to continuous increase of $T(TTL)$. It is advisable to decrease *TTL* properly under the condition of high success ratio guarantee.

IV. Performance Evaluations

A. Experimental configuration

The simulation starts with a network whose topology obeys power-law. We generate power-law topology with *10,000* nodes and the average degree of *3* according to [11] and assume that node capacity also obeys power-law and is the same as its initial degree. There are *2,000* kinds of data which are distributed uniformly among nodes. Each node holds *10* (m_i=*10*) kinds of data, and nodes are randomly chosen to issue queries with the same probability. The query frequency follows Zipf's law (α=*0.95*) distribution. Suppose that hot data are *15%* of total number of data, and queries for hot data are *75%* of total query frequency. *TTL* of flooding is set as 5.

B. Results

1) Efficiency of approach of achieving information Before adapting network topology, we firstly evaluate the efficiency of the approach discussed in sec.III. Each node collects information for computing the average value, and we use the expectation (E) and standard deviation (STD) of the average contributions to measure the efficiency. As Figure 3 shown, in a static network the expectations of both direct contribution and indirect contribution gradually tend to the real value, and the standard deviations decrease continuously to 0. Considering the dynamic situation, we randomly select *0.2%* of nodes to depart per *10,000* times search until *2,000* nodes depart. The results described in Figure 4 indicate that our approach could still achieve approximate real value with low standard deviations.

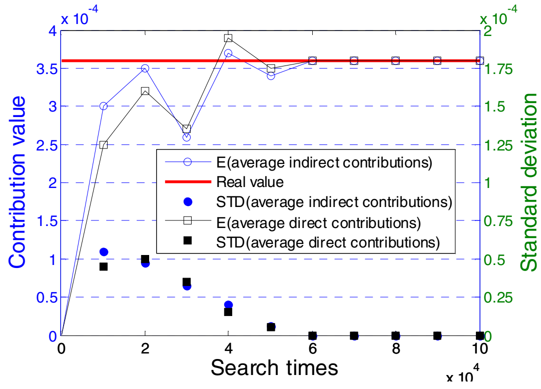


Figure 3. The efficiency of approach in a static network

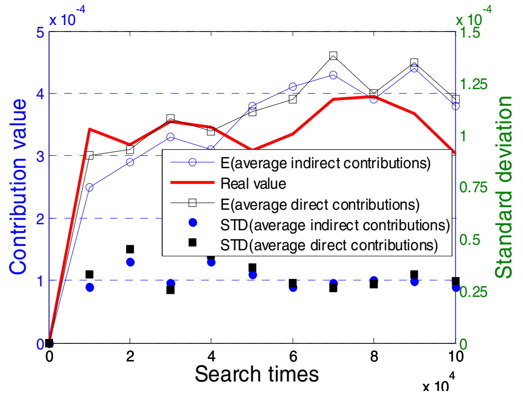


Figure 4. The efficiency of approach under node departure

2) Efficiency of the adaptation algorithms We evaluate the efficiency of adaptation algorithms including performance of success ratio and average hops. Figure 5 shows the node degree distribution after adaptation, which is similar to guntella. By contrast, we find that there is a convex part between *25* and *50*, and then drops immediately. Because overload handling mechanism makes the data be re-allocated among the nodes based on their capacities, especially, hot data are allocated more evenly to more nodes, which implies that the number of hot spots becomes larger and the possibility that the majority of data possessed by a node are hot data is smaller. This interprets why the optimized topology consists of less high degree nodes and more moderate degree nodes. Figure 6 reports the direct contribution for varying node degree, and we conclude that the higher the node degree is the more contribution it provide. Furthermore, we find there are some of moderate degree nodes do not always provide correspondingly high contributions. In fact, a CA can be divided into multiple sub-CAs which consist of one or several hot spots and many other guiders. Some of the aforesaid moderate degree nodes are guiders that connect these sub-CAs, those guiders provide high indirect contributions and are a small portion of convex part in Figure 5.

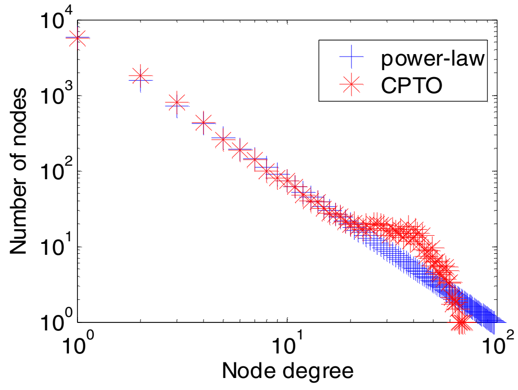


Figure 5. Node degree distribution

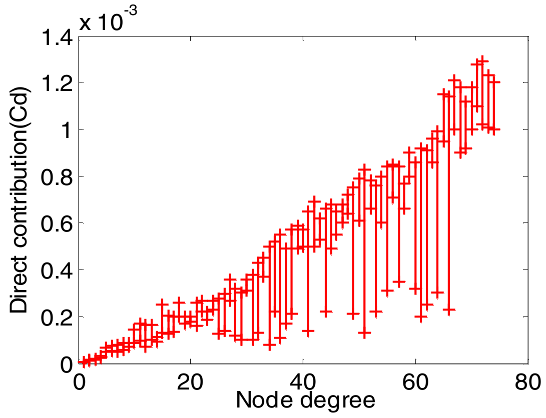


Figure 6. C_d vs. node degree

Figure 7 illustrates the evolutions of success ratio over *TTL* obtained under varying *r*. Success ratio initially increases quickly as *TTL* increases, then it tends to be steady gradually. Considering the influence of data density, we conduct the simulations for three cases (*r=2,000, r=3,000, r=5,000*). Since the total number of data is fixed (*10,000×10*), the more number of data types implies the lower data density. The success ratio of case of *r=2,000* tends to be steady earlier than that of the other cases, furthermore, it always keep lower average hops described by Figure 8. The twain figures also remind us that the gaps of three cases are not big and the gap between cases of *r=3,000* and *r=5,000* is smaller than that of the first two cases, which implies that our adaptation scheme works well on the adaptability of low-density data.

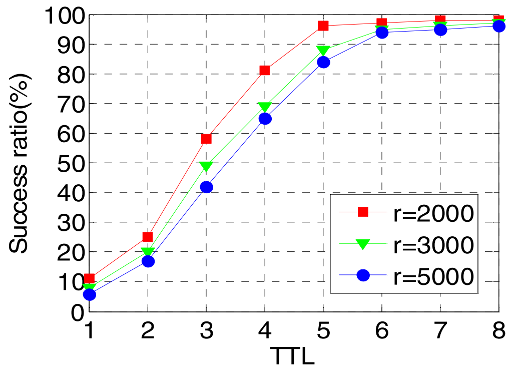


Figure 7. Success ratio for varying *r*

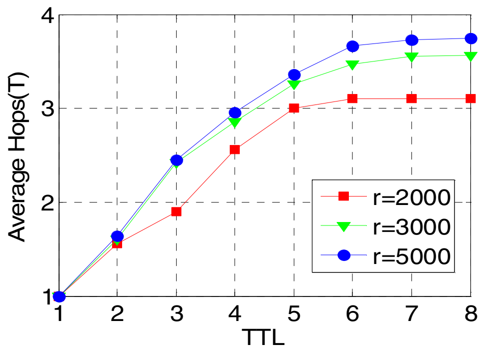


Figure 8. Average hops for varying *r*

3) Comparison of dynamic adaptability We execute search for *1,000,000* times and randomly select *0.2%* of nodes to depart per *10,000* times search until *2,000* nodes depart. The other settings remain the same as parameters in part A. As indicated in Figure 9, the success ratio is improved as the topology is modified, after *100,000* times search, it begins to descend slightly as the increase of leaving nodes, while decrease in power-law topology is more obvious. Search in power-law utilize high degree nodes to improve the performance, and as some of high degree nodes depart from the network, success ratio descends correspondingly. In our approach (CPTA), the search performance is mainly affected by guiders that connect sub-CAs and high degree providers, when some of these nodes depart, CPTA adjusts topology by selecting other competent nodes instead. Figure 10 shows the results under the variation of hot data. After the success ratio comes to a steady state, we continue to choose nodes randomly to issue queries for *250,000* times, and select 100 types of hot data randomly to swap with the same number of non-hot data per *50,000* times search. As hot data change to non-hot data, the quondam hot spots turn to ordinary nodes, their direct contributions to success ratio fall down correspondingly, so it accounts for the performance degradation. However, as the number of queries increases, CPTA begins to adjust the topology by constructing new CA based on the new hot data, and the success ratio returns to the steady state. As for search in power-law, hot data do not always concentrate on the high degree nodes, and queries will be forwarded by the high degree nodes to the new hot spots after a short time of routing update. So it can avoid the impact of variation of hot data in great extent. On the whole, CPTA keeps success ratio over *80%* at all time, and exhibits favorable adaptability to the fluctuation of data query frequency.

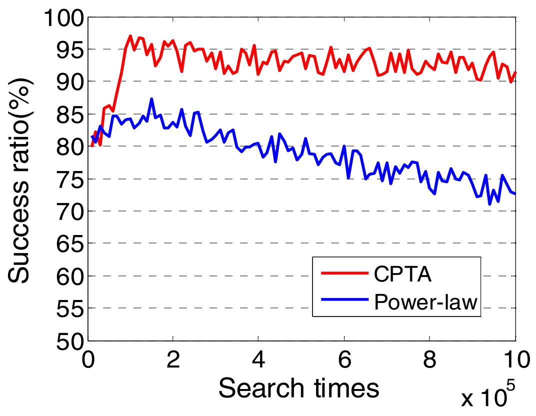


Figure 9. Success ratio under node departure

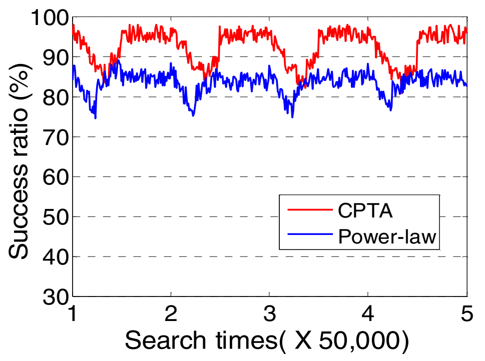


Figure 10. Success ratio under the variation of hot data

V. CONCLUSION

This paper investigates how to boost search performance from the perspective of topology adaptation in a decentralized manner. We propose CPTA, a topology adaptation scheme by constructing CA. In a CA, nodes enhance cooperation through connecting one another according to their roles and capacities. We design algorithms to construct CA, and present a probabilistic model to analyze the expected search performance. The simulation results indicate that the topology optimized by our scheme achieves high success ratio and low average hops, furthermore, it works well on the adaptability to dynamics.

ACKNOWLEDGMENT

This work is supported by National Natural Science Foundation of China under Grants No. 60773103, No. 61070161, No. 61070158, No.61003257 and No. 90912002, National Key Basic Research program of China under Grants No. 2010CB328104, China National Key Technology R&D Program under Grants No.2010BAI88B03, China Specialized Research Fund for the Doctoral Program of Higher Education under Grants No.200802860031, Jiangsu Provincial Natural Science Foundation of China under Grants No. BK2008030, China National S&T Major Project under Grants No. 2009ZX03004-004-04, State Key Laboratory of Information Security, Jiangsu Provincial Key Laboratory of Network and Information Security under Grants No. BM2003201, and Key Laboratory of Computer Network and Information Integration of Ministry of Education of China under Grants No. 93K-9.

REFERENCES

[1] I. Stoica, R. Morris, D. Karger, *et al.*, “Chord: A scalable Peer-to-Peer lookup service for internet applications”, Proc. of SIGCOMM’01, San Diego, 2001, pp. 149-160.

[2] B. Y. Zhao, J. L. Huang, J. Strbling, *et al.*, “Tapestry: A Resilient Global-scale Overlay for Service Deployment”, IEEE JSAC, 2004, 22(1), pp. 1-15.

[3] Gnutelliums LLC. Guntella protocol specification version 0.4. http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf.

[4] B. F. Cooper, “An Optimal Overlay Topology for Routing Peer-to-Peer Searches”, Proc. of Middleware’05, Berlin, 2005, pp.82-101.

[5] L. A. Adamic, R. M. Lukose, A. R. Puniyani, *et al.*, “Search in power-law networks”, Physical Reviews, 2001, 64(4), pp. 46135-46143.

[6] G. F. Feng, J. C. Zhang, Y. Q. Jiang, *et al.*, “Optimization of Overlay Topology for Unstructured Peer-to-Peer”, Journal of software, 2007, 18(11), pp. 2819-2829.

[7] Y. Chawathe, S. Ratnasamy, and L. Breslau, “Making Gnutella-like P2P Systems Scalable”, Proc. of SIGCOMM’03, Karlsruhe, 2003, pp. 407-418.

[8] Q. Lv, S. Ratnasamy, and S. Shenker, “Can heterogeneity make Gnutella scalable”, Proc. of IPTPS’02, Cambrdge, 2002, LNCS 2429, pp. 94-103.

[9] W. T. Li, “Zipf’s Law Everywhere”, Glottometrics 5, 2002, pp. 14-21.

[10] K. Sripanidkulchai, “The popularity of Gnutella queries and its implications on scalability”, http://www.cs.cmu.edu/~kunwadee/research/p2p/gnutella.html.

[11] C. R. Palmer, J. G. Steffan, “Generating network topologies that obey power laws”, GLOBECOM’00, San Francisco, 2000, pp. 434-438.

[12] G. Chen, P. C. Low, Z. H. Yang, “Enhancing Search Performance in Unstructured P2P Networks Based on Users' Common Interest”, IEEE TPDS, 2008, 19(6), pp. 821-836.

[13] D. J. Watts, S. H. Strogatz, "Collective dynamics of 'small-world' networks", Nature, 1998, 393, pp. 440-442.

[14] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors", Communication of the ACM, 1970, 13(7), pp. 422-426.