Searching For Expertise in Social Networks: A Summary of Popular Systems

Lu Yang
Helsinki University of Technology
luyang@hut.fi

Abstract

Seeking answers to questions always occurs in our learning and daily interactions. Before the computer era, we often got answers from acquaintances and books. Today, appearance of the Internet provides us with many new ways to gain answers such as searching web pages by using search engines,posting questions to forums or sending emails to friends. However, in many cases, seeking answers is still time-consuming. If all questions could be answered by a knowledgable person in real-time, it would be much more efficient. Fortunately, systems for fulfilling this need have been designed, that is "Expertise Location Engine". In this paper, we introduce and summarize three of them, and attempt to get some clues about what tradeoffs should be considered while designing an expertise location engine.

KEYWORDS: Social Network, Search Strategy, Expertise Location Engine

1 INTRODUCTION

In daily life, we often face situations that may require answers or advices. A student may need help to solve a mathematical problem. A software engineer might be blocked at a certain bug while developing. A new employee may need advices about how to start a work. Similar scenarios happen anywhere anytime. In these situations, finding the right person who has required knowledge is the key of solving problems. Traditionally, we used to get help from acquaintances and books. However, acquaintances are not always nearby and even if they are available, they may lack needed knowledge. And even though you can get a right book, locating valuable information is a time-consuming process, the rather that sometimes you have no idea about which book is the right one at all. The appearance of the Internet greatly broadens our sight and provides us a new source to get answers from. We can search web pages by using search engines, pose questions to forums, send emails or instant messages to friends. But these ways of seeking information may still cost us a lot of time and energy. If we could find a real-time way, it will be very convenient and efficient. Systems that help find other people with appropriate expertise are called expertise finders or expertise location engines. [11] As a part of research on social network, expertise locating has been explored for many years and some systems have been designed and developed. Yenta, Expertise Recommender, and Connet are three popular ones. Yenta locates a person with required knowledge by searching email archives; Expertise Recommender does it by recommending sets of potential answers for queries; Connet does it by searching contact list of an instant messaging software in a peer-to-peer manner. This paper surveys these three systems and tries to get some clues about what tradeoffs should be considered while designing an expertise location engine.

The paper proceeds as follows: First, we look back the history of research on searching in social networks. Second, we illustrate three expertise location engines: Yenta, Expertise Recommender and Connet. Third, we make a comparison among them and describe some factors influencing the design of an expertise location engine. At last, we discuss design implications.

2 BACKGROUND

In late 1960, Milgram and Travers found that subjects can send a small packet to a target by passing the packet to an acquaintance closest to the target, even though they only have local knowledge of their acquaintances. Milgram and Travers proved that the average length of acquaintance chain is six, which is known as "six degrees of separation".[7][10] This theory proves that a large scale social network is searchable and marks the beginning of research on "small world" problem. However, in practice, there are often a large amount of candidates to select from as the next person. How to select one to lead to a short chain? What are the criteria we should follow? Later,geographic proximity and similarity were proved to be the most popular criteria. [1][4] From the year of 2000, mathematical models were constructed to evaluate those criteria based on an assumption that social networks usually have a structure. Meanwhile, more and more algorithms were proposed to locate the next person. [5] These studies on the small world problem led to two directions. One is to search a specific person within a social network based on an unique identifier. The other is to locate an expertise.

3 EXPERTISE LOCATION ENGINES

3.1 Yenta

3.1.1 General Description

Yenta is a multi-agent referral-based matchmaking system. It aims to introduce users who are interested in similar topics and provide a way to enhance collaboration on the Internet.

Yenta makes use of a multi-agent strategy and decentralized peer-to-peer architecture. It is designed to find groups of people with similar interests and bring them together to form coalitions and interest groups[2]. Its intended working environment is the Internet where there are hundreds of millions of users and agents. This presents some difficult coordination problems such as how to organize persons' interests to make it easier to pick, and how to locate another agent. Randomly retrieving every agent is extremely slow and obviously impossible.

The core idea of Yenta is like this. Firstly, comparing the agents' information in a peer-to-peer decentralized fashion. Secondly, using referrals to find an appropriate agent to search for relevant peers and build clusters of similar agents. Thirdly, introducing users to each others within clusters built in advance and enabling messaging among users. Finally, a persistent agent always runs in background to find and join appropriate groups of agents whose users share the same interests.

3.1.2 Approach

- Capturable and comparable interest
  Interest should be capturable and comparable in some computer-based form. Yenta mainly uses texts as a source of checking interest. The texts can be obtained in many electronic ways such as emails, newsgroup articles and so on. Yenta uses a term grain to refer to any individual chunk of bits associated with a user. By comparing interests of different users, similar grains can be picked out. A collection of such grains is called a granule. For example, user A is interested in music and book. Two granules are created to reflect these two interests respectively. If another user B is interested in book and car. Then A and B forms a cluster based on their common interest book. This is shown in Figure. 1.

- Determination of similarity
  As mentioned above, Yenta organizes users' interests to several clusters according to similarity. Similarity is obtained by using a keyword-vector text comparison metric. Firstly, normalize a given document by removing unnecessary information and compute an inverse-frequency metric for each word. Secondly, compute a vector describing the document based on the result of the first step. Finally, take dot-product of the associated keyword vectors to compute similarity for different documents.

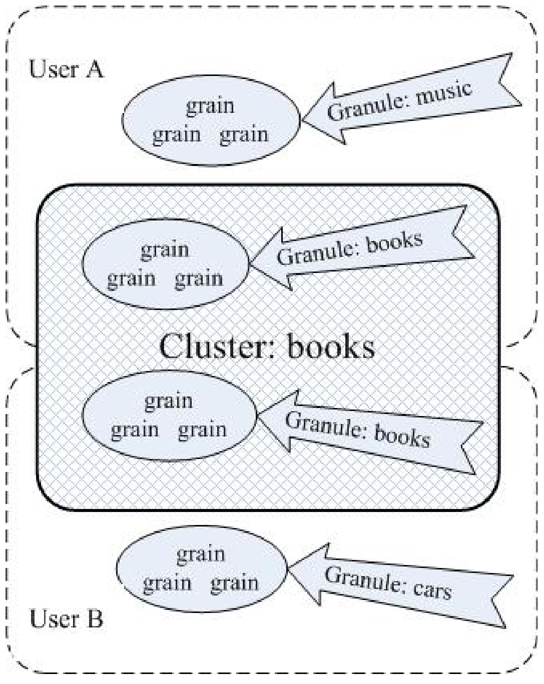

Figure 1: Grain,Granule and Cluster

- Forming clusters of similar agents
  Measuring similarity aims to form clusters. Every operation later will be based on clusters, so this is the heart of the clustering algorithm. Yenta does it in three steps: pre-cluster, bootstrapping and walking through clusters. (i)Pre-cluster is a process of intra-agent initialization. In this step, an agent determines its user's interest in the form of grain. Each grain is then converted to a keyword vector. By comparing with all other grains within the same agent, granules are created. (ii)Bootstrapping is to find at least one other agent with which to communicate. Many technologies can be raised including broadcast, directed multi-cast, asking a central registry and asking users for suggestions. (iii)Walking through clusters is to form clusters of like-minded agents and is based on the former steps. In other words, there should be some places used to store the result of previous steps, which are shown in Table 1.

After an agent A has found at least one other agent B, the process of getting referrals and doing clustering starts, which is very simple. A compares its granules with those of B and B also does the same thing. If matches found, both A and B update their cluster caches to denote they belong to the same clusters on some interest. At the same time, the rest of the data is added to A's rumor cache and its name-list is updated (adding B). If no matches found, A repeats the same process on those agents stored in B's rumor cache. Many agents repeat this process and finally build up a network.

3.2 Expertise Recommender

3.2.1 General Description

The Expertise Recommender(ER) is actually an architecture that has three very good features. First,it can be tailored

Name	Description
Claster cache (CC)	Names of all other agents current known
Rumor cache (RC)	Names and information from the last r agents that this agent has communicated with. Information contains some sub or complete set of text
Pending-contact (PC)	A priority-ordered list of other agents that have been discovered but which the local agent has not yet contacted

Table 1: Data structures used in forming clusters

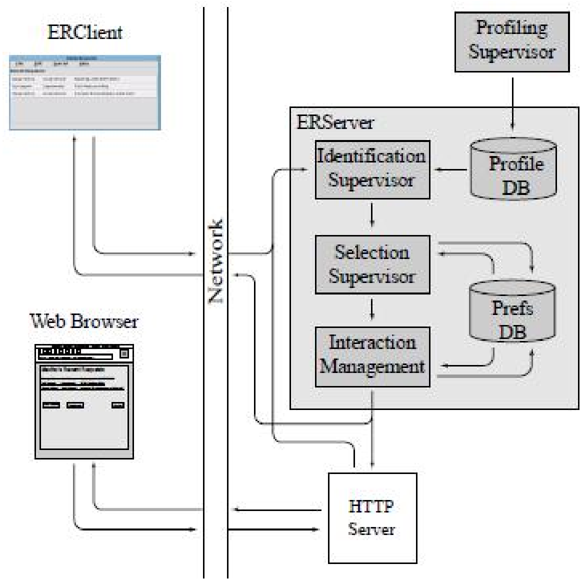

Figure 2: Expertise Recommender Architecture

to many different recommendation situations and is highly suited for expertise recommendation. Second, profiles' generation is based on work products and work byproducts. Third, it teases apart technical concerns involved with making recommendation from the social and collaborative concerns. Therefore, ER can be regarded as a design pattern to some extent. However, compared with design pattern, it provides much more details. In addition, unlike two other systems mentioned in this paper, ER is based on a central server.

3.2.2 Architecture

At a high level, ER is a pipe and filter architecture[8] and is composed of a set of high-level supervisors, easily extensible heuristic modules and their data stores. Supervisors provide general services and are responsible for profile management, identification selection and interaction.The ER architecture is shown in Figure. 2[6].

- ER Server and Client ER Server manages connections and service requests. It defines the rules of communication between clients and server, and sending request and receiving recommendations. There are two kinds of client: ERClient and web browser. In other word, ER supports client/server(C/S) and browser server(B/S) architectures at the same time, which also provides more flexibility.

- Profiling Supervisor
The profiling supervisor is responsible for creating and maintaining profiles. In reality, profiles are stored in database as raw data and one of functions of profiling supervisor is to transform those raw data into profile records. There can be different rules used to do transformation because of different types of data source and purpose. To enhance the flexibility, profiling modules are often defined. According to a predefined common interface, developers are able to implement their own modules and profiling supervisor. Furthermore, in some situation, profile records are needed to be generated from more than one single source and the profiling supervisor coordinates the modules and provides access to the profile database. Generated profile records are stored back into profile database.

- Identification Supervisor
Profile database stores profile records generated by profiling supervisor. Then, identification supervisor picks items from those profile records according to one or more criteria and adds them into a set named recommendation list. Like profiling supervisor, developer are also able to implement different types of modules to identify and the identification supervisor is responsible for coordinates these modules. In addition, criteria used to pick can be indicated by users through ER client.

- Selection Supervisor The input to the selection supervisor is a raw recommendation list. Based on the preference stored in preference database, the selection supervisor reorders or removes items from the list to generate a refine recommendation. Like before, different selection modules can be implemented.

- Interaction Management Interaction management is responsible for communicating with users. It processes the data in the refined recommendation list to generate a final recommendation, tracks and manages user connections, collects feedback and solves issues.

In a word, ER architecture is not based on any specific recommendation application and provides much flexibility. Profiling, selecting, identification and interaction processes data in different levels, and modules can be defined for any data source or users.

3.3 Connet

3.3.1 General Description

Connet is a peer-to-peer(P2P) based people searching system that allows one to find relevant online persons who can answer specified inquiries easily and immediately through people's collective social networks.[3] It makes use of a fact that instant messaging systems have formed a large-scale social network connecting users by contact lists, and focuses on locating an online person who can answer questions in real-time. The core idea is that searching a social network contracts by contracts, which is feasible because most of users' contract are limited and short.

3.3.2 Design of Connet

Connet works like this. When a question is asked, Conect searches through users' contacts based on their interest keyword in profiles or those questions they have answered. Inquirer is also allowed to select a user as the starting point of searching through. Searching continues until a user is found, who should satisfies three criteria. First, he or she is online. Second, he or she is not answering questions at that moment. Third, he or she has successfully answered similar questions. After answering questions, both questions and conversing parties are recorded for future use. The phases of connet are shown in Figure 3.

- Initialization
  When a new user joins Connet, he or she firstly registers to get a new account identified by an unique ID. In addition, new users are mandatory to add their friends to contact list and the keywords of those question they wish to answer. If there are no entries added to contact list, Connet will do it by randomly selected among those who have similar keywords. Multidimensional scaling(MDS)[9] is used to judge similarity.

- Asking and matching questions
  Asking a new question triggers a search on the user's contact list. When a contact receives a question, a similarity measure between this question and those questions which have been asked by the contact is calculated. If the result beyond a predefined threshold, the contact and the inquirer are both added into a recommendation list. there are many ways can be adopted to measuring similarity such as Jacquard and semantic analysis.

- Recommendation
  Human judgement is always more accurate than automatic way. Hence Connet allows any one contact recommend his or her friends to answer questions and the contact itself is named introducer. To evaluate the reliability of recommendation, each contact is assigned an appraisal value. Contact with higher appraisal value is more likely to recommend a right responder.

- Ranking
  After the process of searching and recommendation, a recommendation list is sent back to inquirer. It consists of candidate responders, inquirer, introducer(optional) and similarity measure. Ranking is based on relevance and confidence. Relevance means the pertinence between new asked questions and previous ones, which is the sum of similarity measures of a responder. Confidence denotes the reputation of users and is the sum of appraisal values of a responder.

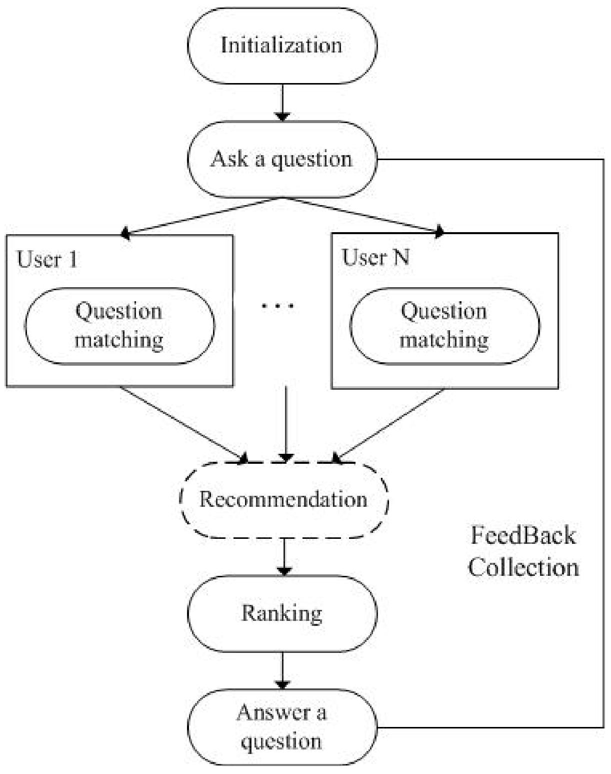

Figure 3: Phases of Connet

- Feedback Collection
  Once a communication between inquirer and responder is finished, the inquirer can give an evaluation on the conversion. Simply, this evaluation is measured by positive or negative remark. The responder's appraisal value is then increased or decreased based on the remark.

In addition to the basic mechanisms above, Connet also provides four enhanced functions to make it more useful in practical scenarios. Message board allow responders still have chance to answer. Anonymous inquiry allows inquirers ask questions anonymously. Answer points are given to every users so that they can trade their own answer points for a successful answer. Block list is used to block those malicious users.

4 COMPARISON

4.1 Centralization vs. Decentralization

Yenta is built on a decentralized peer-to-peer architecture. Despite Connet takes the idea of peer-to-peer, a central server is still needed for account management. Hence we named Connet a partial decentralized system. Compared with above two, Expertise Recommender(ER) makes use of a completely centralized architecture.

Centralization and decentralization of course have their own pros and cons. In a system built based on centralized architecture, one or more server control the system nearly completely. Introducing new services or modules just requires limited changes on server side, and does not affect client side at all. Considering the scenario of expertise location engine, one core problem is how peers are supposed to find each other. A centralized system inherently solves this problem, because a server is able to provide all information and do this matching. However, scaling such an architecture to large numbers of users is very difficult. In systems which must correlate user interests, if all interest matching are done in server side, the server may be burdened very much. Another problem is that a centralized system can not provide high availability. It only provides a single point and its accidental failure results in the whole system failure at once.

Compared with centralized architecture, decentralized architecture is easier to scale to a large number of users and also provides high availability. Matching operations are done in client side and hence consume client-side computational resources. Even if an agent breaks down, there are still many other agents working. The biggest problem of decentralized architecture is how agents are supposed to find each other. It is not so straightforward as in centralized architecture because every agent should not have any idea about others in advance.

4.2 Dependence vs. Independence

Yenta and Expertise Recommender is self-contained and independent, while Connet makes use of existing instant message system. Relying on an existing system surely provides many advantages. A popular instant message system has offered a large-scale social network,so there is no need to do it from the starting point. Contact list is also a good resource to locate expertise. Hence using an existing system is efficient. However, do instant messaging service providers wish to integrate such a expertise location engine to their successful product? Adding new system unavoidably brings in risks such as new potential bugs and vulnerabilities. Another problem stems from the user of instant messaging system. For example, in China, MSN Messenger is often used as a communication tool in working environment. In spare time, another instant messaging product QQ is often used, which provides much more colorful services. While a working user probably may not wish to be an expert to answer questions for those who are outside his or her company. In this situation, integrating expertise location engine to MSN Messenger is really not a good idea. Third, building a independent system is much more flexible than utilizing an existing one.

5 CONCLUSION

In the preceding sections, I have tried to describe those three expertise location engines and discussed two pairs of converse trade-offs. But I have not answered which one is better in both trade-off pairs. As a conclusion, I think there is not a single answer. It depends on different situations and applications. Yenta aims to provide a common scheme to enhance the collaboration on network, and hundreds of millions of user are always involved, so it utilizes decentralized peer-to-peer architecture. Expertise Recommender more emphasizes flexibility and aims to provide an architecture which can be applied to different specific applications. In this situation, the amount of users are not very large and complete control of the system is more important, so a centralized architecture is adopted. In a word, different purposes result in different choices.

References

[1] K. P. D. M. C. Bernard, H. R. Index: An informant-defined experiment in social structure. In *Social Forces*, volume 61(1).

[2] L. N. Foner. A multi-agent referral system for matchmaking. 1996.

[3] Jyun-Jie Huang and Shao-Chen Chang and Shun-Yun Hu. Searching for answers via social networks. pages 289–293, Las Vegas, Nevada, USA, 2008. IEEE. ISBN:1-59593-223-2.

[4] P. Killworth and H. Bernard. Reverse small world experiment. In *Social Networkss*, volume 1.

[5] J. Kleinberg. Navigation in a small world. In *Nature,*, volume 406.

[6] D. W. McDonald and M. S. Ackerman. Expertise recommender: a flexible recommendation system and architecture. In *CSCW '00: Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 231–240, New York, NY, USA, 2000. ACM. ISBN: 1-58113-222-0.

[7] S. Milgram. The small-world problem. In *Psychology Today*, volume 1.

[8] M. Shaw and D. Garlan. Software architecture perspectives on an emerging discipline. 1996.

[9] T.Cox and M.Cox. Multidimensional scaling. 1994.

[10] M. S. Travers, J. An experimental study of the small world problem. In *Sociometry*, volume 32.

[11] J. Zhang and M. S. Ackerman. Searching for expertise in social networks: a simulation of potential strategies. In *GROUP '05: Proceedings of the 2005 international ACM SIGGROUP conference on Supporting group work*, pages 71–80, New York, NY, USA, 2005. ACM. ISBN:1-59593-223-2.