CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2007; **19**:1087–1113
Published online 28 September 2006 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cpe.1094

Networking scientific resources in the Knowledge Grid environment

Hai Zhuge[1,*,†], Lianhong Ding[1,2] and Xiang Li[1,2]

[1] *China Knowledge Grid Research Group, Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, 100080 Beijing, People's Republic of China*
[2] *Graduate School of Chinese Academy of Sciences, 100080 Beijing, People's Republic of China*

SUMMARY

Scientific documents, research instruments, researchers' abilities and co-operation among researchers are fundamental resources of scientific research. This paper proposes an approach for effectively networking scientific resources by detecting community structures in self-organized social network, discovering interest in information flow, capturing changes of interests over time and analyzing the relationship between peers. Copyright © 2006 John Wiley & Sons, Ltd.

Received 28 April 2006; Accepted 28 April 2006

KEY WORDS: e-Science; information flow; information sharing; Knowledge Grid

1. INTRODUCTION

Obtaining up-to-date scientific documents and help from the scientific community is important in scientific research. Some research activities may slow down the efficiency of teamwork; for example, researchers within a team often repeatedly search for the same documents. Appropriate resource-sharing can promote the effectiveness of co-operative research. A precondition of sharing resources is the identification of users' interests.

*Correspondence to: Hai Zhuge, China Knowledge Grid Research Group, Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, 100080 Beijing, People's Republic of China.
†E-mail: zhuge@ict.ac.cn

Contract/grant sponsor: National Basic Research Program of China (973 Project); contract/grant number: 2003CB317000
Contract/grant sponsor: National Science Foundation of China

This paper proposes a scientific resource-sharing approach that actively pushes scientific documents and recommends appropriate helpers by discovering interests and analyzing relationships in information flow within a scientific community.

1.1. Aim and technical path

The research aim is the e-Science Knowledge Grid environment [1], an intelligent Internet-based environment that enables scientists to effectively capture, publish, share and manage explicit knowledge resources.

The technological path of the proposed approach consists of the following four steps.

(1) Building the *general-to-specific* user profile by making use of the information flow. A user profile consists of the community profile and the personal profile. The former represents a community's common preference; the later exhibits the difference between community members.
(2) Constructing the community profile by detecting the common interests of a community in a large social network by graph analysis. Our community-detecting algorithm extends the idea of edge betweenness centrality [2] by introducing a weighting method to differentiate the importance of edges. We also purpose a new set of rules to direct the algorithm to run or stop [3,4]. A user's personal profile is created by mining information flow with reference to the community structures.
(3) Provide appropriate resources (here focusing on scientific documents) for researchers according to subject classification scheme, community profile and personal profile.
(4) Find candidate helpers according to the community structures and to select appropriate helpers from candidates by link analysis [5].

1.2. General architecture

The general architecture shown in Figure 1 consists of the following core components.

(1) *The social network* is a self-organized network that reflects the relationship between members within an organization.
(2) *The Knowledge Grid's resource space model* manages resources in a classification-based semantic space [1]. It comprises information space, document space and knowledge space. Document space has an *area-to-topic* structure, where each area contains disjoint topics. For example, a review paper can be located in its *area*, and a technical paper can be located in a *topic* in a certain *area*.
(3) *The user profiles* give a *general-to-specific* description of member interests and provide support for the document deliverer. A community profile is an area covering all community interests. A personal profile consists of one or several concrete topics in the area that exhibits differences between community members. Community profiles correspond to the area level of the document space, while personal profiles correspond to the topic level.
(4) *The information collector* gathers information flow within the organization and stores useful contents into the information space.
(5) *The network generator* constructs the social network of the organization according to the information flow.

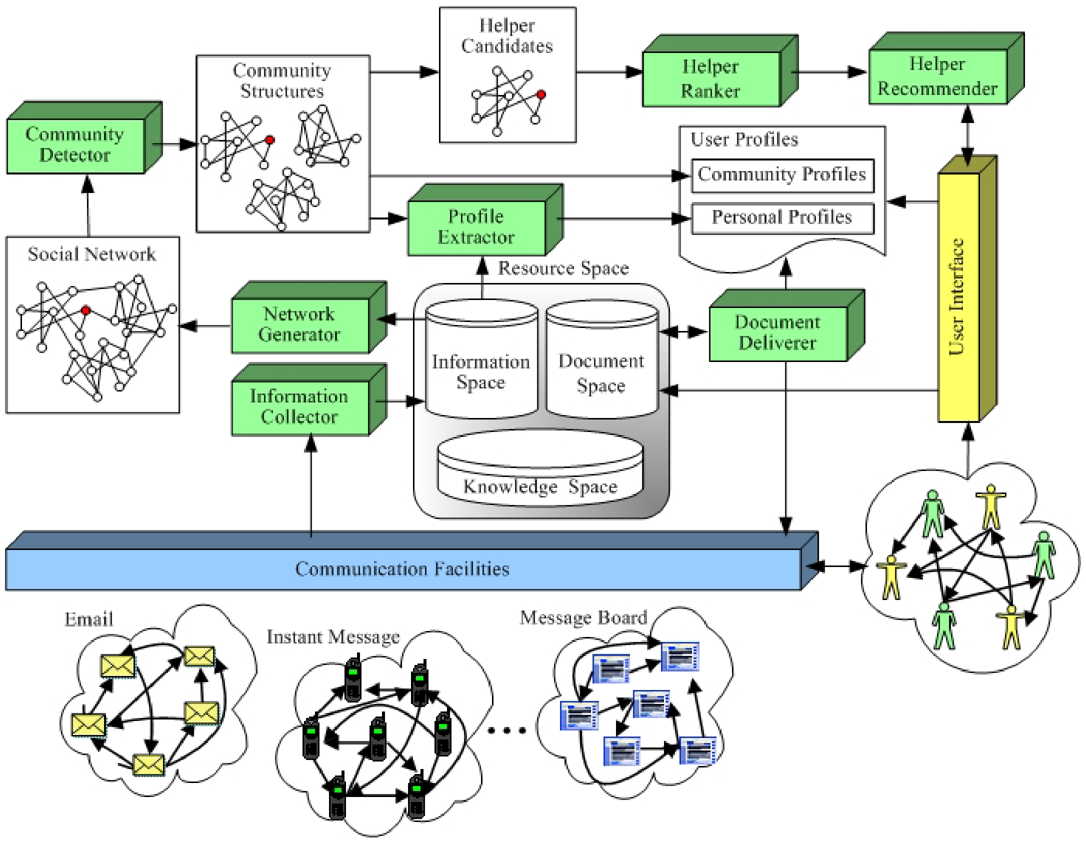

Figure 1. General architecture.

(6) *The community detector* finds common-interest communities in the organization by using the community-detecting method.
(7) *The profile extractor* learns a user's personal profile by mining information flow by referring to community structures.
(8) *The helper candidates* are a set of members within the organization who can help the current user to resolve problems.
(9) *The helper ranker* is responsible for ranking the helper candidates according to their skills.
(10) *The helper recommender* recommends appropriate helpers for users following the ranking results of the helper ranker.
(11) *The user interface* is where by members can tune their own profiles, ask for helpers or upload scientific documents to the document space at any time.

(12) *The document deliverer* is responsible for actively pushing relevant documents to organization members by referring to user profiles.
(13) *The community structures* are used for identifying helper candidates, forming community profiles and personal profiles.
(14) *The communication facilities* support digital communications for peers. It enables the network of peers in the real world to transform to networks in the digital world, such as E-mail flow network and instant message flow network.

1.3. Information collection and service

Information flows in an organization include E-mail, instant messaging and communication via message boards and blogs. A resource has its own profile specifying its type, owner, user instruction and location. Resources in resource space are recommended or searched within the community.

The information collector regularly parses each message and divides it into six parts: *from*, *to*, *date*, *subject*, *body* and *attachment* (if any), unifying the name of the senders and recipients and deleting quotations of other messages and signatures from the message body, then storing them into information space with each message as a record. Here, external messages and messages broadcast to more than 10 recipients are neglected as these messages are usually irrelevant to research.

An owner list is maintained for each document. People who upload a document are added to the corresponding owner list, as are the members who have received that document. Absence in the owner list is a necessary precondition for pushing a document to that person. A review on a certain area at the area level of the document space can be pushed to all members whose community profile corresponds to that area. A document on a certain topic is pushed to the persons whose personal profile contains this topic and the corresponding energy value discussed in Section 3.3 exceeds some threshold: first, we find the common-interest community whose community profile is the area to which the topic belongs, and then choose the right members in this community. Documents in the document space are sent regularly to appropriate members as E-mail attachments by document deliverer.

2. COMMUNITY PROFILES AND COMMUNITY STRUCTURES

Members in a research community share interests. The profile of a community reflects a research area that covers the most interest points of members.

2.1. Social network construction

A social network is a map of relationships between individual activities: vertices represent people, and edges represent communications. Thresholds can be assigned to specify the minimum number of messages passed through edges.

2.2. Community detection

After Freeman's vertex 'betweenness' [6], Girvan and Newman put forward the conception of edge betweenness and the approach to divide a graph into discrete communities of nodes based on the idea of edge-betweenness centrality. The betweenness of an edge is the number of shortest paths that go through it. Communities can be discovered by repeatedly identifying and removing the edges of the highest betweenness because the edges that connect highly clustered communities have higher edge betweenness [2].

In order to obtain the community structures automatically, the following two rules are put forward by Wilkinson and Huberman to direct the Girvan and Newman (GN) algorithm to stop or go on [3,4].

(1) The minimum component rule. The component composed of no more than five vertices should not be partitioned.
(2) The $N - 1$ betweenness rule. The partition process should stop when the highest betweenness is equal to or less than $N - 1$, where, N is the number of vertices. This rule terminates the algorithm before the isolated vertex appears.

We put forward a set of new rules to direct the algorithm to go on or stop, and extend the GN algorithm by introducing edge weight to participate the calculation of the betweenness.

2.2.1. *New rules*

We have implemented the GN algorithm that follows the minimum component rule and $N - 1$ betweenness rule and applied it to the social networks constructed for a research team. Figure 2 illustrates the partition results, where dotted lines are the edges that have been removed by the algorithm. When the threshold = 1, the algorithm stops with three edges removed by the $N - 1$ betweenness rule. When the threshold = 20, it ends with six edges removed for the same reason, where node 23 is an isolated vertex in the social network itself. From the above examples, we can see that the $N - 1$ betweenness rule makes the partition process stop too early.

The partition results of the algorithm discarding the $N - 1$ betweenness rule are given in Figure 3. We can see that discarding the $N - 1$ betweenness rule leads to too many isolated vertices. Therefore the rules proposed previously need modification, especially for the small-scale networks.

We replace the $N - 1$ betweenness rule by the following rule:

New Rule 1: *Remove the edge with the second highest betweenness if the highest betweenness is equal to or less than $N - 1$ and the component is still large enough.*

The number of the vertices in a component can be used to judge if the component is still big enough. It may be different for different scales and different aims.

New Rule 2: *A complete graph should not be partitioned further if $E = N(N - 1)/2$ holds, where E is the number of edges in the component and N is the number of vertices.*

New rules together with the minimum component rule constitute a more reasonable termination condition.

The partition results using the new rules are given in Figure 4. Taking threshold = 1 for example, after the three edges linking node 4 and 13, 4 and 7, 4 and 19 are removed in order, the edge linking node 4 and 21 becomes the one with the highest betweenness. Because the deletion of it will make node 4 isolated, according to the new rules, the edge linking node 13 and 21 is removed instead, which makes the algorithm continue. The complete graph rule newly introduced reserves the community of nodes 3, 6, 13, 15, 16, 22 and 24, although there are more than five vertices.

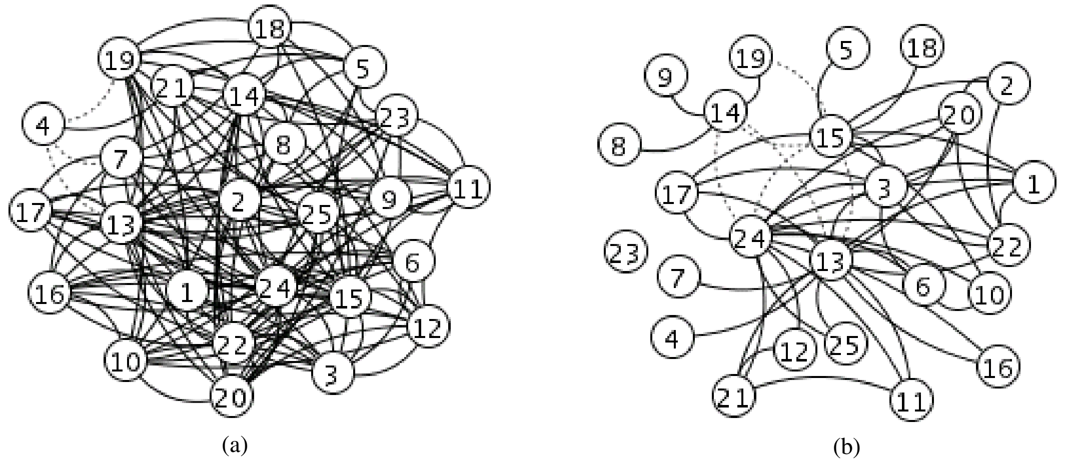

Figure 2. Partition results using the minimum component rule and the $N - 1$ betweenness rule; (a) threshold = 1; (b) threshold = 20.

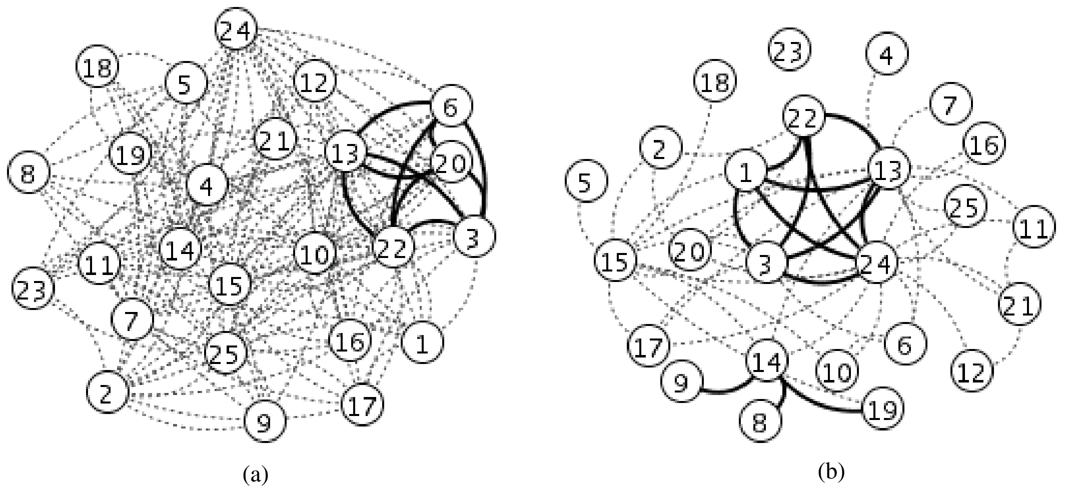

Figure 3. Partition results discarding the $N - 1$ betweenness rule; (a) threshold = 1; (b) threshold = 20.

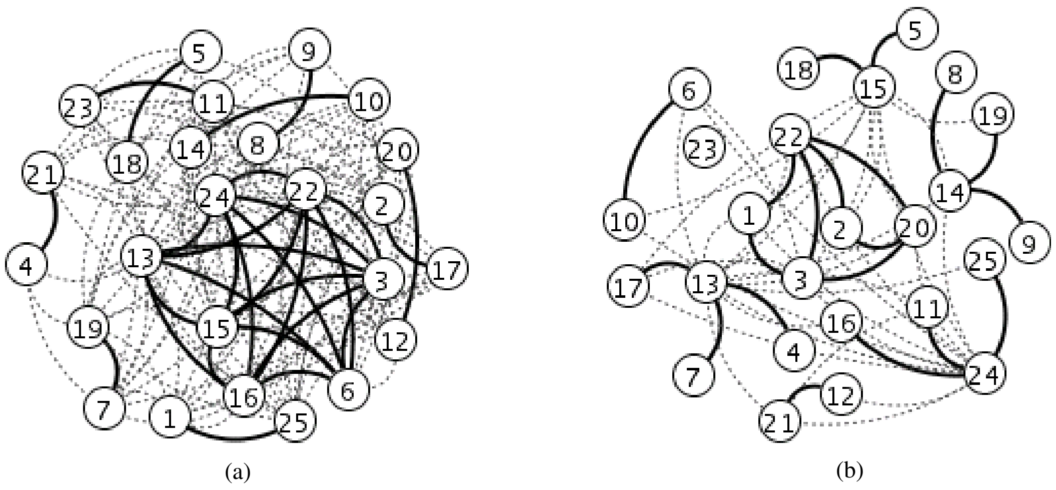

Figure 4. Partition results using new rules; (a) threshold = 1; (b) threshold = 20.

2.2.2. *Community detection of weighted social network*

The identification of the shortest path is the key to the community discovery. Previous works find the shortest path based on the assumption that each edge is equally long. In fact, the length of edges in a social network represents the strength of tie between nodes: the shorter the edge, the closer the relationship denoted by the edge, or the more important one node is to the other [7]. Differentiating edge length helps to find a more exact number of the shortest path, thereby leading to the discovery of more accurate community structures.

One way is to define the edge length according to the absolute importance of the edge; namely, how important the edge is for the whole network. Higher importance means shorter length. The length $Length_{ij}$ of the edge between vertex i and j can be measured as follows:

$$Length_{ij} = \frac{t}{Num_{ij}} \tag{1}$$

where t is the threshold used to construct the social network from information flow and Num_{ij} is the number of messages that have been passed between i and j.

Another way to define the edge length is based on the relative importance of the edge, i.e. how important the edge is for its vertices.

$$Length_{ij} = \frac{Num_{i\to all} \times Num_{j\to all}}{Num_{i\to j} \times Num_{j\to i}} \tag{2}$$

where $Num_{i\to all}$ is the number of messages that i has sent to others, $Num_{j\to all}$ is the number of messages that j has sent to others; $Num_{i\to j}$ is the number of messages that i has sent to j, and $Num_{j\to i}$ is the number of messages that j has sent to i.

Table I. Comparison between the three methods, where method A means unweighted, B means weighted by absolute importance and C means weighted by relative importance.

	Threshold				
Method	1	10	20	30	40
A (%)	24	72	60	72	56
B (%)	44	88	92	88	80
C (%)	60	72	92	84	76

Table I lists the partition precision when the three methods are adopted to calculate the edge length, respectively. For each method, the medium thresholds brings the highest precision. When the threshold equals 10, 20, 30 and 40, the absolute importance-based method brings the best partition results, followed by the relative importance-based method and then the unweighted method. When the threshold equals 1, the relative importance-based method brings the highest precision. In general, the weighted method produces better results than the unweighted method.

2.3. Network generator and community detector

The network generator takes messages collected as input, and related social network data stored in a Pajek .net file as output. The Pajek .net format is a strictly defined, text-only file format used by Pajek for reading and writing networks (Pajek was developed by the University of Ljubljana for the analysis of large networks [8]). The threshold and the concrete method to calculate the length of edge can be specified. Community detector inputs the Pajek .net file and displays the social network of the organization as an undirected graph. The vertices of a community are linked up by solid edges and different communities in the organization are linked up by dotted edges. The location of each vertex can be moved so as to view the network and communities clearly.

3. PERSONAL PROFILES AND PROFILE EXTRACTOR

Messages a member has read or written implicate preference. Therefore members' personal profiles can be obtained by tracing their daily use of all kinds of message. As Figure 5 illustrates, it consists of message filtering, relevant-message classification and personal profile computing.

First, irrelevant messages are filtered out. Relevant messages contain research information. Then, each relevant message is assigned to a specific topic by relevant-message classification. The topic represents the main meaning of a relevant message. Finally, a profile computing component determines personal profiles by a statistical evaluation of the results of relevant-message classification. The principle directing this statistical algorithm is that *the more relevant messages a member writes or reads about a topic, the more attention he/she pays to the topic*.

The user profile can also help a research team keep track of what everyone is doing and has done by browsing current user profiles and historical user profiles, and by tuning personal profiles to make accurate interest-description and document-delivery.

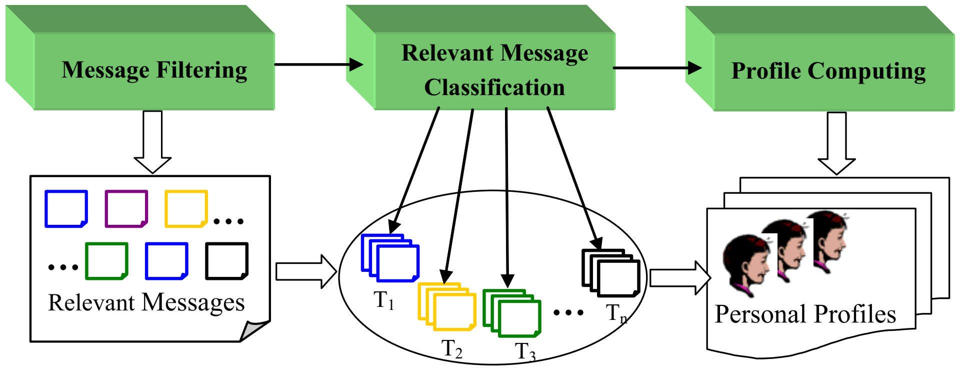

Figure 5. Discovery of personal profiles.

3.1. Message representation

Each message m is represented as a weighted term vector $\mathbf{m} = (m^{(1)}, m^{(2)}, \ldots)$ by standard TFIDF function.

$$m^{(i)} = TF(w_i, m) \log\left[\frac{|D|}{DF(w_i)}\right] \tag{3}$$

where the term frequency $TF(w_i, m)$ is the number of times word w_i occurs in message m, $|D|$ denotes the total number of messages in the training set and the $DF(w_i)$ is the number of messages containing the word w_i at least one time.

E-mail flow is a typical form of information flow. The text in the *subject* field and *body* field of E-mail is treated separately and identically in [9]: a word's one-time appearance in the *subject* and in the *body* is equal. Here, we also consider the text in the *subject* field and *body* field of a message separately but discriminatively.

Usually, *subject* is the outline of the *body*'s content, so words in the *subject* field are more descriptive and discriminative, unlike the words in the *body* field. That is, they are more important for the classification. Here, words in the *subject* field are assigned larger weights. When calculating the weight of word w_i, there is no change for $DF(w_i)$. For $TF(w_i, m)$, a one-time appearance in the *subject* field is equal to t appearances in the *body* field. The enhancement of $TF(w_i, m)$ strengthens the importance of w_i, while no change for $DF(w_i)$ ensures that the tuning of w_i will not be weakened. The increase of a words' weight reinforces their discriminative ability in turn [10].

Because a message such as E-mail is typically short, and the message *body*, *subject* and *attachment* normally express a common theme, the text attachment reserved is treated as a part of message *body*.

3.2. Message filtering

Models of relevant messages and irrelevant messages are represented as prototype vectors $\mathbf{m}_1$ and $\mathbf{m}_2$, respectively. Both the positive and the negative examples are taken into account.

$$\mathbf{m}_1 = \alpha \frac{1}{|M_1|} \sum_{\mathbf{m} \in M_1} \frac{\mathbf{m}}{|\mathbf{m}|} - \beta \frac{1}{|M_2|} \sum_{\mathbf{m} \in M_2} \frac{\mathbf{m}}{\mathbf{m}} \tag{4}$$

$$\mathbf{m}_2 = \alpha \frac{1}{|M_2|} \sum_{\mathbf{m} \in M_2} \frac{\mathbf{m}}{|\mathbf{m}|} - \beta \frac{1}{|M_1|} \sum_{\mathbf{m} \in M_1} \frac{\mathbf{m}}{\mathbf{m}} \tag{5}$$

Where α and β are parameters for adjusting the relative impact of positive and negative examples. As recommended in [11], α and β are set to 16 and 4. M_1 and M_2 are the training messages of relevant messages and irrelevant messages, respectively; $|M_1|$ and $|M_2|$ are the number of messages in M_1 and M_2. The vector representation of message m is $\mathbf{m}$ and $|\mathbf{m}|$ denotes the Euclidean length of $\mathbf{m}$. Here, M_1 is positive examples for relevant messages and negative examples for irrelevant messages. M_2 is positive examples for irrelevant messages and negative examples for relevant messages. When transforming a training message into a feature vector, the words that occur with similar proportion of times in both M_1 and M_2 are deleted. Messages m is relevant if cosine similarity between $\mathbf{m}$ and $\mathbf{m}_1$ is higher than that between $\mathbf{m}$ and $\mathbf{m}_2$, otherwise, it is irrelevant.

3.3. Message classification and profile computing

The relevant-message classification specifies a topic for each relevant message by general document classification. First, a prototype vector for each topic in the document space is built. Documents stored in the topic level are utilized as training documents for corresponding topics. Then, for each relevant message within certain common interest community, a prototype vector that gives the largest cosine of the message vector and the prototype vector itself is found. This topic is a good representation for the relevant message. Here, only the prototype vectors for the topics that belong to the area corresponding to the community profile are considered.

The more relevant messages a member reads or writes about a topic, the more the member is interested in that topic. The personal profile of user U_j takes the form as a finite set of $\langle topic_i, energy_{ij} \rangle$, where $energy_{ij}$ denotes the importance degree of $topic_i$ in the personal profile of U_j, $energy_{ij}$ is defined as

$$energy_{ij} = \frac{\alpha \sum_{(m \in from_j) \cap (m \in T_i)} 2^{-age(m)/hl} Sim(m, t) + \beta \sum_{(m \in to_j) \cap (m \in T_i)} 2^{-age(m)/hl} Sim(m, t)}{\alpha \sum_{(m \in from_j)} 2^{-age(m)/hl} Sim(m, t) + \beta \sum_{(m \in to_j)} 2^{-age(m)/hl} Sim(m, t)} \tag{6}$$

where $From_j$ is the relevant messages U_j has sent to others in the common interest community and To_j is the relevant messages U_j has received from others in the common interest community. The set of relevant messages associated with $topic_i$ is denoted by T_i and $Sim(m, t)$ is the cosine similarity between m and the topic it belongs to. The parameters α and β adjust the relative impact of relevant messages flowing from and to U_j separately. Stronger impact is specified to the relevant messages flowing from a member by a larger α and a smaller β.

Because user interests often change, it is important to adjust the user profile incrementally [12]. A time factor $2^{-age(m)/hl}$ is introduced to adjust the contribution of the relevant message for the personal profile according to its age $age(m)$, which makes the descriptive ability of the relevant message decay with time. The algebraic difference between the current date and the date when m was sent is given by $age(m)$. The half-life span hl is set at 30 on the assumption that the effect of relevant messages on a topic reduces by $\frac{1}{2}$ in one month [13]. The personal profile adapts to changes in members' interests with the accumulation of messages and the passing of time. Here, no techniques such as relevance feedback, users' registration and users' ratings are employed.

4. IDENTIFYING AND RANKING HELPER CANDIDATES

When a researcher encounters a hard problem, an efficient strategy is to ask for help within the organization. The person they need should be a member who is skilled in the area in which they are engaging. In order to recommend the right helper, the following two tasks must be accomplished.

(1) Find the persons who may be helpful within the organization. They should be members who are interested in the problems; namely, they should be the members who share interests with the current user. We can easily find them out from the organization by referring to the organization's community structures because members in the same common interest community share a common community profile. Therefore, all members belonging to the same common interest community (including the current user) constitute helper candidates.

(2) Select the persons from the helper candidates who are skilful helpers. The solution needs relationship analysis among the helper candidates. All the candidates are sorted by rank whose value indicates their skill level. Here, the ranks are computed by the weighted PageRank algorithm on the helper candidate graph. The helper candidate graph is composed of helper candidates and the relationships among them. When one asks for help, the candidates with higher ranks will be recommended as helpers.

4.1. PageRank and weighted PageRank

The PageRank is computed by the following formula:

$$PR(u) = d \sum_{v \in B(u)} \frac{PR(v)}{N_v} + (1 - d) \tag{7}$$

where u represents a Web page. $B(u)$ is the set of pages that point to u. $PR(u)$ and $PR(v)$ are rank scores of page u and v, respectively. N_v denotes the number of outgoing links of page v, d is a dampening factor that is usually set to 0.85. PageRank has applications in search, browsing and traffic estimation [14,15]. It was used for expertise identification and ranking [16,17].

The original PageRank algorithm divides the rank score of a Web page or an expert evenly over to its successors. The weighted PageRank algorithm assigns larger rank values to more important pages, because the more popular the Web pages are, the more linkages are linked to them [18]. Here, we measure the capability of helper candidates by the weighted PageRank algorithm by the fact that the tie strength between helper candidates is different. The main idea is that the closer the relationship between v and u, the more rank values should be transferred from v to u.

$$CR(u) = d \sum_{v \in B(u)} W_{v \to u} \times CR(v) + (1 - d) \tag{8}$$

where, v and u are vertices in the helper candidate graph, $W_{v \to u}$ is the weight of the edge, which starts from candidate v and ends at u in the helper candidate graph. In order to distinguish the rank score of Web page, we use $CR(u)$ to represent the rank score of the helper candidate u.

4.2. Helper candidates ranking

The ranking process consists of four steps: community recovery, direction transform, weight assignment and rank computing. The ranking process will be introduced with the social network example in Figure 6(a), where the filled vertex is the person who is asking for help.

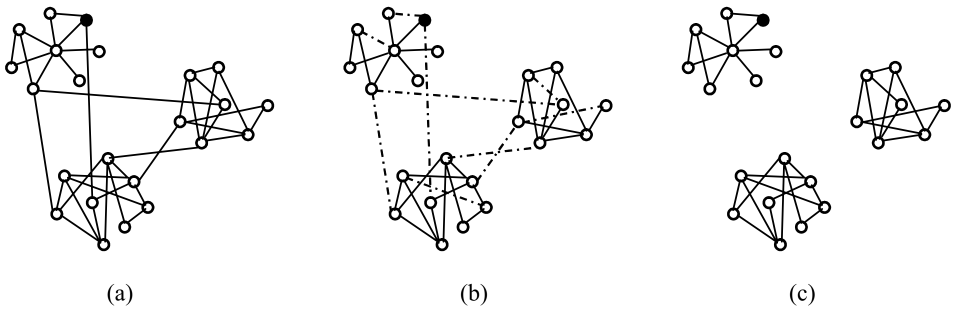

Figure 6. Community recovery process: (a) original social network; (b) detecting results; (c) community recovery results.

4.2.1. Community recovery

In order to find the common interest communities within the organization, we compute edge betweenness and remove the edge with higher betweenness repeatedly (as introduced in Section 2.2).

Figure 6(b) illustrates the common-interest communities identified in the social network shown in Figure 6(a), where the dotted lines are the edges that have been removed. From Figure 6(b) we can see that when the edges between communities are removed, a part of the edges within communities are deleted as well. The removal of the inner edges may affect the ranking precision as our method is based on the helper candidate graph to which the asker belongs.

The first step towards helper candidates, ranking is community recovery, whose responsibility is to add the inner edges that have been removed back to common-interest communities. Each common-interest community output by community recovery is the helper candidate graph for the members in it. The recovery results of Figure 6(b) are given in Figure 6(c).

4.2.2. Direction transform

The PageRank computes each Web page's rank score based on the graph of the Web pages [14]. The social network we constructed is an undirected graph, as is the helper candidate graph. In order to rank helper candidates by the PageRank algorithm, each tie between candidates in the helper candidate graph must have a direction. This step specifies a direction for each edge in the helper candidate graph according to the characteristics of information flow. Namely, the helper candidate graph will be converted from an undirected graph to a directed graph in this step.

Several methods can be used to determine an edge's direction.

- The first is based on the message quantity exchanged between two linked candidates: if candidate v has sent more messages to candidate u than that u has sent to v, then the edge between v and u directs to v or else it directs to u.
- The second and third are degree-based. In the social network, vertex degree is often used to measure the power of vertices: a higher degree indicates more power [19]. When the degree is

considered, there are two conditions for the vertices in the helper candidate graph. One is that each edge on a vertex contributes one to its degree. The other is that each message sent out by a candidate to others in the helper candidate graph contributes one to his degree. For convenience purposes, we call the degree under the first condition *edge degree*, the degree under the second condition *message degree*. In fact, the message degree of a vertex is the sum of messages that it has sent to others in the helper candidate graph. No matter what the condition, our method lets each edge go from the vertex with the lower degree to the one with the higher degree.

- The fourth is based on the relative impact of the edge on the two vertices that are linked by it. The edge e_{vu} starts from u and ends at v if the impact of e_{vu} on vertex u is larger than that on vertex v, or e_{vu} starts from v and ends at u. The impact of e_{vu} on v, I_{vu-v}, is calculated as

$$I_{vu-v} = \frac{Num_{v\to u}}{Num_{v\to all}} \tag{9}$$

where $Num_{v\to u}$ is the number of messages that v has sent to u, $Num_{v\to all}$ is the number of messages that v has sent to others in the helper candidate graph.

Figure 7(a) is the undirected helper candidate graph which is obtained from Figure 6 for the person who is asking for help through community recovery. We can see that the member C who needs help is also a part of his helper candidate graph. Figure 7(b) is the directed representation of Figure 7(a) after the direction transformation. The original PageRank algorithm can now be adopted to compute each candidate's rank if the tie strength between them is not taken into account.

4.2.3. *Weight assignment*

The original PageRank algorithm evenly divides a helper candidate's rank over to his successors, ignoring the tie strength between helper candidates [14]. A different edge length has been used in Section 2.2.2 to represent the tie strength between two vertices. It is reasonable to believe that a different tie strength should lead to a different distribution of the candidate's rank. In order to make the rank distribution more accurate, the weight of the edge that directs the rank distribution should be assigned.

Two kinds of method are used to calculate edge weight for the candidate graph. The first only considers the successors of the edge's origin vertex; the second considers both the successors of the edge's destination vertex and the predecessors of the edge's origin vertex.

The first method in the first class defines the edge weight as

$$W_{v\to u} = \frac{Num_{v\to u} + Num_{u\to v}}{\sum_{t\in F(v)}(Num_{v\to t} + Num_{t\to v})} \tag{10}$$

where, $W_{v\to u}$ is the weight of the link from v to u, $F(v)$ is the set of candidates to which v directs, namely, successors of v. From Equation (10), we can see that this method is based on the message exchanged between the two vertices that are linked up by the edge.

The message degree is the basis of the second method. The denominator in the right of this equation is the message degree of v.

$$W_{v\to u} = \frac{Num_{v\to u}}{\sum_{t\in F(v)} Num_{v\to t}} \tag{11}$$

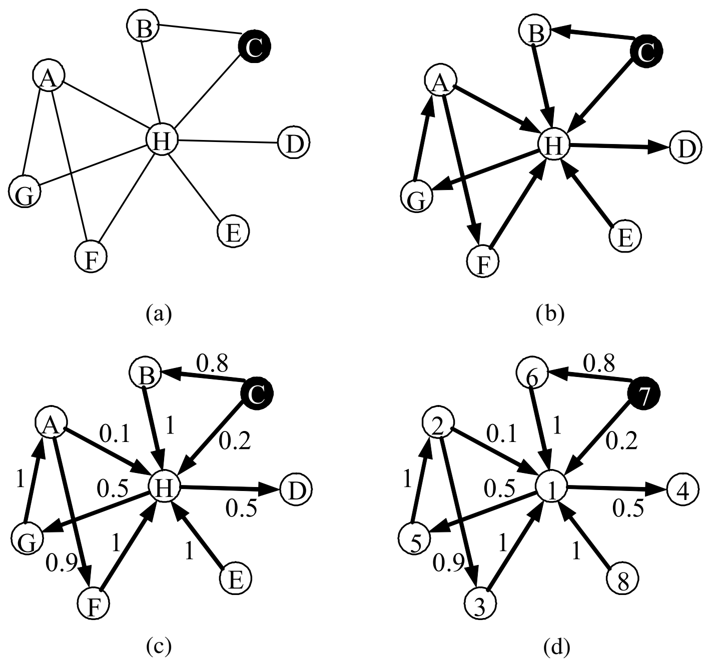

Figure 7. Direction transform, weight assignment and rank computing: (a) undirected helper candidate graph; (b) directed helper candidate graph; (c) weighted, directed helper candidate graph; (d) results of rank computing.

The last method in this class is based on the relative importance of the edge. $W_{v\rightarrow u}$ is defined as

$$W_{v\rightarrow u} = \frac{Num_{v\rightarrow u} \times Num_{u\rightarrow v}}{\sum_{t\in F(v)} (Num_{v\rightarrow t} \times Num_{t\rightarrow v})} \tag{12}$$

There are three methods in the second class also. The first method computes edge weight according to the out-links of the edge's destination and the out-links of the predecessors of its origin. First, each edge's out-link proportion is calculated by

$$Out_{v\rightarrow u} = \frac{\sum_{w\in F(u)} Num_{u\rightarrow w}}{\sum_{p\in B(v)} \sum_{q\in F(p)} Num_{p\rightarrow q}} \tag{13}$$

where, $Out_{v\rightarrow u}$ is the out-link proportion of the edge from v to u, $B(v)$ is the predecessors of v. Then $W_{v\rightarrow u}$ is defined as

$$W_{v\rightarrow u} = \frac{Out_{v\rightarrow u}}{\sum_{t\in F(v)} Out_{v\rightarrow t}} \tag{14}$$

The second method in this class is based on the in-links of the edge's destination and the in-links of the predecessors of its origin. Similarly, each edge's in-link proportion is calculated first.

$$In_{v\to u} = \frac{\sum_{w\in B(u)} Num_{w\to u}}{\sum_{p\in B(v)} \sum_{q\in B(p)} Num_{q\to p}} \tag{15}$$

$In_{v\to u}$ is the in-link proportion of the edge from v to u. Then $W_{v\to u}$ is obtained by

$$W_{v\to u} = \frac{In_{v\to u}}{\sum_{t\in F(v)} In_{v\to t}} \tag{16}$$

The next method decides an edge's weight by the combination of the edge's out-link proportion and in-link proportion.

$$W_{v\to u} = \frac{Out_{v\to u} \times In_{v\to u}}{\sum_{t\in F(v)} (Out_{v\to t} \times In_{v\to t})} \tag{17}$$

The design idea of the second class of methods comes from [18]. We develop it by adding characteristics of information flow and renormalizing the weight of all edges that start from the same vertex to sum to one.

If we replace the $W_{v\to u}$ in Equation (8) by the $1/N_v$ in Equation (7), we will obtain the original PageRank algorithm. In order to compare with the original PageRank algorithm we call this method the unweighted method. Together with the unweighted condition, there are seven ways to compute the edge weight. As shown in Figure 7(c), after this step, each link between helper candidates has a weight value.

4.2.4. *Convergence properties and rank computing*

As with the PageRank algorithm, the weighted PageRank algorithm can also be regarded as a Markov chain [20]. Its transition matrix $\mathbf{M}'$ is:

$$\mathbf{M}' = d \times \mathbf{M} + (1-d) \times \mathbf{E} \tag{18}$$

where $\mathbf{M}$ is a square matrix whose element m_{vu} is $W_{v\to u}$, N is the number of the vertices in the helper candidate graph and $\mathbf{E} = [1/N]_{N\times N}$.

The seven methods given here to calculate $W_{v\to u}$ insure $\sum_{u=1}^{N} m_{vu} = 1$ for all v, so $\mathbf{M}$ is a stochastic matrix [20]. The combination of the stochastic matrix $\mathbf{M}$ and the stochastic perturbation matrix $\mathbf{E}$ insures that $\mathbf{M}'$ is both stochastic and primitive. Therefore, the weighted PageRank algorithm will converge [21].

The last step of the helper candidates' ranking is to determine each candidate's recommendation order according to their capability. This is achieved through the weighted PageRank algorithm on the helper candidate graph, which is obtained as in the last section. Here, we specify an equal initial rank score for each helper candidate as their initial values will not affect the final values [14].

After rank computing, candidate labels in Figure 7(c) are replaced by their recommendation orders in Figure 7(d). The recommender can now recommend helpers by order. The final rank score value for each helper candidate in Figure 7(d) can be found in Table II.

Table II. Rank computing results.

Candidate label	Score value	Rank order	Candidate label	Score value	Rank order
A	0.159 158	2	E	0.034 351	8
B	0.057 709	6	F	0.156 107	3
C	0.034 351	7	G	0.146 832	5
D	0.146 832	4	H	0.264 661	1

4.3. Accuracy evaluation

In order to evaluate our method, which ranks the candidates, we adopt the recall-based analysis method used by [17]. The recall on top n is defined as the fraction of top n helper candidates of the correct (ground-truth) ranking ρ that are contained in the top n helper candidates of the ranking γ of our method.

The raw data of our experiments are 98 138 E-mails, collected from a research organization of 312 members from 5 September 2004 to 16 March 2006. First, the social network that describes the member relationship within the organization is built from these E-mails. When the social network is constructed, the threshold 10 is used. Then, different common interest communities in the organization are found out by the community detection method that was introduced in Section 2.2. There are nine research communities that are made up of 52, 25, 61, 54, 19, 15, 31, 22 and 33 members, respectively. Our next experiments are conducted on the helper candidate graph which is obtained through community detection and community recovery from the research community and which consists of 25 members.

In order to obtain the ground-truth ranking ρ, we ask each member in that research community to sort the community members according to their capabilities. As a result, we obtain 25 member lists. First, we compute the total number for each member, which is the sum of the order number in each list. Then, we rank the members in descending order of their total number. The result will be used as the ground-truth ranking ρ for evaluations.

The weighted PageRank algorithm is a Markov process that is characterized by its transition matrix. The direction and weight of the edges in the helper candidate graph determine the transition matrix. Therefore the edge direction assignment approach together with the edge weight computing method determines the final rank result. Approaches have been put forward in Section 4.2 to specify direction and calculate weights for the edges in the helper candidate graph. Their combination brings 28 patterns to characterize the helper candidate graph. Figure 8 depicts the helper candidate graph that has been directed by message quantity and weighted by message degree.

First, we look at the effect of the methods that specify edge direction for the helper candidate graph. Figures 9–15 illustrate the ranking results of the helper candidate graphs that are weighted by a certain concrete way. The vertical axis shows recall, abscissa is top size. Here, the top size spans from 1 to 25. There are four recall curves in each figure. Each curve describes the ranking results of the candidate graph, which is directed by a certain method and weighted by way of corresponding to the current figure.

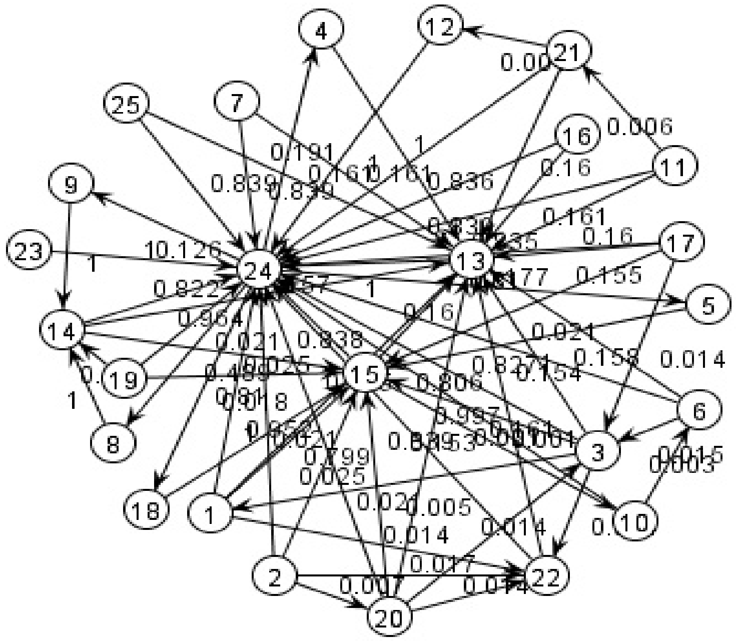

Figure 8. The helper candidate graph directed by message quantity and weighted by message degree.

These figures show that, under most conditions, the helper candidate graphs perform best when they are directed by message degree and the helper candidate graphs always perform worst when they are directed by relative impact. In order to compare these four methods by one figure, we give the definition of average recall. Here, the average recall under a certain pattern is defined by

$$\bar{r} = \sum_{n=1}^{n_{\max}} 2^{(n-1)/hl} \times r_n \Big/ \sum_{n=1}^{n_{\max}} 2^{(n-1)/hl} \tag{19}$$

where n is the top size, r_n is the recall under the current pattern when the top size equals n. This definition integrates the recall at a certain pattern when the top size n spans from 1 to $n_{\max}$ to obtain the average recall, $\bar{r}$, to evaluate the performance of the current pattern. Because the recall is more important when the top size is small than the recall when the top size is large when the average recall is calculated, the impact of the recall is adjusted by $2^{-(n-1)/\mathrm{hl}}$, which makes the influence of the recall decrease with the increase of n. Here, $hl = \lceil n_{\max}/2 \rceil$ and $n_{\max} = 25$ is the maximum value of n.

For each pattern we calculate its average recall and illustrate the results in Figure 16, where the vertical axis shows average recall. There are seven points in the abscissa axis, each denoting a method for computing the edge weight. Unweighted, weighted by message exchanged, weighted by message degree, weighted by relative importance, weighted by out-links, weighted by in-links and weighted by out-links and in-links are represented by w_un, w_e, w_m, w_r, w_out, w_in and w_l, respectively.

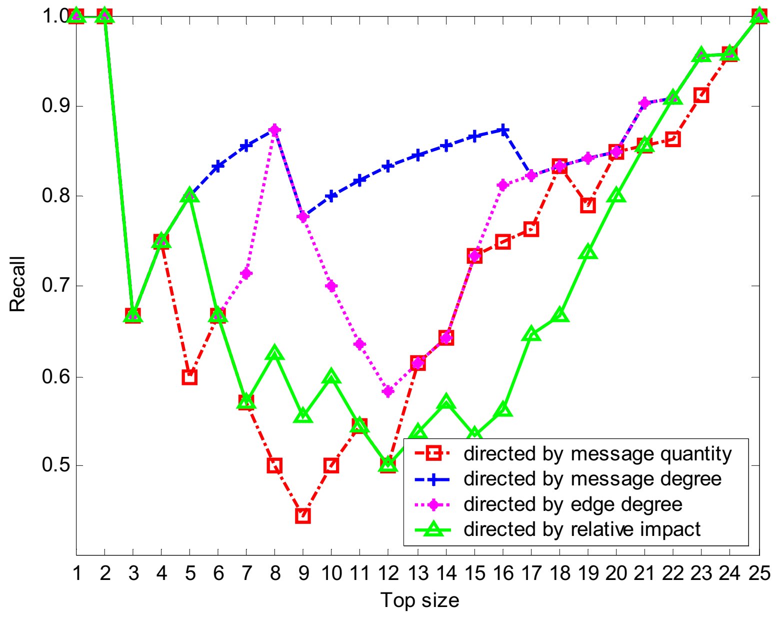

Figure 9. Rank results of the unweighted helper candidate graphs.

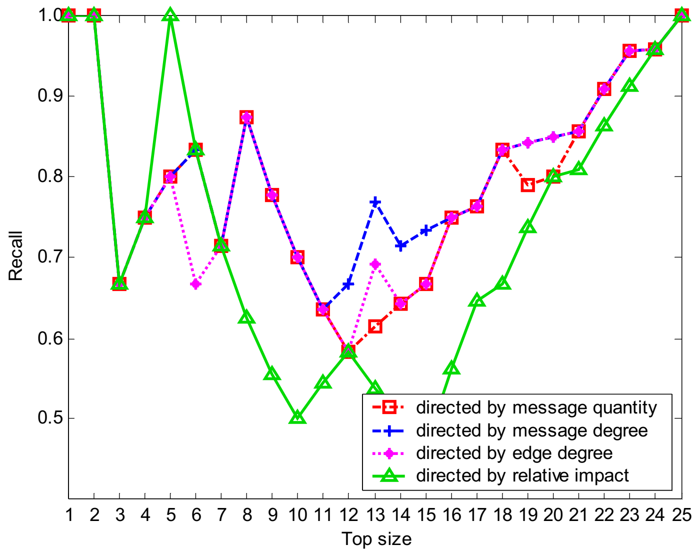

Figure 10. Rank results of the helper candidate graphs that are weighted by message exchange.

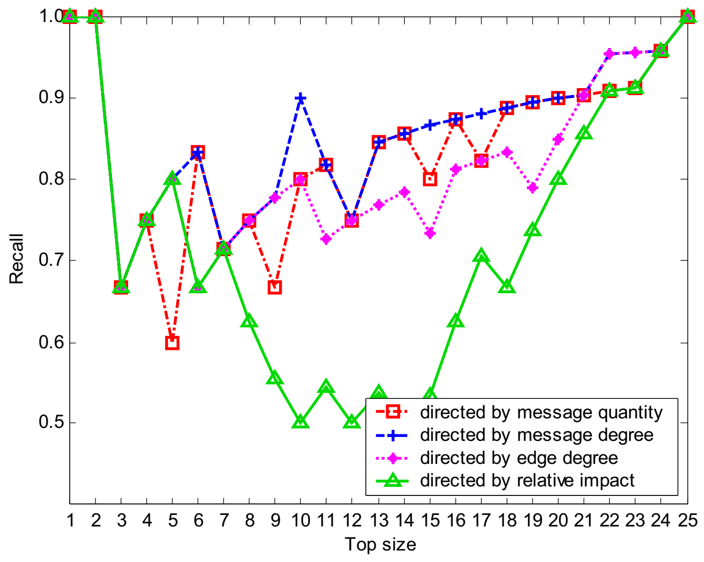

Figure 11. Rank results of the helper candidate graphs that are weighted by message degree.

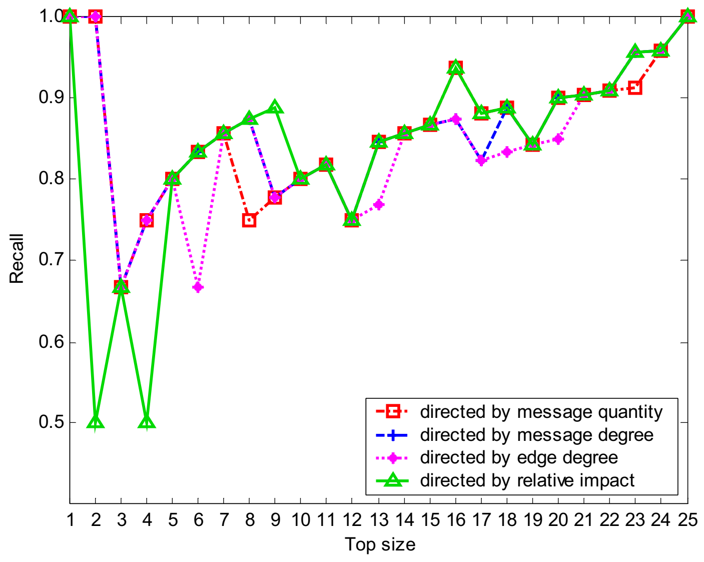

Figure 12. Rank results of the helper candidate graphs that are weighted by relative importance.

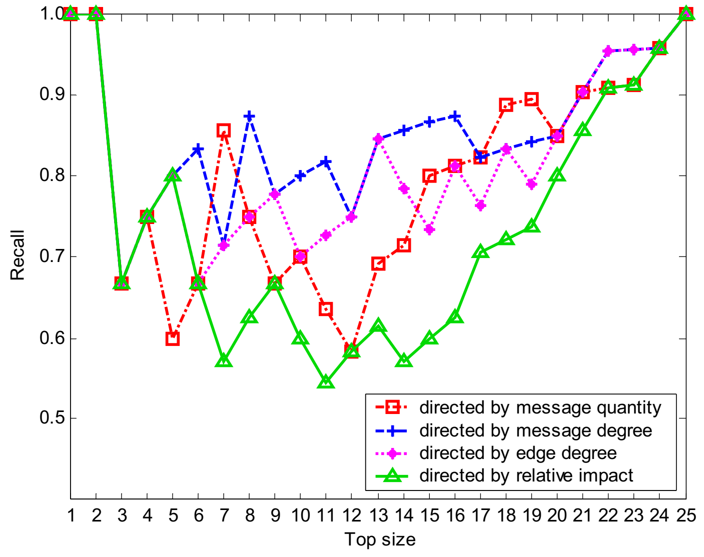

Figure 13. Rank results of the helper candidate graphs that are weighted by out-links.

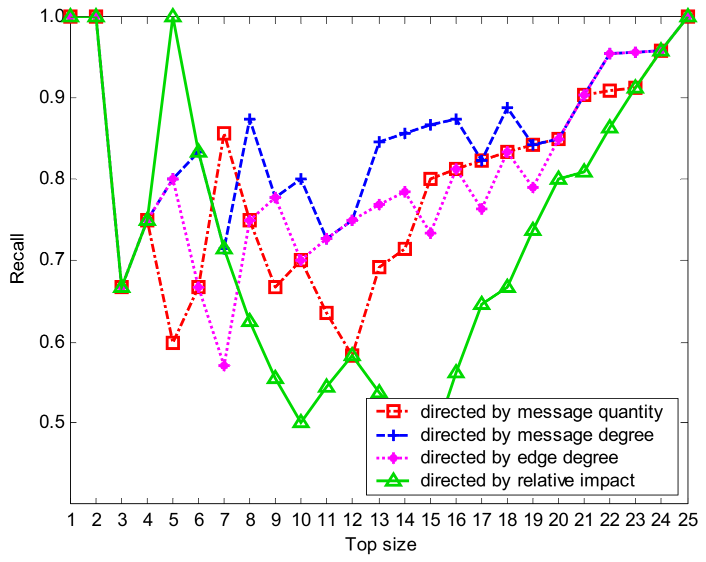

Figure 14. Rank results of the helper candidate graphs that are weighted by in-links.

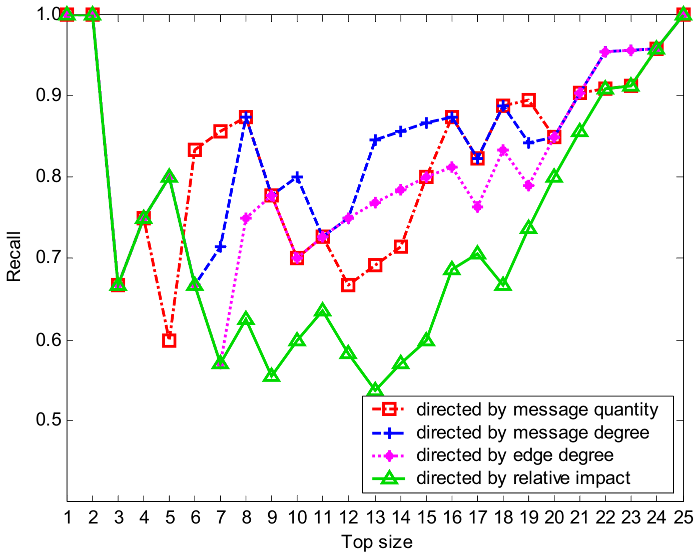

Figure 15. Rank results of the helper candidate graphs weighted by out-links and in-links.

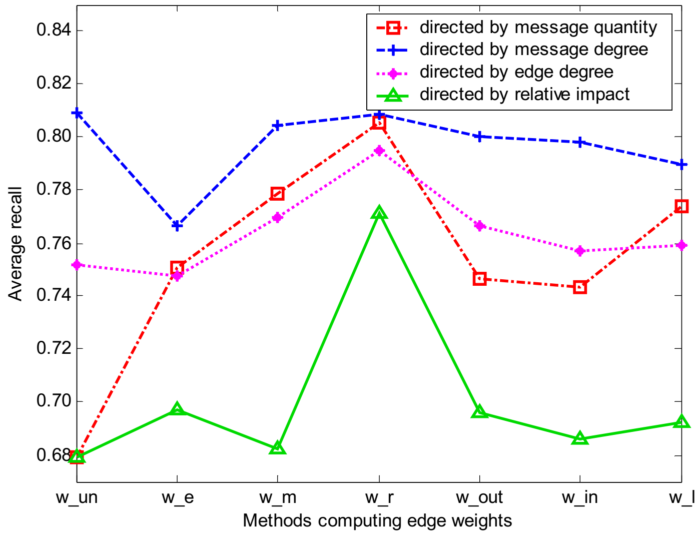

Figure 16. Comparison between methods that specify direction for edges.

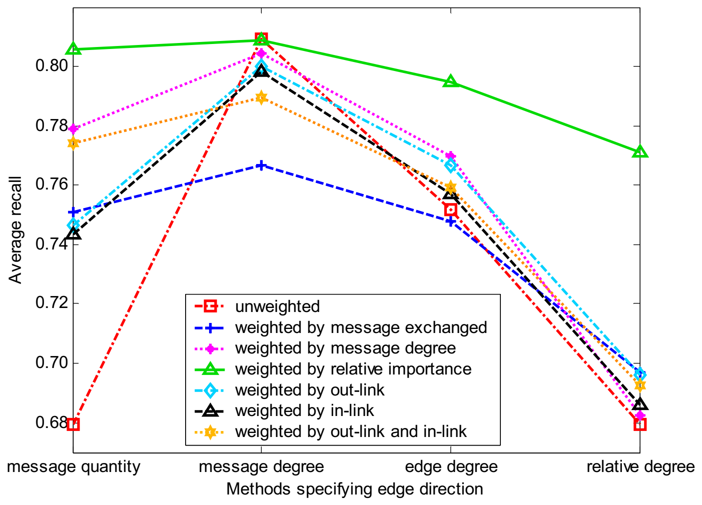

Figure 17. Output when different methods are adopted to calculate edge weight.

The results of the graphs, which are directed by the same method, are linked up as the average recall curve for that method. From this figure, we can conclude that the helper candidate graphs that are directed by message degree always perform best and the helper candidate graphs that are directed by relative impact always perform worst. This agrees with the former figures.

Figure 17 compares methods for calculating the edge weight. The abscissa axis is the method for determining the direction for each edge; the vertical axis is the average recall. From Figure 17, we can see the method for calculating edge weight according to the relative importance of the edge performs best, while, contrary to our expectations, the candidate graphs that are unweighted do not always bring the lowest average recall. Although the unweighted helper candidate graphs perform fairly well when they are directed by message degree, the unweighted helper candidate graphs directed by other methods produce some of the worst performances. From Figure 17 we also find the methods that consider both the successors of the edge's destination and the predecessors of the edge's origin do not perform better than the methods that only consider the successors of the edge's origin. Each point in Figure 17 corresponds to a recall curve in Figures 18–21. Figure 17 contains four groups of points with the same horizontal value. Figures 18–21 describe the four groups in detail.

5. RELATED WORK AND DISCUSSION

Learning and constructing user profiles without any human intervention is an important feature of our method. It does not depend on other techniques such as relevance feedback, users' register and users' ratings.

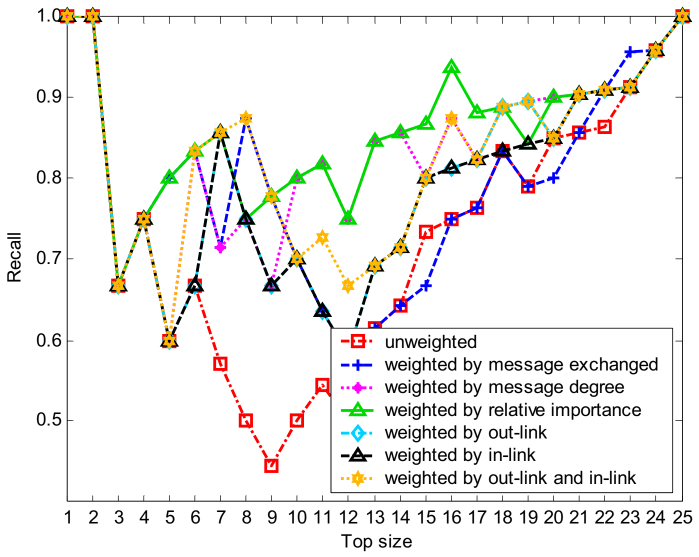

Figure 18. Ranking results of the helper candidate graphs directed by message quantity.

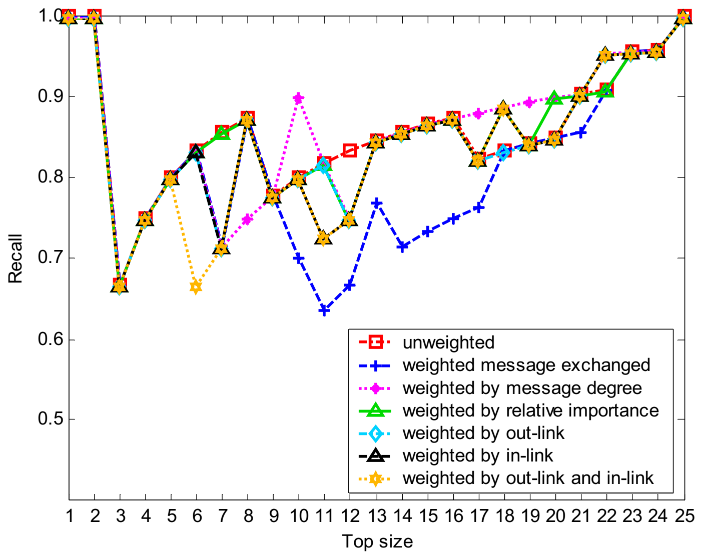

Figure 19. Ranking results of the helper candidate graphs directed by message degree.

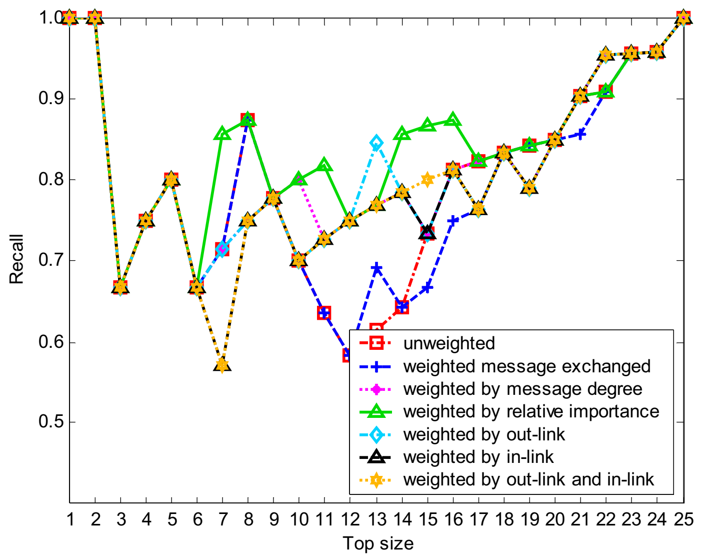

Figure 20. Ranking results of the helper candidate graphs directed by edge degree.

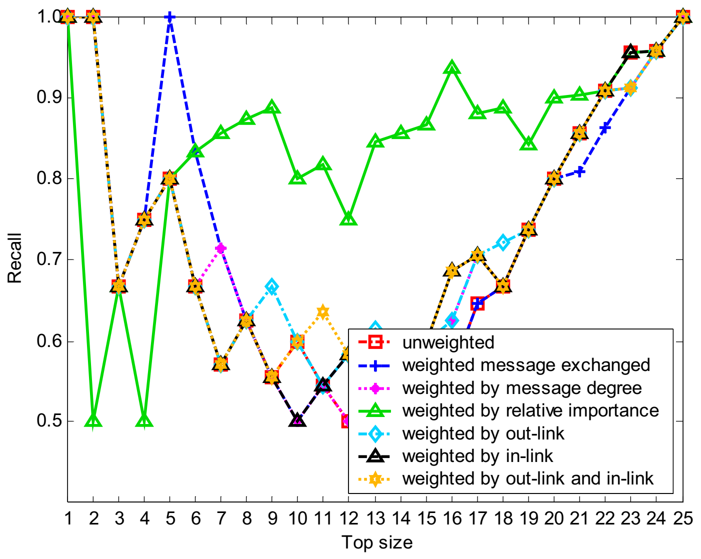

Figure 21. Rank results of the helper candidate graphs directed by relative importance.

The recommended system proposed in [22] asks users to evaluate a set of documents. Then a vector of keywords is extracted from these documents according to the evaluation results. The keyword vector is used to identify users. Most recommending systems learn interests of users via relevance feedback and keep track of them daily using a search engine. It may satisfy a user's immediate information interests but usually is not sufficient for persistent interests because users are relatively poor at using these systems [23]. E-mail and instant messaging have recently become some of the most popular communication trends. Information flows such as e-mail flow, instant message flow and flow within and between message boards supply a rich and persistent resource for learning user profiles.

Usually, recommendations are given by listing the pages or documents when a user begins a new session of using a system [12,24]. These occur only when a user chooses to use the system. It is an occasional and customizing behaviour, not a persistent behaviour. Our method pushes relevant documents by E-mail for users at regular intervals, which ensures that users can obtain relevant documents in time and consistently. Users do see the documents without any separate action because E-mail is a 'push' delivery mechanism in contrast to the 'pull' mechanisms such as Web searching. It makes our method low cost for organizations to adopt.

Awareness means 'understanding of the activities of others, which provides a context for your own activity' [25]. In [26], each group member has to write a 'today' message at the end of the day explaining what they did that day for group awareness. This type of awareness is immediate and short term (one day). The awareness in our work is long term. Through user profiles, users can know what others have done, are doing and will do in a longer period. Members can quickly and accurately locate themselves in the whole organization by browsing community structures or user profiles. User profiles also tell us what we can learn, from whom and in which community.

A number of automatic expert finding methods were proposed. A user's technical document is used to build the expertise index [27,28]. The expertise index is built by mining relationships between users and various documents by text analysis [29]. Kautz *et al.* believe the best way to find an expert is by 'referral chaining' [30,31]. Textual analysis is used by most of the previous systems as the basis for expert/expertise identification. Unlike previous methods that are based on textual analysis, our approach is based on information flow, identifies persons who may be helpful for the user by common interest community detecting by a social network analysis method and ranks them according to capability by link analysis between them. It is similar to the method of using Web logs to compute the relevance of a Web user to a given query through link analysis between Web pages and Web users [32] but the information flow used in our approach contains richer semantics.

6. CONCLUSION

Scientific documents and researchers are inseparable parts of scientific research. The proposed approach combines the two parts and explores the usage of the information flow network generated during research. It has the following features:

(1) automatically discovering interest and helper;
(2) capturing interest in information flow between researchers;
(3) adapting profile to reflect the change of interest with time;
(4) actively and persistently pushing documents in time without additional effort;

(5) graphically displaying community structures and user profiles;
(6) automatically recommending helpers who can directly help or recommend appropriate scientific documents.

Ongoing works include developing a semantic community discovery approach and to incorporate the knowledge flow mechanism [5] into our approach to reflect the dynamic knowledge exchange between researchers in the cross-disciplinary e-Science Knowledge Grid environment IMAGINE-1.

REFERENCES

1. Zhuge H. *The Knowledge Grid*. World Scientific: Singapore, 2004.
2. Girvan M, Newman M. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the United States of America* 2002; **99**:7821–7826.
3. Tyler J, Wilkinson D, Huberma B. Email as spectroscopy: Automated discovery of community structure within organizations. *Proceedings of the 1st International Conference on Communities and Technologies*, Amsterdam, 2003. Kluwer: Deventer, 2003; 81–96.
4. Wilkinson D, Huberman B. A method for finding communities of related genes. *Proceedings of the National Academy of Sciences of the United States of America* 2004; **101**:5241–5248.
5. Zhuge H. Discovery of Knowledge flow in science. *Communications of the ACM* 2005; **49**(5):101–107.
6. Freeman L. A set of measures of centrality based upon betweenness. *Sociometry* 1977; **40**(1):35–41.
7. Yee J, Mills R, Peterson G, Bartczak S. Automatic generation of social network data from electronic-mail communications. *Proceedings of the 10th ICCRTS*, McLean, VA, 2005; 1.
8. Batagelj V, Mrvar AP. Analysis and visualization of large networks. *Graph Drawing Software*, Jünger M, Mutzel P. (eds.). Springer: Berlin, 2003; 77–103.
9. Yang J, Park S. Email categorization using fast machine learning algorithms. *Proceedings of the 5th International Conference on Discovery Science* (*Lecture Notes in Computer Science*, vol. 2534). Springer: Berlin, 2002; 316–323.
10. Ye Y, Ma F, Rong H, Huang J. Enhanced Email classification based on feature space enriching. *Proceedings of the 9th International Conference on Applications of Natural Languages to Information Systems (NLDB 2004)*, Salford, U.K. (*Lecture Notes in Computer Science*, vol. 3136). Springer: Berlin, 2004.
11. Buckley C, Salton G, Allan J. The effect of adding relevance information in a relevance feedback environment. *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Dublin, 1994. Springer: Berlin, 1994; 292–300.
12. Chen C, Chen M, Sun Y. A Web document personalization user model and system. *Proceedings of the Workshop on Machine Learning, Information Retrieval and User Modeling*, Sonthofen, Germany, 2001 (in German). Available at: http://www.cs.rutgers.edu/ml4um/mirrors/mlirum-2001/papers/ [May 2006].
13. Sugiyama K, Hatano K, Yoshikawa M. Adaptive Web search based on user profile constructed without any effort from users. *Proceedings of the 13th Conference on World Wide Web (AAMAS 2004)*, New York, 2004. IEEE Computer Society Press: Los Alamitos, CA, 2004; 675–684.
14. Page L, Brin S, Motwani R, Winograd T. The PageRank citation ranking: Bringing order to the Web. *Technical Report SIDL-WP-1999-0120*, Stanford Digital Libraries, Stanford, CA, 1999.
15. Ridings C, Shishigin M. Pagerank uncovered. *Technical Report*, 2002. Available at: http://www.voelspriet2.nl/PageRank.pdf [May 2006].
16. Campbell C, Maglio P, Cozzi A, Dom B. Expertise identification using email communications. *Proceedings of the 2003 ACM CIKM International Conference on Information and Knowledge Management*, New Orleans, LA, 2003. ACM Press: New York, 2003; 528–531.
17. Dom B, Eiron I, Cozzi A, Zhang Y. Graph-based ranking algorithms for E-mail expertise analysis. *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, San Diego, CA, 2003. ACM Press: New York, 2003; 42–48.
18. Xing W, Ghorbani A. Weighted PageRank algorithm. *Proceedings of the 2nd Annual Conference on Communication Networks and Services Research (CNSR 2004)*, Fredericton, Canada. IEEE Computer Society Press: Los Alamitos, CA, 2004; 305–314.
19. Hannemann R, Riddle M. *Introduction to Social Network Methods*. Online textbook, 2005. http://faculty.ucr.edu/∼hanneman/nettext/networks.zip [May 2006].
20. Haveliwala T, Kamvar S. The second eigenvalue of the Google matrix. *Technical Report*, 2003. Available at: http://dbpubs.stanford.edu/pub/2003-20.

21. Langville A, Meyer C. Deeper inside PageRank. *Internet Mathematics* 2005; **1**(3):335–380.
22. Vel O, Nesbitt S. A collaborative filtering agent system for dynamic virtual communities on the Web. *Proceedings of the Conference on Automated Learning and Discovery*, Pittsburgh, PA, 1998.
23. Somlo G, Howe A. QueryTracker: An agent for tracking persistent information needs. *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multi-Agent Systems*, New York, 2004. IEEE Computer Society Press: Los Alamitos, CA, 2004; 488–495.
24. Bollacker K, Lawrence S, Giles C. A system for automatic personalized tracking of scientific literature on the Web. *Proceedings of the 4th ACM Conference on Digital Libraries*, Berkeley, CA, 1999. ACM Press: New York, 1999; 105–113.
25. Dourish P, Bellotti V. Awareness and coordination in shared workspaces. *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, Toronto, Canada, 1992. ACM Press: New York, 1992; 107–114.
26. Bernheim A, Borning A. 'Today' messages: Lightweight support for small group awareness via Email. *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, Waikoloa, HI, 2005. IEEE Computer Society Press: Los Alamitos, CA, 2005; 1.
27. Steeter L, Lochbaum K. An expert/expert locating system based on automatic representation of semantic structure. *Proceedings of the 4th IEEE Conference on Artificial Intelligence Applications*, San Diego, CA, 1998. IEEE Computer Society Press: Los Alamitos, CA, 1988; 345–349.
28. Steeter L, Lochbaum K. Who knows: A system based on automatic representation of semantic structure. *Proceedings of RIAO88: User-oriented Context-based Text and Image Handling*, Cambridge, MA, 1988. CID: Paris, 1988; 380–388.
29. Mattox D, Maybury M, Morey D. Enterprise expert and knowledge discovery. *Proceedings of the 8th International Conference on Human-Computer Interaction (HCI International'99)*, Munich, Germany, 1999. Lawrence Erlbaum: Mahwah, NJ, 1999; 303–307.
30. Kautz H, Selman B, Shah M. The Hidden Web. *The AI Magazine* 1997; **18**(2):2–36.
31. Kautz H, Selman B, Milewski A. Agent amplified communication. *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI-96)*, vol. 1, Portland, OR, 1996. American Association for Artificial Intelligence: Menlo Park, CA, 1996; 3–9.
32. Wang J, Chen Z, Li T, Ma W, Liu W. Ranking user's relevance to a topic through link analysis on Web logs. *Proceedings of the 4th ACM CIKM International Workshop on Web Information and Data Management (WIDM 2002)*, SAIC Headquarters, McLean, VA, 2002. ACM Press: New York, 2002; 49–54.