This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the ICC 2007 proceedings.

LiPS: Efficient P2P Search Scheme with Novel Link Prediction Techniques

Yaodong Zhang
Shanghai Jiao Tong University
Shanghai, China
zhydong@sjtu.edu.cn

Guobin Shen
Microsoft Research Asia
Beijing, China
jacky.shen@microsoft.com

Yong Yu
Shanghai Jiao Tong University
Shanghai, China
yyu@cs.sjtu.edu.cn

Abstract— **The usability of peer-to-peer (P2P) file sharing systems highly depends on their search (or query, content routing) efficiency. In this paper, we present LiPS: an efficient P2P search scheme with novel link prediction techniques. LiPS is a natural combination of recent technical thrusts from two different disciplines, namely 1) the exploitation of user interests in P2P search field, and 2) the link prediction in the complex networks field. Based on the experiential observation that people's social circle typically expands through friends' friends, we propose a novel neighbors' common neighbor link predictor (NCNP) and its two optimized variations. Trace-driven simulation results demonstrate the proposed link predictors and the effectiveness of LiPS. Specifically, the proposed refined and popularity-aware NCNP algorithm can double or even triple the prediction accuracy, as compared with normal common neighbor predictor. LiPS also significantly outperforms (by as large a margin as 15%) the original Shortcuts search method [1] and achieves up to 60% hit rate. Meanwhile, the query traffic is also slightly reduced.**

I. Introduction

The usability of peer-to-peer (P2P) file sharing systems highly depends on how effective and efficient one can find and retrieve data. Although it has attracted extensive studies in recent years, efficient P2P search remains a most challenging design problem. P2P networks appear in two different flavors, namely structured and unstructured. In general, unstructured networks is more preferable to the DHT-based structured ones because the intrinsic exact key matching mechanism of the latter, while providing other benefits, severely impairs its search efficiency [4]. For unstructured overlay networks such as Gnutella [2], researchers have mainly focused on improving the original naive flooding query mechanism using controlled flooding such as expanded ring search with random walks [3]. We refer readers to [4] for a comprehensive survey of P2P search technologies.

Exploiting various localities has been approved as a viable approach to improve the search efficiency in different application contexts. For example, the temporal locality is utilized to design efficient local file search algorithm [5]. Exploitation of the locality embedded in human interests has also received a lot of attention for its effectiveness in guiding search queries [6], [7], [1], [8], [9]. For instance, in [9], the authors propose to use partial index of shared data to reflect users' interests and to relate users with similar interests. In [8], the pattern and properties of file sharing between peers with common interests are also studied by using "data sharing graphs". An interest-based shortcuts algorithm (referred to Shortcuts hereafter) was designed and laid on top of the original flooding based query mechanism of Gnutella in [1].

It is observed in [1] that interest-based shortcuts result in a graph with *small world* properties, which is a typical characteristic of *social networks*, and this partially accounts for its resultant high search efficiency. Social networks stand for a specific class of network structures whose nodes represent people or other entities embedded in a social context, and whose edges represent interaction, collaboration, or influence between entities. Handy examples of social networks include the set of scientists in a certain discipline that co-author papers, the collection of colleagues that collaborate on projects, the collection of movie stars that collaborate on movies, etc. Social networks have two prominent properties, namely small graph diameter and high clustering coefficient, that are favorable to, among others, efficient search. Many proposals have been made to explicitly construct structures akin to social networks to provide high search efficiency and/or handle flash crowd more effectively [10], [11], [12].

In the discipline of complex networks, researchers have devoted a considerable amount of attention to the computational analysis of social networks, especially on the network evolution. In [13], the authors studied the link prediction problem in social network. Link prediction problem essentially focuses on efficient ways to exploit node proximity.

Motivated by the progresses in both threads (link prediction in complex networks and interests locality exploitation for P2P search) and also based on experiential observation that people's social circle typically expands through friends' friends (to be elaborated later on), in this paper, we propose novel link prediction algorithms to predict possible future links between peers to improve the search efficiency. We further design the search scheme, *LiPS*, which stands for Link Prediction and Shortcuts, by integrating the proposed link prediction algorithms with the interest-based Shortcuts [1]. Note that the proposed link prediction algorithms can be combined with other query methods as well. To the best of our knowledge, this is the first piece of work that explicitly studies the link prediction problem in P2P scenario and applies it to help P2P search.

Trace-driven simulation results demonstrate the effectiveness of LiPS. It can achieve up to 60% hit rate, significantly outperforms (by a margin as large as 15%) the original Shortcuts that has been proved to be most efficient P2P query method in [1]. Moreover, the cost in terms of query traffic is also slightly reduced. Note that LiPS improves the search efficiency but does not affect the correctness and the reliability of the underlying overlay since we can always resort to the basic query mechanism, such as controlled flooding, of the underlying overlay.

The rest of the paper is organized as follows: in Section II, we present some works that are related to and have motivated this work. We describe the rationale of link prediction, formulate the link prediction problem, design a new neighbors' common neighbor predictor and present the LiPS query procedure in Section III, followed by two link prediction optimization algorithms in Section IV. Extensive experiments using trace data from real P2P systems were performed and the results are presented in Section V. Finally, Section VI concludes the paper and talks about future works.

II. Related Works

In this section, we provide more detailed review of some related works to reveal their insights and rationale, which are also motivations of this work.

A. Utilizing locality for efficient search

Based on the observation that if a peer has a particular piece of content that one is interested in, it is very likely that it will have other items that one might be interested in as well, the interest-based shortcuts algorithm was designed and laid on top of the original flooding-based query mechanism of Gnutella in [1]. Simulation results demonstrated significant savings in search traffic. One interesting finding in the work is that Shortcuts results in a graph with high clustering coefficient, a typical characteristic of "social networks". This partially accounts for significant search efficiency improvement.

In [9], the authors propose assisted search by using partial index of shared data to reflect users interest and to relate users with similar interest. The partial index is maintained by a logically separate overlay on behalf of the peers. As in Shortcuts, assisted search also leverages the interest locality, but instead of gradually learning from history, peers express their interests explicitly and actively seek partners via the intermediate index overlay.

In [12], the authors used latent-variable clustering with Hierarchical Dirichlet Processes to model user preferences on music styles. The model is utilized to create social network akin overlay networks by connecting users with same music style preference. They demonstrated that overlay networks based on social characteristics indeed improve the performance of P2P networks. Clearly, they are also exploring the locality among user preferences. However, the user preference locality is used in a static way.

In this work, we also exploit the locality of user interest. However, unlike Shortcuts, we seek to actively and explicitly perform predictions for new possible queries. The rationale is based on another real world observation which is also different from that of Shortcuts.

B. Predicting links in social networks

Predicting the evolution of networks is hard and largely based on the network's topology and there is no universal models in the theory of evolving networks. In [13] the authors performed comprehensive study on the link prediction problem in social networks. They propose and evaluate several prediction models which can be classified into different categories according to whether they are based on information of paths, or based on information of node's neighborhood, or some high-level approaches such as clustering. According to experiments in [13], the successful rate of the best model does not exceed 16% in the co-authors' network.

Note that not all models fit for the P2P environment. For instance, some models require globe information and some other models take a long time to compute for a prediction, which are unacceptable for real P2P applications. More specifically, the prediction models for P2P networks should have good prediction accuracy, require only local or a very limited range of information and be easy to compute. In our study, we found that prediction models using only neighborhood information may satisfy such requirements. We propose several modified link prediction algorithms based on the common neighbor predictor to make them better fit the requirement of more efficient query over P2P networks. Furthermore, the proposed link prediction algorithms expedite the evolution for the P2P network towards the one with social network properties.[1]

III. LiPS: Link Prediction and Shortcuts

A. The idea of link prediction

Shortcuts [1] is based on the observation that if a peer has a particular piece of content that one is interested in, it is very likely that it will have other items that one is interested in as well. Therefore, it makes connections between peers having successful query-reply relations. In this work, link prediction is based on another observation from the real world that people make new friends through their existing friends. Only very occasionally, people will make new friends impromptu. Even in this situation, people tends to explore something in common, such as they both know someone or are from same town, etc., to consolidate the new friendship. Therefore, such kind of evolution is someway predictable.

In P2P applications, we can also find the similar kind of evolution. If we link two peers if they have common interests. Then, the common interest relation is highly transitive. That is, if peer A has a shortcut to peer B and peer B has a shortcut to peer C, we can predict, with high probability, that peer A may share some interests with C. This is exactly the common neighbor link prediction (CNP) concept [13] which predicts a

[1]Due to space limit, we are not able to present the results on this respect.

link between A and C from the fact that they have a common neighbor B.

B. Problem statement

Our goal is to predict a peer's future links based on his existing ones. Mathematically, the link prediction problem can be stated as follows: given a snapshot of a P2P network at time t_1, represented as a graph $G_{t_1}(V_1, E_1)$, where V_1 is the set of online peers at time t_1 and $E_1 \subset V_1 \times V_1$ is the set of existing shortcuts between these online peers. Link prediction is an algorithm with input $G_{t_1}(V_1, E_1)$ and output $E_{pre} \subset V_1 \times V_1$.

$$E_{pre} = LinkPred(G_{t_1}(V_1, E_1)) \tag{1}$$

where E_{pre} denotes the set of links predicted by the link prediction algorithm. Applying E_{pre} onto the original graph $G_{t_1}(V_1, E_1)$, we will get $G'_{t_1}(V_1, E_1 \cup E_{pre})$. Suppose that we take another snapshot of this P2P network at t_2, where $t_1 < t_2$, represented as $G_{t_2}(V_2, E_2)$. It is clear that we can use $G_{t_2}(V_2, E_2)$ as the ground truth to verify the prediction result and calculate the prediction successful rate of the prediction algorithm, which is defined as follows:

$$\alpha = \frac{|E_{pre} \cap E_2|}{|E_{pre}|} \tag{2}$$

Clearly, we need to find an appropriate prediction algorithm to maximize the prediction accuracy α.

C. Neighbors' common neighbor predictor

The simple common neighbor based predictor (i.e., CNP) leads to exponential growth of predicted links across generations of prediction and the results will inevitably become overly noisy. We therefore propose a more stringent link prediction method called neighbors' common neighbor predictor (NCNP), which is defined formally as follows:

$$NCNP^{\gamma}(p) = \bigcap_{p_i \in \Delta(p)} \Delta(p_i) \tag{3}$$

In the equation, γ is the iteration parameter, $NCNP^{\gamma}(p)$ represents the set of predicted links after γ-th iteration. $\Delta(p)$ denotes the set of neighbors of peer p prior to the γ-th iteration. Evidently, the larger γ is, the more selective the prediction will be. This fact makes the prediction converge and stabilize quickly.

We give a simple example in Figure 1 to illustrate our idea and show the difference between CNP and NCNP. Suppose peer A has shortcuts to peer B and C, and both peer B and C have a shortcut to peer D, respectively. When peer A observes that both peer B and C have a shortcut to D, it is more possible that peer A has some common interests with peer D. Therefore, we can predict the shortcut from peer A to peer D for peer A by computing one peer's neighbors' common neighbor. In a second iteration, if using NCNP, peer A will not add new neighbors since his neighbors (peer B, C, and D) do not have in common any other neighbor. On the contrary, if CNP is applied, then peer A will add peer E and F as new neighbors in a second round and peer G and H in a third round, and so on.

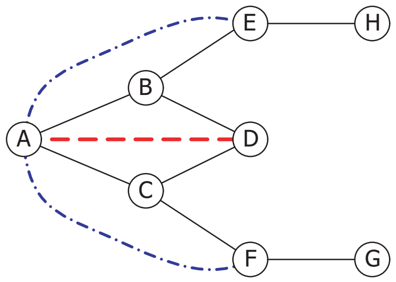

Fig. 1. Example of shortcuts prediction

Algorithm 1: The **LiPS** procedure

```
Input: $query$ (Query term)
Result: $hitList$ (Query result)
begin
    $hitList \longleftarrow searchExpressList(query)$
    if $hitList == NULL$ then
        $hitList \longleftarrow goUnderlyingOverlay(query)$
        if $hitList == NULL$ then
            $noRecordFound()$
            return
        $updateExpressList(hitList)$
        $predictionList \longleftarrow doPrediction()$
        $replaceExpressList(predictionList, n)$
    else
        $shuffleExpressList(hitList)$
end
```

D. Proposed LiPS query procedure

The proposed LiPS query procedure contains the following three simple steps, as listed in Algorithm 1:

1) Perform search over the express search list. If hit, shuffle the express search list by reordering according to the successful hit value, and then procedure terminates.
2) Perform search over the underlying overlay (using basic flooding-based search scheme). If not hit, prompt "no record found" message and terminate the procedure.
3) Update the express search list with the replying peers (i.e., Shortcuts idea). Then perform link prediction algorithm to generate a set of n predicted links and replace last n entries.

Clearly, the key idea here is to form the *express search list* with Shortcuts and the predicted links, and always first search the express search list with an expectation of early hit. Note that, in $doPrediction()$, we apply the proposed link prediction algorithms. In order to perform prediction, peers need to exchange express lists information with their neighbors. Such exchange of neighbor information is the actual cost we paid for using prediction. Fortunately, the cost is affordable and tolerable because 1) such information can be piggybacked in the keeping alive information that is regularly exchanged between neighboring peers; 2) based on the observation from real data set and empirical analysis, most peers do not frequently issue queries.

IV. Link Predictor Optimization

Equation 3 implies very stringent condition for a successful prediction. Peers may fail to generate any prediction result, i.e., the resulting intersection set is empty, because their neighbors do not have neighbors in common. This is not what we want. Therefore, we loosen the intersection rules by counting the occurrences of peers, instead of performing strict set intersection, of all neighbors' neighbor lists and select the first k most frequently appeared peers as prediction results. We call this algorithm Refined NCNP (RNCNP). More specifically, for each peer p, we generate the set of candidate peers by combining its neighbors' neighbor list.

$$C(p) = \bigcup_{p_i \in \Delta(p)} \Delta(p_i) \tag{4}$$

Let $F(a)$ represents the frequency that a peer a appears in $C(p)$, then

$$F(a) = \sum_{p_i \in C(p)} f(a, \Delta(p_i)) \tag{5}$$

where

$$f(a, \Delta(p_i)) = \begin{cases} 1 & \text{if } a \in \Delta(p_i) \\ 0 & \text{if } a \notin \Delta(p_i) \end{cases}$$

For each peer in the candidate set, we apply a ranking function, denoted as $s(a)$. We first simply let $s(a) = F(a)$. The prediction results are the peers whose score are greater than or equal to a given threshold TH. That is,

$$RCNP(p) = \{a : a \in C(p), s(a) \geq TH\} \tag{6}$$

The RNCNP algorithm is detailed in Algorithm 2.

To further explore the impact of different ranking functions, we propose another ranking function, taking into account the popularity of a given peer. It is widely accepted that the powerful and stable peers often play a critical role to the success of any P2P system. The popularity of a peer is measured by counting the number of times it uploads files to others. The resulting link prediction algorithm, called Popularity-aware NCNP (PANCNP), is the same as that of RNCNP, except the ranking function is changed to $s(a) = F(a) \cdot Pop(a)$, where $Pop(a)$ is the popularity of a peer a. Note that the threshold here is typically magnitude of order greater than that in the RNCNP algorithm.

Algorithm 2: Refined/Popularity-aware NCNP

```
Data: Graph $G(V, E)$
Result: $P_{Set}$ (set of prediction links)
begin
    $wList \longleftarrow NULL$
    while $i < \gamma$ do
        for $n \in V$ do
            for $p \in \Delta(n)$ do
                for $p_i \in \Delta(p)$ do
                    if $s(p_i) \geq Threshold$ then
                        $wList \longleftarrow wList \cup \{p_i\}$
            $Tmp \longleftarrow First(wList)$
            for $p_i \in Tmp$ do
                $addLink(n, p_i)$
            $P_{Set} \longleftarrow P_{Set} \cup Tmp$
end
```

V. Experimental Results

We performs extensive experiments to verify our prediction ideas. The whole evaluation is divided into two parts: 1) we first verify the effectiveness of the proposed link prediction algorithms, and 2) we emulate a real case of Gnutella [14] through trace-driven simulation.

A. *Effectiveness of link prediction algorithms*

We use the data set *Can-o-Sleep* collected from the Open-Nap system which contains the data over an 81 days period between February 28 and May 21, 2003. The data set records all the transactions for every participants and is organized in a relational database structure with four distinct object types and seven link types, which suffices our experiment requirement.

We divided the Can-o-Sleep data set into two subsets G_1 and G_2 based on their time stamps. G_1 is created with data collected from day 1 to day 39 and G_2 with the data from day 40 to day 79. Let G_w and G_t be the working set and test set, respectively. G_w consists of the nodes in $G_1 \cap G_2$ and the links from G_1 by removing the ones which either the start node or the end node is not in $G_1 \cap G_2$, and G_t contains the nodes from $G_1 \cap G_2$ and the links from G_2 after the same removing process as for G_w. The resulting statistics of the data set is shown in Table I.

TABLE I
Statistics of the Can-o-sleep data set

Graph	Nodes	Links
G_1	1595	21318
G_2	5902	79306
$G_1 \cap G_2$	1357	5958
G_w	1357	19108
G_t	1357	33889

We run the three link prediction algorithms (NCNP, RNCNP, and PANCNP) on G_w. Each algorithm generates a predicted link set E_{pre} based on the input graph G_w. Then, for every predicted link in E_{pre}, we verify its appearance in graph G_t. As stated before, the prediction accuracy is defined as the ratio of the number of predicted links that appear in G_t to the total number of predicted links. In RNCNP and PANCNP, we set the threshold to be one and zero, respectively. The results are shown in Table II.

From Table II, we see that the successful rates of proposed link predictors are already much higher than the results in [13] of which the highest one is about 16% only. This confirms that the simple interest relation in P2P network is easier to be predicted than the complex relation between individuals. Among

TABLE II

PREDICTION SUCCESSFUL RATE

Algorithms	Successful Rate	Correct / Total
NCNP	19.5%	38 / 195
RNCNP	23.0%	210 / 916
PANCNP	34.6%	311 / 916

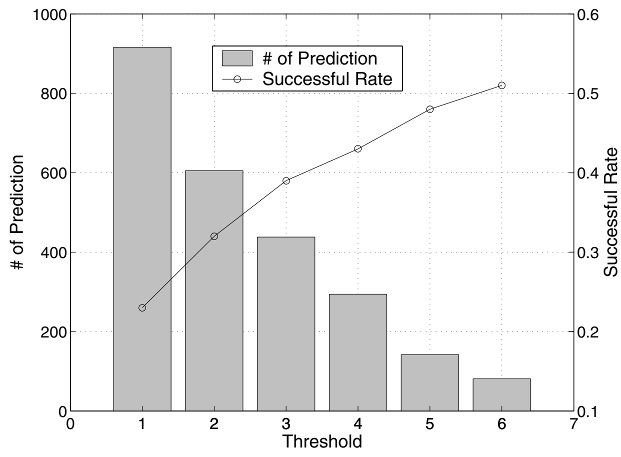

Fig. 2. Number of resulting predictions of RNCNP and the prediction successful rate versus threshold.

the three proposed link prediction algorithms, NCNP leads to the lowest successful rate, mainly due to the strict common neighbor condition. As we will see below, the successful rate can be further improved if we set proper thresholds. In fact, the proposed link predictors can double or even triple the performance of normal common neighbor predictor (see Figure 2 and 3).

To find out the proper threshold, we study the resulting number of predictions and the prediction accuracy against different thresholds for RNCNP and PANCNP algorithms, as shown in Figure 2 and 3.

The two figures reveal that most of wrong predictions indeed occurred when the threshold is too loose. Setting a larger threshold, more prediction noise can be filtered out. As a result, the successful rate improves while the total number of predictions decreased. Clearly, there exist a tradeoff between the prediction accuracy and the number of predictions. We are still working on effective means to obtain the optimal tradeoff.

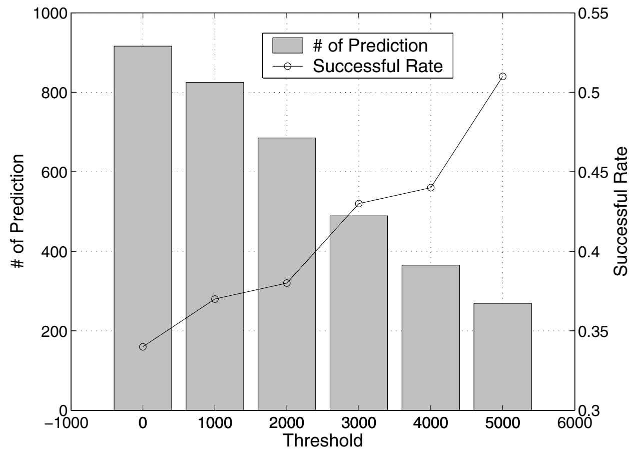

Fig. 3. Number of resulting predictions of PANCNP and the prediction successful versus threshold.

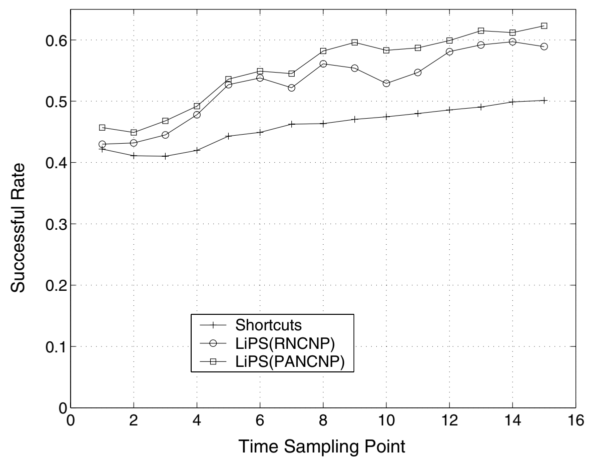

Fig. 4. Comparison of successful hit rate between Shortcuts and LiPS versus overall time passed.

B. Search Efficiency and Comparison with Shortcuts

We use Gnutella as the underlying overlay structure to perform the search efficiency test. The data is collected from a real Gnutella's network over a extended time period, consisting of 4,447 distinct users, 845,454 shared files and 11,075 queries issued by users. Each query record contains a time stamp denoting when the query is made by a user. We sort the query events by their time stamp, start the simulation at the time the first query is issued and end the simulation when the last query finishes. In the experiments, the query routing table size is set to 15 for Shortcuts. Accordingly, we set the replacement length to be 5 (i.e., $n = 5$ in the query procedure in Algorithm 1) and the size of express search list to be 15 for LiPS for fair comparisons.

The experimental results are shown in Figure 4. The horizontal axis represents the time sampling point. The interval between two subsequent ticks is approximately 0.87 days and the whole data set lasts for 13 days. The vertical axis represents the average successful hit rate of Shortcuts and LiPS, which is the ratio of the number of hit on link in the list and the total number of link visited, i.e., excluding the hits that are obtained through the underlying flooding-bases query mechanism of Gnutella. From the results, LiPS obviously outperforms the Shortcuts in terms of successful rate by as large a margin up to 15%. We did not perform the NCNP predictor since based on the results got from the first part of the experiment, NCNP predicts too few links which consequently brings little gains of the prediction successful rate.

Detailed analysis reveals that LiPS successfully predicts and preserves many powerful peers which Shortcuts cannot keep

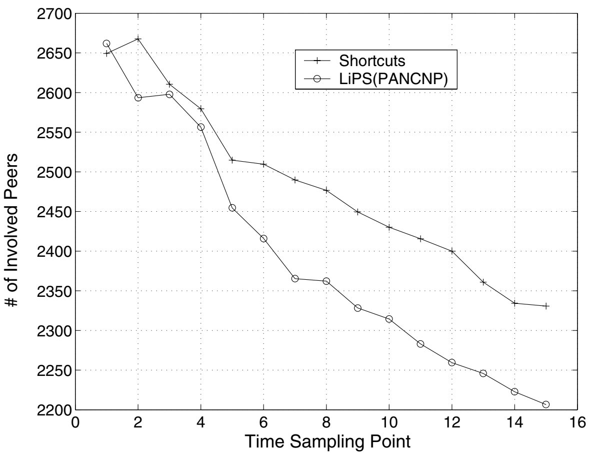

Fig. 5. Work Load

because its shortcut replacement policy is very sensitive to the peer's sudden change of interest. When the peer occasionally changes his interest for a short time period (i.e., transient interest), those already built shortcuts to the powerful and popular peers that have similar interest to his primary interest will soon be replaced by some other peers fulfilling the transient query. On the contrary, LiPS reacts to the sudden change in interest more smoothly because it is unlikely that all neighboring peers change their tastes synchronously. Therefore, when user changes back to his primary interest, some of previous powerful peers still exist and are certainly helpful. This also accounts for the result that the performance of LiPS is close to that of Shortcuts at beginning because the knowledge of neighbors have not been accumulated yet.

For quantify the query traffic cost, we measure the number of the involved peers for each query event. The involved peers are defined as the set of peers visited during a query session. Note that for LiPS, the number of involved peers includes the number of neighbors of the predicting peer in order to take into account of traffic overhead for prediction. The results are shown in Figure 5. In the figure, the PANCNP link prediction algorithm is adopted in LiPS. At the beginning, there is no big difference between Shortcuts and LiPS. However, LiPS gradually shows more savings in the query traffic because of the built-up of express search list and the improvement on successful rate.

VI. Conclusions and Future Work

Motivated by the progresses in both threads (link prediction in complex networks and interest locality exploitation for P2P search) and also based on experiential observation that people's social circle typically expands through friends' friends (to be elaborated later on), in this paper, we present *LiPS*, an efficient P2P search scheme, by integrating the several proposed novel neighbors' common neighbor link predictors with the interest-based Shortcuts scheme. To our best of knowledge, it is the first time that link prediction techniques are applied into P2P applications.

The effectiveness of LiPS was demonstrated through trace-driven simulation where the Can-o-Sleep data set is used to verify the efficiency of proposed link predictors and the Gnutella data set for its effectiveness when applying to P2P search. Specifically, both the refined NCNP and popularity-aware NCNP can double or even triple the performance of normal common neighbor predictor. The hit rate of LiPS can achieve up to 60% and exceeds the original Shortcuts by as large a margin as 15%. In the mean time, the query traffic is slightly reduced.

We are working on effective means to obtain the optimal tradeoff between number of predictions versus the prediction accuracy. We are also implementing a real P2P application using the mechanisms proposed in this paper. In a longer term, we plan to find out the root causes for link predictability in P2P network and perform some theoretical study.

VII. Acknowledgment

The PROXIMITY Can-o-Sleep database is based on data collected by the Privacy, Internet working, Security, and Mobile Systems Laboratory at the University of Massachusetts Amherst with additional preparation by the Knowledge Discovery Laboratory, University of Massachusetts Amherst.

References

[1] K. Sripanidkulchai, B. M. Maggs, and H. Zhang, "Efficient content location using interest-based locality in peer-to-peer systems," in *IEEE Proceedings of the INFOCOM 2003*, pp. 2166–2176, 2003.
[2] Gnutella 0.6, "http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html."
[3] B. Yang and H. Garcia-Molina, "Improving search in peer-to-peer networks," in *IEEE Proceedings of the ICDCS 2002*, pp. 5–14, 2002, .
[4] J. Risson and T. Moors, "Survey of research towards robust peer-to-peer networks: Search methods," *Computer Networks*, 50(17):3485–3521, 2006.
[5] C. A. N. Soules and G. R. Ganger, "Connections: using context to enhance file search," in *Proceedings of the SOSP 2005*, pp. 119–132, 2005.
[6] A. Crespo and H. Garcia-Molina, "Routing indices for peer-to-peer systems," in *IEEE Proceedings of ICDCS 2002*, pp. 23–32, 2002.
[7] E. Cohen, A. Fiat, and H. Kaplan, "Associative search in peer to peer networks: Harnessing latent semantics," in *IEEE Proceedings of the INFOCOM 2003*, pp. 1261–1271, 2003.
[8] A. Iamnitchi, M. Ripeanu, and I. T. Foster, "Small-world file-sharing communities," in *IEEE Proceedings of the INFOCOM 2004*, pp. 952–963, 2004.
[9] R. Zhang and Y. C. Hu, "Assisted peer-to-peer search with partial indexing," in *IEEE Proceedings of the INFOCOM 2005*, pp. 1514–1525, 2005.
[10] G. S. Manku, M. Bawa, and P. Raghavan, "Symphony: Distributed hashing in a small world," in *USENIX Symposium on Internet Technologies and Systems*, 2003.
[11] K. Y. K. Hui, J. C. S. Lui, and D. K. Y. Yau, "Small world overlay P2P networks," in *IEEE International Workshop on Quality of Service (IWQoS)*, pp. 201–210, 2004.
[12] A. Fast, D. Jensen, and B. N. Levine, "Creating social networks to improve peer-to-peer networking," in *Proceedings of ACM SIGKDD 2005*, pp. 568–573, 2005.
[13] D. Liben-Nowell and J. M. Kleinberg, "The link prediction problem for social networks," in *Proceedings of CIKM 2003*, pp. 556–559, 2003.
[14] S.-T. Goh, P. Kalnis, S. Bakiras, and K.-L. Tan, "Real datasets for file-sharing peer-to-peer systems," in *Proceedings of Intl. Conf. on Database Systems for Advanced Applications (DASFAA)*, pp. 201–213, 2005.