QuME: A Mechanism to Support Expertise Finding In Online Help-seeking Communities

Jun Zhang
School of Information
University of Michigan
junzh@umich.edu

Mark S. Ackerman
Dept. of EECS and
School of Information
University of Michigan
ackerm@umich.edu

Lada Adamic
School of Information
University of Michigan
ladamic@umich.edu

Kevin Kyung Nam
School of Information
University of Michigan
ksnam@umich.edu

ABSTRACT
Help-seeking communities have been playing an increasingly critical role in the way people seek and share information. However, traditional help-seeking mechanisms of these online communities have some limitations. In this paper, we describe an expertise-finding mechanism that attempts to alleviate the limitations caused by not knowing users' expertise levels. As a result of using social network data from the online community, this mechanism can automatically infer expertise level. This allows, for example, a question list to be personalized to the user's expertise level as well as to keyword similarity. We believe this expertise location mechanism will facilitate the development of next generation help-seeking communities.

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

General terms: Design, Human Factors, Algorithms
Keywords: Expertise Finding, Expertise Location, CSCW, Social Networks

INTRODUCTION
Help-seeking communities have been playing an increasingly critical role in the way people seek and share information online. For example,

- Yahoo Answers has 30 million plus answers and gains 2500+ new questions every hour.

Innumerable additional sites exist, from online stock trading discussions to medical advice communities. In these communities, people help strangers voluntarily for various motivations, e.g. altruism, incentives to support one's community, reputation-enhancement, expected reciprocity, and direct learning.

Unfortunately, these help-seeking communities are often primitive technically; we would like to find mechanisms to augment their functionality and social life. Research is proceeding to make use of the available information in online communities to design new systems and algorithms. For instance, ContactFinder [4] used text and addresses of messages on bulletin boards to find the right person to answer specific questions.

Systems like ContactFinder are called expertise finders [7]. Expertise finders are CSCW systems that help locate people with specific information or expertise, for example, to help answer questions in online communities. They are an important class of recommender systems, but they suffer from a general problem: Current expertise finders cannot infer expertise levels very well. Traditionally, expertise finders have relied on the standard information similarity measures (such as term vector comparisons). Being able to add the level of expertise would be a major step forward for expertise finders, and would likely open up a range of new application possibilities.

In this paper, we report on a new technique for inferring expertise levels. We first analyze expertise-finding problems in one help seeking community, the Java Forum [2]. Then, we describe how adding expertise level could alleviate these problems. We then describe our algorithm for inferring expertise level for the forum participants as well as the QuME system and interface as a prototype system that uses this algorithm.

EXPERTISE FINDER PROBLEMS IN AN ONLINE HELP-SEEKING COMMUNITY

The Java Forum
The Java Developer Forum is an online community where people come to ask questions about Java. There is a large diversity of users, ranging from beginners learning Java to the top Java experts. There are approximately 17,900 users.

The interface for the Java Forum is basically a standard web-forum. If a user posts a question, it is listed in the forum along with the thousand other questions waiting for a potential helper to see and answer it. The questions are generally ordered by the latest updates in the question threads.

Consistent with studies of other forums, we found that on the Java Forum:

- More than 55% of users usually only ask questions, while there are about 25% of users who are core users who regularly ask and answer questions.
- Many questions are answered by few advanced users while a majority of users only answer a few.

UIST'07, October 7–10, 2007, Newport, Rhode Island, USA.

- Top repliers answer questions for everyone. However, less expert users tend to answer questions of others with a lower expertise level.

These findings made us wonder whether the Java Forum's current form is the best way for a community to work, or whether we could build new mechanisms and interfaces to improve the current functionality. In particular:

- Could we motivate those users who have not answered questions to answer a few?
- Could we help expert users make better use of their time?

These questions motivated us to look further at the details of question asking and answering activities in the Forum, which in turn revealed two problems that could be improved.

The Response Time Gap

First, we found a significant difference regarding the first response time between high expertise users' questions and low expertise users' question. High expertise users' questions have a statistically significant longer waiting time to get a first answer. The average waiting time of a high expertise user to get a reply for a question is about 9 hours (s = 191 min.), compared with 40 minutes (s = 23 min.) for a low expertise user.

The result can be explained intuitively: High expertise users ask harder questions than low expertise users. The higher one's expertise is, the less chance that others will be capable enough to answer one's questions.

While the low availability of high expertise users in the forum is the main reason for this situation, the current interface makes the problem worse. The high expertise users' questions are often lost in the flood of newbie questions. If we follow Lakhani et al.'s [5] findings in their study of the Apache Support Forum, "direct learning" is a primary reason for users to provide answers. However, the cost of a question and answer match-up falls upon the potential helper, since helpers accomplish the match-up task by reading or scanning questions posted in the forum. The current interface does not support the "question-helper" match-up tasks well. Thus, one possible improvement is to bring these questions to the attention of the more capable users.

The Expertise Gap

As mentioned, high expertise users tend to answer questions for everyone. However, an expertise gap problem can happen when experts try to share their expertise with novices [1]: An expert's instructions to novices may be too difficult for novices to grasp. A range of research shows that novices perform better on a target task when instructed by those with less expertise (e.g. [1]).

This problem can be seen in a Java Forum exchange. A user asked how to read a file that included both ASCII and binary data. The first response was from a top expert:

"Read it all as bytes. If you have an array of bytes that represent ASCII text then there's an obvious String constructor that makes them into a String."

The expert assumed that the asker knew Java well and just provided a general solution. However, the asker did not know the techniques mentioned and asked for clarification, which another user answered at a different level:

"... [use] the method readFully(byte[] b) ...As for reading from a file and buffering, the tutorial (in java.sun.com) goes into that."

This answer is likely to be easier for a novice to follow.

Both of the above problems, the time-to-response for questions posed by high-expertise users and the expertise gap between asker and replier, could be addressed by properly matching the expertise level of asker and helper. While expertise level is in general difficult to determine in an automated way based purely on textual content, we are able to automatically infer expertise by constructing a community asker-helper network based on historical posting-replying data. Note that this readily accessible network data is distinct from acquaintanceship network data more commonly used to map social networks of individuals, which in this case may be neither available nor relevant to expertise evaluation. We describe our algorithm in the next section.

EXPERTISE RANK BASED MATCHING ENGINE

Asker-Helper Network and Expertise Ranks

The asker-helper network is a network constructed using posting/replying threads in a community by viewing each participating user as a node, and linking the ID of an asker to a replier's ID, as shown in Figure 1.

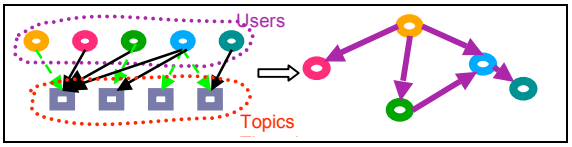


Figure 1: Constructing the asker-helper network

After the network is constructed, various algorithms can be used to calculate users' relative ranks in the network. Since a user replying to another user's question usually indicates that the replier has superior expertise on the subject than does the asker, a higher network rank thus indicates a higher expertise level. We call these ranks Expertise Ranks (ER). Our expertise ranking algorithms include: Indegree (how many people one helps), a Z_score measuring the proportion of people one has helped vs. received help from, an adaptation of Google's PageRank algorithm using the network of people helped [5], and an adaptation of the HITS algorithm for identifying authoritative sources [3]. (These are detailed in [8].) Since these ranking algorithms perform differently in different communities, one can decide which algorithm to use within our system based on the analysis of the community's network characteristics.

In the Java Forum's case, we found the Z_score provided a satisfactory result and was simple to calculate:

$$ER_{Zscore}(U) = \frac{n_{in}(u) - n_{out}(u)}{\sqrt{n_{in}(u) + n_{out}(u)}}$$

where $n_{in}(u), n_{out}(u)$ correspond to the number of people one helped and the number of people one was helped by. Z_score will be important to the matching algorithm described below. However, we anticipate that the other measures may perform better in more densely connected answer networks (such as those within companies).

Using these ranking algorithms, we can now construct enhanced expertise profiles that not only include the keywords extracted from users' posts, as traditional expertise finders do, but also a ranking score that indicates a user's relative expertise level in the community.

Question Matching Algorithm

The expertise profile can then be used in a matching engine to process how questions are matched to a potential helper. Here is the calculation that is at the heart of our question-matching algorithm:

$$M(q,u) = \alpha \cdot kmatch(q,u) + \beta \cdot ediff(a,u) + \gamma \cdot recency(q) + (1-\alpha-\beta-\gamma) \cdot stat(q)$$

In this formula, α, β, γ are tunable parameters and $\alpha + \beta + \gamma \le 1$. q represents the question, u represents the user, and a represents the asker.

$kmatch(q,u)$ is the keyword matching score between the question and the user's profile. It is calculated using cosine similarity between the two term vectors, a standard technique used in information retrieval.

$ediff(a,u)$ represents the expertise rank difference between a user and the asker of the question. We define *ediff* to give higher weight to an *ER* difference where the replier is a bit more expert than the asker: $ediff(a,u) = e^{-[(ER(u)-ER(a))-\lambda\cdot\sigma_{ER}]^2/(2\cdot\sigma_{ER}{}^2)}$

We used ER_{Zscore} to calculate *ER* in the prototype described below. σ_{ER} is the standard deviation of expertise ranks for all users, and λ is a tunable parameter. For example, if we set λ =0.5, it means that a user is most likely to help another user who is half a standard deviation below them in expertise.

$recency(q)$ is calculated as $e^{-\tau(now-timePosted)}$ where τ is tunable. The recency score will range from 0 for a very old post to 1 for a current post.

$stat(q)$ is calculated as $e^{-\kappa(Num\,Replies)}$. Thus, questions already being answered are ranked lower.

Above all, this formula gives us the flexibility to build a completely tunable system. First, we can select the relative importance of expertise level match, topic similarity, recency, and status. Second, we can set the parameter to decide how recent is recent, and how close the expertise match needs to be.

In the implementation of the prototype below, we de-emphasized the keyword matching factor because a high expertise user can usually answer a large range of questions even if his or her profile does not contain the particular keywords from the question. And we wanted to emphasize matching advanced questions with top experts. Thus, we set $\alpha = 0.1, \beta = 0.5, \gamma = 0.2$. As well, based on our analysis of the Java Forum, we wanted to match users to answer questions from people who are 1/3 standard deviation below them in expertise, thus we used λ =0.33. We set τ=0.7, so that the recency score of a question will drop by about half every hour. And κ=1, so that questions with no or few replies receive high scores.

PROTOTYPE INTERFACES FOR JAVA FORUM

QuME Engine

Figure 2 shows the system structure of the Question Matching Engine (QuME), which implements the techniques described above for a new forum prototype.

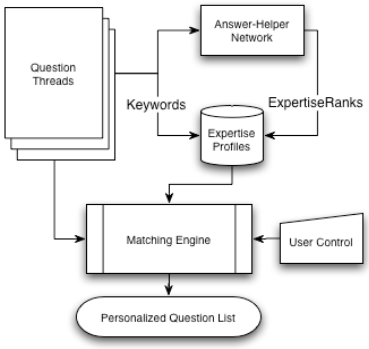


Figure 2: The system structure of QuME

The expertise rank based profiling and matching engines in QuME bring many new design opportunities to personalize the interface for online help-seeking communities. We built several new interfaces for the Java Forum to demonstrate these possibilities. These prototypes use real time data from the Java Forum, allowing for easy comparison with the original interface.

Personalized Java Forum Web Interface

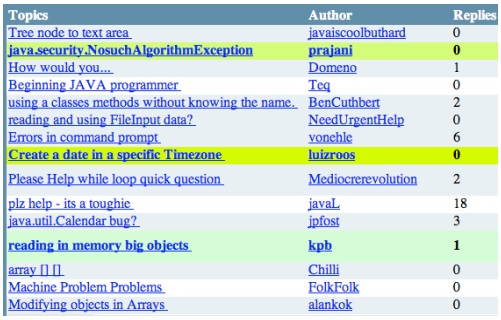
Figure 3 shows two screenshots of an interface that is very similar to the original Java Forum. However, in this new interface, the order of the questions is customized for users according to their expertise profiles. The screenshot in the front is what a high expertise level user would see, and the one in the back is what a low expertise level user would see.

Note that the bar next to author's name indicates her Expertise Rank score. Questions are listed in a roughly descending order according to the askers' expertise ranks. Thus, a user will first see the questions that are slightly

Figure 3: Screenshot of a personalized interface re-ordering questions by goodness of match.

below his expertise level. These questions have a higher probability of allowing the answerer to gain new experience while still being capable of providing answers. The expertise-level bar will also help users know whom they are helping.

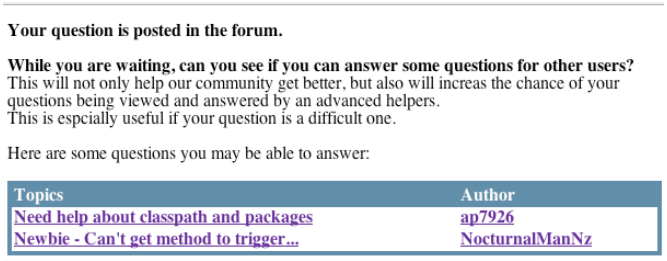
Completely re-ordering the questions for each user may be too aggressive. Figure 4 shows a less aggressive interface design using the QuME algorithm. In this interface, the order of the questions is the same as the original Java Forum. However, the questions with a matching score above a threshold are highlighted. Thus, it can help helpers quickly locate questions that they are capable of answering.



Topics	Author	Replies
Tree node to text area	javaiscoolbuthard	0
java.security.NosuchAlgorithmException	**prajani**	**0**
How would you....	Domeno	1
Beginning JAVA programmer	Teq	0
using a classes methods without knowing the name.	BenCuthbert	2
reading and using FileInput data?	NeedUrgentHelp	0
Errors in command prompt	vonehle	6
Create a date in a specific Timezone	**luizroos**	**0**
Please Help while loop quick question	Mediocrerevolution	2
plz help - its a toughie	javaL	18
java.util.Calendar bug?	jpfost	3
reading in memory big objects	**kpb**	**1**
array [] []	Chilli	0
Machine Problem Problems	FolkFolk	0
Modifying objects in Arrays	alankok	0

Figure 4: Screenshot of personalized interface highlighting well matched questions.

There is one potential problem in the matching engine. There are many new users joining a community every day. Since they have no answering history in the forum, their automatically calculated expertise ranking is the lowest possible. However, a new user is not necessary a newbie in the specific knowledge domain. To work around this problem, the system prompts a new user to answer some questions after they submit a question, as shown in Figure 5. These questions are picked according to their askers' rankings in an ascending order. A user who does not answer any of the questions is assigned to the lowest rank.



Your question is posted in the forum.

While you are waiting, can you see if you can answer some questions for other users?
This will not only help our community get better, but also will increas the chance of your questions being viewed and answered by an advanced helpers.
This is espcially useful if your question is a difficult one.

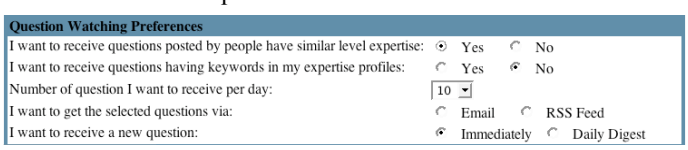
Here are some questions you may be able to answer:

Topics	Author
Need help about classpath and packages	ap7926
Newbie - Can't get method to trigger...	NocturnalManNz

Figure 5: Screenshot showing suggested questions to new users.

Personalized Question Feeder

Figure 6 shows another interface. With this interface, instead of coming to the site, the users can subscribe to receive customized questions.



Question Watching Preferences		
I want to receive questions posted by people have similar level expertise:	◉ Yes	○ No
I want to receive questions having keywords in my expertise profiles:	○ Yes	◉ No
Number of question I want to receive per day:	10	
I want to get the selected questions via:	○ Email	○ RSS Feed
I want to receive a new question:	◉ Immediately	○ Daily Digest

Figure 6: The interface for users to subscribe to personalized questions

New questions qualified by these criteria will be sent via email or an RSS feed to the subscribed users inviting them to come to the forum to provide help. If there are a lot of new questions, the system can distribute questions to different users according to their preferences.

Note that many online communities, including the Java Forum, already provide an RSS feed so that a user can subscribe to all messages in a forum. A problem is that there are too many questions, leading to overloading users' email boxes or RSS readers. Our new interface alleviates this problem by only sending a limited number of filtered questions that may be more interesting to potential help providers, thus decreasing the overload issue and increasing participation in the helping activities.

With these interfaces, one can allocate questions more efficiently to various users in the community. An advanced user's question will have a higher probability of being viewed by more advanced users, thus increasing its chances to be answered faster. General users will receive questions that they can answer, and this will encourage them to answer more questions, thus improving their expertise ranking. This will eventually benefit both themselves and the community.

Our system has not yet been evaluated. As the next step, we plan to invite Java Forum users to use our system. We will evaluate both user satisfaction in using the new interfaces, the closeness of the expertise match, and the distribution of reply times for questions of varying difficulty.

ACKNOWLEDGMENTS

This work has been funded, in part, by the National Science Foundation (IRI-9702904). We also wish to thank George Furnas and Derek Henson for feedback and suggestions.

REFERENCES

1. Hinds, P., Patterson, M., and Pfeffer, J. Bothered by Abstraction. *Journal of Applied Psychology,* 86 (6): 1232-1243
2. Java Forum, http://forum.java.sun.com
3. Kleinberg, J.M. Hubs, authorities, and communities. *ACM Computing Surveys*, 31. U21-U23
4. Krulwich, B. and Burkey, C., ContactFinder agent: answering bulletin board questions with referrals. *In Proceedings of AAAI'96*, Portland, OR, 1996, 10-15
5. Lakhani, K. and von Hippel, E. How open source software works: "free" user-to-user assistance. *Research Policy*, 32 (6), 923-943
6. Page, L. PageRank: Bringing order to the web. Stanford Digital Libraries Working Paper, 1997.
7. Terveen, L. and McDonald, D. W. Social matching: A framework and research agenda. *ACM Trans. Comput.-Hum. Interact.* 12, 3 (Sep. 2005), 401-434.
8. Zhang, J., Ackerman, M.S. and Adamic, L. Expertise Networks in Online Communities: Structure and Algorithms, *In Proceedings of WWW2007*, Banff, Canada.