Searching for Answers via Social Networks

Jyun-Jie Huang*, Shao-Chen Chang†,Shun-Yun Hu ‡
National Central University, Taiwan, R.O.C.
*†{hugo,cavour}@acnlab.csie.ncu.edu.tw ‡syhu@yahoo.com

***Abstract*—Seeking answers to questions is a natural part of our learning and social interactions. Although search engines, web-based forums, and inquiries to friends via e-mails or instant messengers are all methods we use today, in many cases, much time is still spent to search, organize, or wait for responses. If knowledgable people can be found online to answer questions in real-time, then the time spent to browse webpages, wait for forum responses, or ask multiple people may be much reduced.**

In this position paper, we propose *Connet*, a peer-to-peer (P2P)-based people search system that helps people to have questions answered in real-time by knowledgable contacts via people's collective social networks. Users share successful experiences of finding responders and achieve greater efficiency when seeking answers to everyday questions. Connet relies on similarity measures between historic and new questions asked, and the trust among mutual friends, to provide more timely question-answering mechanisms. Meanwhile, it also enables a new type of online interactions with the friends of friends.

I. Introduction

Human beings are natural inquirers. In daily life, we often face situations or questions that may require answers or advices. Before the computer era, we rely on answers from acquaintances or consultations from books. In today's Internet society, we could search for webpages via search engines, pose questions on forums, or ask distant friends via e-mails or *instant messengers* (IMs). However, while short and factual answers can be found via a quick Google search, other times a question involves multiple parts that cannot be easily resolved in a simple search. Likewise, although in-depth answers are obtainable on forums, responses usually take time. Suitable forums for niche inquiries may also be hard to find. If a knowledgeable person is available to answer, much of the time for searching or waiting can be saved. Real-time interactions with known contacts thus would be ideal, but when no contacts from a person's existing social network can be found, or if the contact is unavailable at the moment, then the inquirer is forced to find answers alone or wait for later resolutions.

One of the the largest digital social networks today is the instant messaging networks, whose users range in 100 millions with millions of concurrent users [1]. As such, IM networks constitute some of the largest human resources tappable online. However, each IM user's contact list is often restricted to their own acquaintances. So even though people may ask their direct contacts questions, the spread of the question is quite restricted. If the contacts' social networks can be combined, a more powerful network may emerge. That is, users can find resources from the friends of a friend rather than just using one level of the social network. A recursive search that forwards and spreads a question beyond one's own contacts may thus tap on the collective experiences and knowledge of a larger pool of people. Intrinsic trust is also built within the combined social networks, making inquiries to the friends of friends somewhat more reliable and trustworthy than asking mere strangers.

In this paper, we propose a *peer-to-peer* (P2P)-based people search system that allows one to find relevant on-line persons who can answer specified inquiries easily and immediately through people's collective social networks. Unlike *personalized search* proposals that try to build profiles for specified people on the Web (e.g. *Spock* [2] tags people with different labels to build up and search for profiles on individuals), we focus on the goal of finding on-line people who can actually answer questions in real-time. Although some P2P systems have also utilized social networks to support enhanced functions (e.g. *Tribler* [3] uses the relations among friends to enhance file-sharing, while *Maay*[4] offers distributed personalized search that allows users to share, search, and retrieve documents from one another), we take on the direction to find actual *persons* who might be willing to have live interactions. One of our design considerations is that we do not seek to find the best responders, as it would require global knowledge and extensive search, but simply someone who is *good enough* to answer a question. We thus would like a system that can quickly help one to find appropriate people to ask any question specified in natural language. Inquirers and the responders may even become new acquaintances over time.

II. Problem Definition

Our goal is to allow any given natural language question be answered by some knowledgable online persons through real-time interactions. The backdrop is the fact that IM softwares have proliferated to the range of millions of concurrent users in recent years [1]. IMs allow any user to keep an online presence among friends or associates. Any two persons in the system can add each other to a *contact list*, and initiate text or voice conversation subsequently at any time. An IM system thus candidly captures the dynamics of online social networks through users' contact lists. If we assume that for a given *inquirer* (i.e. a user who has a question), there exist some knowledgable *responders* on the IM network who can answer the question, and that any user can initiate an interaction request to any other user currently online, our problem can then be defined as: how to match up the inquirers and the responders *accurately* (i.e. the responder is indeed qualified to answer the question satisfactorily) and *efficiently* (i.e. little time or resources is required to find the right responders).

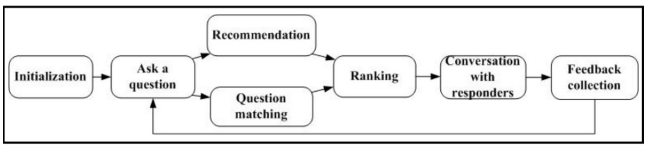

Fig. 1. Phases in Connet

III. Challenges

We describe some of the challenges involved as follows:

A. Search Accuracy and Efficiency

We first need a mechanism to match up potential responders with inquirers, which involves two basic issues: building knowledge profiles of the responders, and matching a question posed against the profiles. Other issues include the format of the knowledge profiles, and how to perform the matching efficiently and scalably, given that IM networks have millions of users. We might also want to give feedbacks on match quality to the system so to improve search accuracy. However, measuring search effectiveness can be an issue, as the success of an inquiry is very subjective and difficult to quantify. The satisfaction of users can also differ individually, as perceptions about a given interaction usually vary between persons.

B. Incentive and Availability

For P2P file-sharing networks, users only need to contribute storage and bandwidth, and the network can function without the users' attention. However, for real-time answering, the responder needs to be online and spend time to answer. Whether enough incentives exist will thus be crucial. But even if a potential responder is able and willing to answer questions, he or she may also not be online at the moment of need.

C. Question Bombardment

In the real world, popular or capable people are always overloaded with tasks and requests. Similarly, popular or able responders could have a higher probability to encounter bombardments of questions from inquirers. However, responders can lose their patience or even participation, if they receive requests beyond what they are willing to accommodate.

D. Malicious Users

Malicious users could violate system rules to make the system unfair or non-functional. This can be a serious problem that affects the willingness to participate in the system. Potential malicious acts include: 1) Collusion: where consented people collaborate to take advantage of unknowing third parties, and possibly boost up certain system metrics in favor of them. For example, malicious users or advertisers might collude to improve the search results on them in order to become highly ranked responders. 2) Pranking: some users might trick the inquirers just for fun, which could waste the inquirers' time and reduce their trust of the system. Avoiding pranking thus is another important issue that should be considered.

IV. The Design of Connet

Our proposal to find online responders is a P2P-based people search system called *Connet*. The key idea is that while a person may have only limited contacts, a search through the contacts of contacts (i.e. spreading inquiries over the collective social network) may prove to be more fruitful at finding potential responders. The search criteria are based on questions asked or answered in the past. If previous questions do not exist, then keywords of personal interests in a contact's profile is used instead. When a question is posed to Connet, it goes through a chain of contacts to collect a list of currently available people who had successfully answered similar questions. The inquirer then selects a potential responder from a ranked list to initiate real-time text or voice conversations. If the inquirer is satisfied with the result, both the question and the conversing parties are recorded to improve future searches. The recursive search through contacts addresses the challenge of collecting sufficient data to build knowledge profiles on responders. As we assume that human judgments may be more accurate than machine analysis, and that people are more willing to help acquaintances than strangers. To enhance the search with human judgments, a person may also optionally recommend other suitable responders during a search. To avoid overloading popular responders, a responder can be absent from other searches when answering a question.

A. System Architecture

Connet is mainly a P2P system, but a server is still used for account management, client bootstrapping, and user profile tracking. The P2P part is where each user directly connects with his or her contacts to perform actual searches. When new users join Connet, they must first register for an account to obtain unique IDs. The search for responders is conducted by local processing on each client node along the inquiry paths, as inquiries spread through the social networks of each contact.

Figure 1 shows a general work flow in Connet, where each user first builds an initial profile by providing some keywords that the user is willing to answer. During a search for responders, a natural language inquiry is first sent to the inquirer's *contact list*, which may contain manually input friends, or contacts provided by the system. The question will be forwarded recursively up to a specified depth (i.e. the levels of contacts, which is set to 3 by default, but can be adjusted). When a contact receives a question, a *question matching* procedure calculates the similarity between the inquirer's question and past questions that this contact had asked successfully. If the contacts are willing to take a look at the currently asked questions, they may also optionally *recommend* people from their social networks for suitable responders. After a recursive collection of potential responders, a *ranked* recommendation list is produced for the inquirer to select and initiate real-time conversations. Once the conversation is over, if the inquirer is satisfied, the system is notified to record the question and the responder at the inquirer's node to help future inquiries. We now describe the five main phases of Connet in details:

B. Execution Phases

Initialization For a new Connet user, a personal profile with basic information such as age, sex, education, interests, etc., is first created. The user then manually adds known friends as contacts. In case there are no known contacts, or if the number of contact has not met a system-specific size (e.g. 30 people), the system completes the initial *contact list* by using a spider to perform a random walk from an initial user with similar interests as the joiner. The spider will crawl until the *contact list* reaches the system-specific size. A *contact list* thus provides a starting point for future searches, and is composed of people from both automatic crawling and manual additions of existing friends. The initial *contact list* may be complete strangers with similar interests, but as the number of known friends increases, the *contact list* will likely be composed of known friends only.

We use *multidimensional scaling* (MDS) [5] to judge the similarity in user interests. It is a set of related statistical techniques often used in data visualization to explore similarities or dissimilarities in data. The formula is

$$Phi(d, \delta) = \sum_{i=1}^{n} (d_i - \delta_i)^2$$

where n represents the number of interests in one's profile, d_i stands for the level of a particular interest i in user d's profile, and δ_i is level of the same interest of another user δ. The level of interests of each user can be determined via some simple questionnaires. Basically, d_i and δ_i can be regarded as coordinate points within a particular interest dimension, and the difference indicates their closeness. If the distance between two people is less than a threshold, then they are considered to have similar interests.

Question Matching A new question posed by the inquirer is likely not the same as previously answered questions. So when a contact receives a question, a *similarity measure* between the question and each previous questions asked by the contact is calculated. If the similarity exceeds a given threshold, the system would add the contact (i.e. the previous inquirer), the responder, the *similarity measure*, and optionally, the question itself to a *recommendation list*. The reason for recording the previous inquirer is to treat them also as potential responders, so to relieve popular responders from being overloaded. Existing methods that compare the similarity between two text strings (i.e. the current and previous questions) is adopted here, e.g. the *Jaccard* index [6], which is a statistic used to compare the similarity among diverse sample sets. However, as *semantic similarity* is preferred over text similarity in our case, other similarity calculations between two words in a hierarchical semantic knowledge base [7] can also be used. Semantic knowledge bases such as *Wordnet* [8] are readily available due to the success of computational linguistic projects, as well as sentence similarity research such as [9]. With improvements on semantic similarity research, the comparison can be expected to improve with time.

Recommendation Question matching is an automatic procedure to find potential responders. However, while automatic selection saves human effort, we consider that human judgement is still more accurate when it can be applied. We thus would like to recruit and integrate human judgement to the formation of the *recommendation list*. So besides question matching, people can also manually recommend a friend or himself/herself as potential responders. This might increase the probability to solve a question, as humans usually have better contextual knowledge of a question, and even personal knowledge about the inquirer's background. Additionally, when the questions come from one's own friends or their acquaintances, people likely would be more willing to help.

TABLE I
EXAMPLE OF A RECOMMENDATION LIST

Candidate responders	Inquirer	Introducer	Similarity measure
Snow White	Brin	-	63
Yao	Wang	-	82
Cinderella	Hilary	-	78
Page	Gates	Shrek	-
Clark	Bush	-	80

A contact who recommends a qualified friend is called an *introducer*. However, to prevent abuses such as spamming the recommendation list with unqualified persons, every introducer has an *appraisal value* which gets increased only with successful recommendations. The *appraisal value* for each user is stored at the main server, and higher *appraisal value* indicates that the recommendation may be better trusted.

Ranking When an inquirer receives the *recommendation list* as show in Table I, a preferential sorting occurs to facilitate the inquirer's decision. The *recommendation list* contains entries made up of 1) the candidate responder, 2) previous inquirer, 3) introducer, and 4) *similarity measure* between the inquiry and the responder's previously answered or asked question.

The recommendation list's ranking is then based on:

1) *relevance* between the current and previous questions:

$$relevance = \sum_{i=1}^{q} sim(Q_i),$$

where $sim(Q_i)$ is the *similarity measure* between the inquiry and each of the q questions a given responder has previously answered or asked. For entries that come from a previous inquirer (i.e. an asked question), *relevance* will be multiplied with a *damping factor* α, where $0 < \alpha < 1$. This is due to our design that previous inquirers also can be responders for the same question, as they might have gained knowledge through previous Q&A sessions, albeit at reduced qualifications.

2) *confidence* for a given responder based on the reputations (i.e. *appraisal values*) of the responder's introducers.

$$confidence = \sum_{i=1}^{n} app(Introducer_i),$$

where n indicates the number of introducers who recommend a particular responder and $app(Introducer_i)$ is the *appraisal value* of the introducer i retrieved from the server. The summation therefore indicates how trustworthy a recommendation may be.

Ranking therefore can be shown either by *relevance* or *confidence*, and the inquirer can see a ranked recommendation of contacts for determining which person to initiate a conversation. *Relevance* is the automatic searching along the social networks, whereas *confidence* is the manual recommendation, where the introducers have to individually recommend a potential responder. But note that the value of the recommendation rests on automatically recorded appraisal values from previous recommendations.

Feedback Collection Once a conversation between the inquirer and responder is finished, the inquirer may give an evaluation on the conversation. To simplify the evaluation, we use a simple system of providing only either a positive or negative remark. If the remark is positive, the *appraisal value* of the responder's recommender will be increased at the centralized server. The responder and the question asked will also be recorded at the inquirer to facilitate future searches.

C. Enhancements

In addition to the basic mechanism above, we describe four more enhancements for Connet useful in practical scenarios.

Message board To allow suitable but offline responders still the chance to answer, questions asked but have yet been responded can be shown at each passed-through contact's *message board*. So even if responders were not available at the time a question was posed, they can still check the message boards later and contact the inquirer. Doing so can provide a somewhat asynchronous Q&A session that increases the chance of resolving questions.

Anonymous inquiry Connet searches answers mainly via social networks, where questions are probably forwarded to known friends. But there are questions that an inquirer may not want certain friends to know. With such privacy concern, we can also allow users to ask questions anonymously. In this mode, the inquirer's name will simply not be added with the question during forwarding. As each contact who receives the question knows only the question and where to return search results, users on the forwarding path can hardly know for sure who is the inquirer. Although, anonymous inquiry could reduce the willingness of potential responders to answer.

Answer points To provide incentives for the responders, a form of point reward system can be added to Connet. One example would be having *answer points*, which are artificial currency maintained at the central server. Users receive basic points periodically from the server, and can use their answer points to attract better responders. When posing a question, a user may trade certain answer points for a successful answer. This way, people are encouraged to answer by collecting answer points to either attract responders or to show prestige.

Block list As the basic design of Connet is to collect responders in a flood-like fashion, malicious users may exist at various levels in the social networks. To prevent malicious users from disturbing the system, users can put suspected malicious users onto a personal *block list*, such that the blocked users would be filtered out from any subsequent recommendation or search.

V. Discussions

Connet is designed with both centralized and P2P components. It is P2P mainly for two reasons: 1) the questions and contacts collected are private data which users may only want to share with close friends or associates. By not centralizing these data, the system would not need to rely on a single trusted entity. 2) P2P systems allow computing and storage resources of the clients be pooled collectively to support demanding tasks, so highly scalable and affordable systems can be created. However, certain tasks such as account management, or the tracking of performance metrics such as *appraisal value*, have security or fairness concerns, and thus are better handled by a centralized server. However, if all the work is centralized at the server, when tens of thousands of inquirers are asking questions at the same time, the server likely will require tremendous amount of resources for similarity computations. We discuss Connet's design with regard to the challenges in people search:

A. Search Accuracy and Efficiency

To perform useful search, knowledge profiles on potential responders must first be available. If such profiles were to be collected and organized centrally, it would not only require tremendous resources, but could also open potential privacy violations. The knowledge profiles used in Connet are the ever increasing questions people have previously asked, and are maintained distributively by each inquirers. Privacy thus is protected, while resources grow scalably with the system size. If previous inquirers are willing to publicize their questions, they are welcome to do so to help new inquirers perform more informed selections. But even if not, the system still works by reporting the *relevance* back to the inquirers. Basically, Connet provides an easy method to collect and construct the knowledge profiles regarding each responder, and the profiles grow continuously through successful conversations.

Search accuracy is accomplished by the matching against similar questions answered previously (i.e. via *relevance*) as well as people's collective recommendations (i.e. via *confidence*). Higher relevance indicates that the responders may have better knowledge, communication skills, or willingness to help the inquirers. It is thus a meaningful measure for what people can expect from the responders. Although the particular method to calculate similarity between two questions can impact search accuracy, it is a problem beyond our scope and can be expected to improve over time. Additionally, ranking by confidence provides people with a more human-oriented recommendation to connect previously unrelated or unknown people. Both mechanisms together thus will provide relevant information to find responders. In terms of the search efficiency, although the recursive search through contacts may cost extra bandwidth and time, note that the load is distributed among the peers, and the search is also performed concurrently at various layers of contacts. A question thus may still spread more widely and effectively than could be otherwise achieved.

B. Incentive and Availability

We believe that interactions between people and the chance to share personal knowledge or experience with others is the best incentive for a system like Connet. Although there is no hard evidence, the success of Q&A sites such as *Yahoo! Answer* [10] could provide some hints. Another source of incentives is that as the inquirers are the friends of friends, people may be more willing to answer them than to mere strangers. The use of answer points may provide additional incentives. By rewarding successful responders with answer points, people are encouraged to answer more, as answer points can both symbolize prestige, and be useful when asking questions (i.e. inquirers can use answer points to attract better responders). The issue of availability may be addressed by a simple message board function at each user, where if a question goes unanswered, there is still chance that potential responders would see it later.

C. Question Bombardment

One potential concern in Connet is that knowledgable or qualified responders may get overwhelming requests. However, the nature of the P2P search in fact distributes the load to different responders. As the search always starts and proceeds in localized areas, the list of responders returned will differ among inquirers even for the same question. One key difference between Connet and centralized search such as Google is that the goal is not to find the best results in the whole system, but simply ones with *good enough* chances. Inquirers are also free to choose between ranking by relevance or by confidence, which could point to different responders. At a final measure, responders can prevent excessive requests by setting their status as *busy* when engaging in a conversion, thus be excluded from other searches. Another design in Connet is that inquirers also become responders automatically after getting answers successfully. This is due to the assumption that an inquirer can also gain knowledge and be able to answer questions after successful Q&A sessions. Of course, the qualifications of these new responders need to be discounted somewhat. But over time, they will become significant as a source of potential responders to answer similar questions.

D. Malicious Users

Connet is based on the trusts people place towards their friends and contacts, and the extension of that trust into wider layers of friends. If someone abuses this trust and provides false or inaccurate information or recommendation, it could easily affect the quality of the search. For example, as people only connect directly with their first level contacts, any information passed beyond the first level can in theory be modified maliciously. However, if such situations occurs (e.g. a person finds that a responder is in fact an advertiser, or that someone has made unqualified recommendations), users can block the offender with the blocklist. Afterwards, the search will not consult people on the blocklist and thus can limit the scope of the malicious acts during the question matching or recommendation phases.

E. Comparisons

Gnutella [11] uses flooding to query for a file, which can be bandwidth-intensive and slow in general. On the other hand, semantic routing [12] records each node's interests, and routes a query only to nodes with matching interests, improving both the speed and scalability of a system [13]. The query for responders via semantic routing thus may be more efficient than blind flooding. Although the query spread in Connet looks similar to flooding, as the contacts are known friends or people with similar interests, it is not flooding among unrelated people, but a semantically relevant search among people with semantic relations. As such, its performance may be better than random flooding. Both *Tribler* [3] and *Maay* [4] are also P2P networks based on social networks. Connet is similar to them by leveraging social groups to enhance search results. However, their targets are the search for files or documents, whereas we focus on the search and availability of responders.

VI. Conclusion

Connet allows inquirers in an IM network to find suitable responders by searching the social networks of contacts based on previously answered questions and recommendations. It distributively collect people's knowledge profiles for the search, thus protecting people's privacy by allowing users to reveal little beyond their immediate contacts. The potential responders come from 1) system selection, which finds people who have previously answered similar questions, and 2) manual recommendation, which leverages the collective social networks and people's knowledge to discover better responders. Subsequently, results of the interactions are feedback to the system to improve both the knowledge base and the reputations of the recommenders. As more users participate, the search may become more accurate and efficient. Question matching is a key problem that will affect search quality, organizing the questions by tagging thus could narrow down the search scope, such that only questions in related domains are compared for similarity. We envision that a successful deployment of Connet will create new ways for online socialization, and we currently are evaluating Connet through an online implementation.

References

[1] "Wikipedia," http://en.wikipedia.org/wiki/Instant_messaging, 2007.
[2] "Spock," http://www.spock.com, 2007.
[3] J. Pouwelse *et al.*, "Tribler: A social-based peer-to-peer system," in *Proc. 5th Intl. Workshop on Peer-to-Peer Systems*, 2006.
[4] F. D. Ngoc, J. Keller, and G. Simon., "Maay: A decentralized personalized search system," in *Proc. ISAI*, 2006.
[5] T. Cox and M. Cox, "Multidimensional scaling," Chapman & Hall, 1994.
[6] P.-N. Tan *et al.*, "Introduction to data mining," Addison-Wesley, 2005.
[7] Y. Li *et al.*, "An approach for measuring semantic similarity between words using multiple information sources." *IEEE TKDE*.
[8] G. Miller, "Wordnet: A lexical database for english." *CACM*, vol. 38, no. 11, pp. 39–41, 1995.
[9] Y. Li *et al.*, "Sentence similarity based on semantic nets and corpus statistics," *IEEE TKDE*, vol. 18, no. 8, pp. 1138–1150, 2006.
[10] "Yahoo! answers," http://answers.yahoo.com, 2007.
[11] "Gnutella," http://www.gnutella.com.
[12] A. Grespo and H. Garcia-Molina., "Semantic overlay networks for p2p systems," in *Stanford Technical Report*, 2002.
[13] S. Joseph, "Neurogrid: Semantically routing queries in peer-to- peer networks," in *Proc. Intl. Workshop on Peer-to-Peer Computing*, 2002.