Extracting Reputation in Multi Agent Systems by Means of Social Network Topology

Josep M. Pujol
Software Department, Technical University of Catalonia
C/Jordi Girona 1-3 C5-221, 08034 Barcelona, Spain
+34934017989
jmpujol@lsi.upc.es

Ramon Sangüesa
Software Department, Technical University of Catalonia
C/Jordi Girona 1-3 C6-204, 08034 Barcelona, Spain
+34934015640
sanguesa@lsi.upc.es

Jordi Delgado
Software Department, Technical University of Catalonia
C/Jordi Girona 1-3 C6-105, 08034 Barcelona, Spain
+34934017992
jdelgado@lsi.upc.es

ABSTRACT
The problem of calculating a degree of reputation for agents acting as assistants to the members of an electronic community is discussed and a solution presented. Usual reputation mechanisms rely on feedback after interaction between agents. An alternative way to establish reputation is related with the position of each member of a community within the corresponding social network. We propose a method based on this idea, which is also used by well-known ranking algorithms, discuss its properties as well as experimental results and compare them to other reputation mechanisms for electronic communities supported by agents. The method proposed uses only local information in order to extract reputation and it is able to adapt automatically to the topology of the network or graph.

Categories and Subject Descriptors
I.2.11 Distributed Artificial Intelligence---Multiagent systems, G.3. Probability and Statistics---Markov processes, Stochastic processes, I.2.8 Problem Solving, Control Methods, and Search (F.2.2)---Graph and tree search strategies, and G.2.2 Graph Theory (F.2.2)---Graph algorithms.

General Terms
Algorithms, Measurement, Human Factors.

Keywords
Social Networks, Electronic Communities, Multi agent Systems, Reputation, Trust, Reputation mechanisms, Ranking algorithms, Web Intelligence.

1. INTRODUCTION
Electronic communities evolve around a given area of activity or topic of interest. The basis for their sustainability and persistence over time is the interchange of services between community members. Members of electronic communities can be people or agents acting on behalf of the former ones. Typical interchanges include commercial transactions, documentary information and knowledge exchanges, responses to answers, advice, suggestions, help, reference to further information, etc. As well-known examples of each type of communities one can cite e-Bay [8], Firefly [11], or Experts' Exchange [9]. We focus our discussion on communities created for knowledge management and organizational learning purposes, specifically those ones supported by a multi agent system where each agent represents a single user [20,28]. Given that such communities thrive through service interchange or mutual support, it is crucial for each member to be aware of the trust that could be given to other members as well as their reputation. These two characteristics are inherently dynamic and some schemas have been devised in order to keep track of them for several types of communities. Usually they are calculated by means of some type of feedback mechanism that require involving actions on the part of the human user or the corresponding agent. This approach has some drawbacks that we try to overcome in this paper by means of an alternative approach based on the analysis of the location of each user within his community's social network [26]. In section two, the characteristics of multi agent systems for community support are presented together with currently used mechanisms for trust and reputation maintenance. A characterization of social networks is given in section three that also analyzes their properties. Section four describes a new algorithm that extracts reputation mechanism for this type of multi agent systems. Section five presents a set of experiments on a concrete knowledge sharing community and Section six closes by discussing results and pointing to further work.

AAMAS '02, July 15-19, 2002, Bologna, Italy.

2. MAS FOR COMMUNITY SUPPORT AND THEIR REPUTATION MECHANISMS
Knowledge management in collaborative environments involves the interchange of information and knowledge among members of a community. The knowledge management cycle involves detecting when new knowledge is generated, who may be interested about it and delivering this knowledge to that people. Several approaches have been proposed in order to develop systems that detect each user's knowledge needs [5]. Usually they

are based on measuring the similarity in the competences of community members. This requires the maintenance of some type of *user profile*. Several multi agent systems have been developed to serve these purposes. Their agents act on behalf of community members, maintaining profiles or routing questions to other members' agents as, for example, in *MARS* [28] or the *i2CAT Collaboratory* [22, 23]. Communities based on user preferences also share some treats with these systems. For example, agents in Firefly [11] or Yenta [12] use profile similarity as a criterion for finding possible partners, which is at the basis for locating expertise within a community [24]. This same idea is used by recommender systems based on collaborative filtering [19]. In the case of knowledge sharing communities formed by a set of people with expertise in a given domain not only it is important that agents be able to detect which people (represented by other agents) possess the adequate expertise for solving the problem at hand but it is also crucial to assess to which extent some of the members are recognized as experts by their colleagues in the community. The trust and reputation of experts has been typically assessed as a function of the quality of their response to knowledge requests coming from other members in the community. This is the schema used in some organizational learning systems as, for example, *Answer Garden* [1] and some commercial knowledge communities [4,9,10], see [13] for a discussion of the different ratings that can be obtained by analyzing response quality. All these systems rely on feedback (in the form of a rating) from the person receiving the response to a previous demand. By combining these ratings a numerical value for each expert's reputation can be calculated. A reputation measure gives an idea of the confidence one can have on the quality of an expert's responses and serves as a basis for guiding the search for experts. The disadvantage of this type of reputation mechanism is that it needs the explicit and frequent involvement of users that issue ratings. This implies that a good reputation calculation and maintenance depends critically on the involvement of users and continued contribution of ratings. Usually an insufficient number of users or a low participation are at the root of bad collaborative filtering performance. Continuous requirements for feedback also create fatigue in users who tend to end up not issuing ratings at all. See a discussion about this in [14,19]. So, alternative, less intrusive and less demanding in terms of involvement methods are clearly of interest. This of course, raises the question of how can reputation be measured in the absence of any user feedback for expert's responses. This goal is also important in devising reputation mechanisms for other types of agent systems. See [21] for a general discussion of trust and reputation in MAS and [29] for a similar discussion in a more restricted type of community dealing with e-commerce. We propose a system that tackles these problems by using the social network of the community.

2.1 Social Networks

A *social network* [26] is a representation of the relationships existing within a community. Even within the same community several types of social network can be built depending on the social relationship taken into account: kinship, acquaintanceship, friendship, mutual support, cooperation, and similarity are typical criteria used in establishing the social relationship components of a community. The corresponding social networks are represented as graphs. The construction of social networks for electronic communities helps in mapping the relationships among people that may not be aware of being related, given the special type of detached interaction peculiar to online communities. The basis for reconstructing these social interactions are the usage of the tools that give support to the online community which involve to some degree public information about each member [7]. See *Referralweb* for a multiagent system supporting that type of community mapping [15] and *MARS* [28] or *NetExpert* [20] for multi agent systems supporting knowledge sharing communities on the basis of the corresponding social network structure.

2.2 Social Network topology and reputation

The location of a given member of a community within a social network can be used to infer some properties about his or her degree of expertise, i.e., his or her reputation. Experts who are well-known and highly regarded by most other members of the community tend to be easily identified as highly connected nodes in the social network graph of their community [2,16,17]. This relation information could be a basis for a reputation mechanism used by users' assistant agents instead of having to resort to explicit ratings issued by each user. We describe in the following how this approach has been tested in the context of a multi agent system that gives support to a knowledge sharing community.

3. SOCIAL NETWORKS IN A COLLABORATORY

The *Collaboratory* is a multi agent system supporting a research community. For a description detailed description, see [22,23]. The users of this system are currently the researchers involved in the Internet2 project in Catalonia, i2CAT *(http://www.i2.-cat.net)*. They share a document repository and use a set of collaborative agents to obtain recommendations or for finding experts in a given topic within the community. Central to these functionalities is an agency that builds and maintains the social network of the community [18,20]. The social network building agency uses information located in he community members personal web pages and other sources. This information can be complemented with the knowledge contributed by users to the rest of the *Collaboratory* as well as the knowledge they obtain from it. This is done by resorting to several user profiles. Details of their construction can be seen in [19]. For the purposes of the experiments discussed in this paper the test community has been extended to all members of the Software Department of the Technical University of Catalonia (UPC).

Each type of community may need its particular sources of information [26], for social network construction. Some of them are:

- Personal web pages
- Reports or documents authorship
- Participation in a project
- Hierarchical structure in the community or organization.
- Sharing of physical resources.
- Sharing of virtual resources such as news group, forums, etc.
- Email traffic.

3.1 Building the Social Network

In order to build the social network for the present experiment it was necessary to create knowledge and content models of users' personal web pages. This was done by using *WebMining*, which is an agent that builds generic models from web pages, see [18,20]. For simplicity purposes one can see that model as a set of *terms* describing the knowledge contained in the personal pages. Once these models are obtained for each member of the community the Social Network is built by the *RelationshipMaker* agent by means of a similarity function that takes into account the strength of association between any two members in the community. Members of the community appear as nodes in the social network graph and the directed edges are calculated as follows.

$$w(a \to b) = w_{email}(a \to b) + w_{link}(a \to b) + w_{name}(a \to b) \quad (1)$$

The weight of a relationship is the sum of three factors:

$$w_{email}(a \to b) = \exists email(a \to b) \quad (2)$$

$\exists email(a \to b)$ is 1 when email address of member *b* exists in the web pages of member *a* , 0 otherwise.

$$w_{link}(a \to b) = \sum_{\forall \alpha \in R(a,b)} \frac{2}{depth(\alpha, a) + depth(\alpha, b)} \quad (3)$$

$R(a,b)$ is the set of resources (files than can be reached through the Web) that belong to member *b* and they have been found in the personal web page of member *a*.

$depth(\propto,x)$ is the depth of the resource $\propto$ in the personal web page of member *x*.

$$w_{name}(a \to b) = \frac{1}{2} \sum_{i=0}^{i \le \#name(a \to b)} \frac{1}{i^2} \quad (4)$$

#name(a→*b)* returns the number of occurrences of the name of member *b* within the personal web pages of member *a.* Extracting the name of a member of the community from the personal web page of another one is not trivial. Web pages are plain text information with html tags, there is no semantic information about names, thus a set of rules has to be used to disambiguate names, see [18] for the details.

A second, undirected graphs, results from this step also. Their weight are calculated as follows:

$$w(a \leftrightarrow b) = w(a \to b) + w(b \to a) \quad (5)$$

3.2 Social Network properties

The social network generated using the proposed heuristics has the typical properties of the social networks. Figure 1 shows a partial view of the UPC's undirected social network.

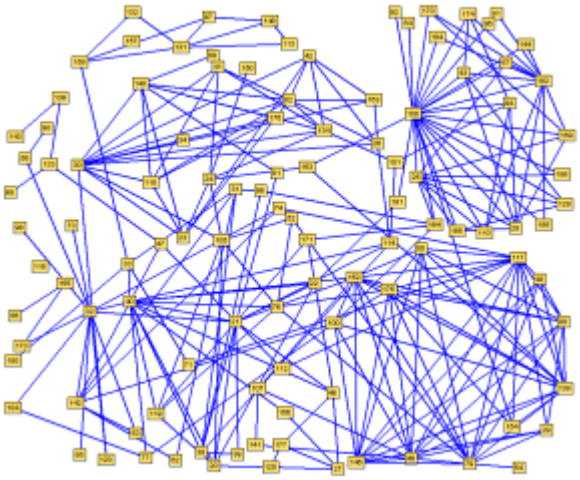

Figure 1. Fragment of Social Network

Table 1 shows the most important characteristics of the undirected Social Network. To formalize the notion of a small world, Watts and Strogatz [27] define the clustering coefficient *C*, and the characteristic path length *L*. We take the diameter of a graph as well *D*. [3] Clustering coefficient is a real in the interval [0..1].

	Social Network	**Random Network**
#nodes	139	139
#edges	394	394
C	0.7045	0.3774
L	3.6572	3.016
D	9	6

Table 1: Social Networks characteristics

Watts and Strogatz define a small world graph as one is which $L \geq L_{rand}$, or $L \approx L_{rand}$, and $C >> C_{rand}$ where L_{rand} and C_{rand} are the characteristic path length and clustering coefficient of a random graph with the same number of nodes and edges. It is accepted that a small world graph should has $(C/C_{rand})/(L/L_{rand}) > 1$, our Social Network has $(C/C_{rand})/(L/L_{rand}) = 1.5394$.

The distribution of the node degree of the UPC Social Network follows clearly an exponential distribution, few nodes with very high degree and many with low degree. That confirms that the social Network is a small world in the Watts and Strogatz sense. In figure 2 the node degree histogram is shown in a log scale. Because of the few nodes that the UPC Social Network has, the cumulated frequency has been used instead of the frequency. Thus, the fluctuation produced by the lack of points can be reduced.

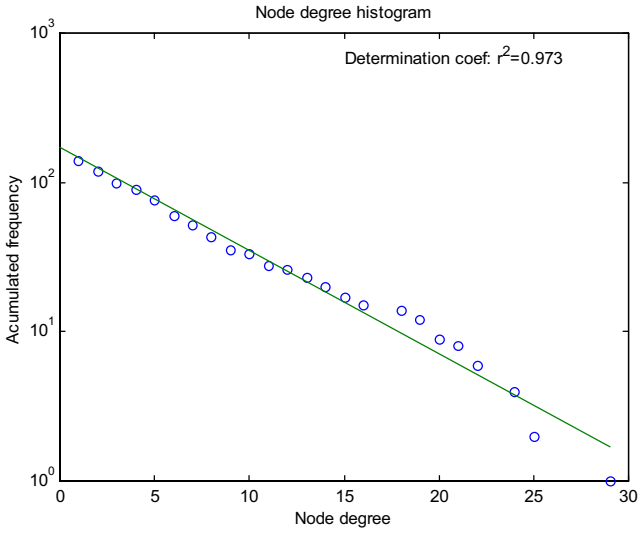

Figure 2. Node degree Histogram

Social network properties such as "smallworldliness" should be considered in order to improve knowledge extraction from the network and expertise location, because by exploiting these properties more efficient search [25] and propagation algorithms can be used, instead of traditional generic graph algorithms.

4. USING SOCIAL NETWORK METRICS FOR REPUTATION MEASUREMENT

NodeRanking is our proposal for creating a ranking of reputation ratings of community members by means of the corresponding Social Network.

The rating that *NodeRanking* creates is based on the idea that each node on the graph has an associated *degree of authority* that can be seen as an importance measure. Initially, all nodes are assumed to have the same authority. After running *NodeRanking*, the resulting authority measure is used to infer the reputation of a node within the graph, that is, the reputation of a member within his community.

Authority of a node, *a*, is calculated as a function of the total measure of authority present in the network and the authority of the nodes pointing to *a*. If a node is not referred by any other node in the network, it is assigned a default authority value. Authority values are positive values.

The algorithm for authority calculation is inspired in the ranking algorithms for web pages based on web topology [16,17]. The idea is to apply a similar reasoning about link topology in web pages to the topology of a Social Network, i.e., the link topology of a directed graph. In a directed graph the edges have a direction, the out-edges of a node are the edges that start in this node, the out-nodes are the nodes that can be reached through out-edges.

The main idea of the *NodeRanking* algorithm is that each node has an authority and a part of this authority is propagated to the out-nodes via out-edges.

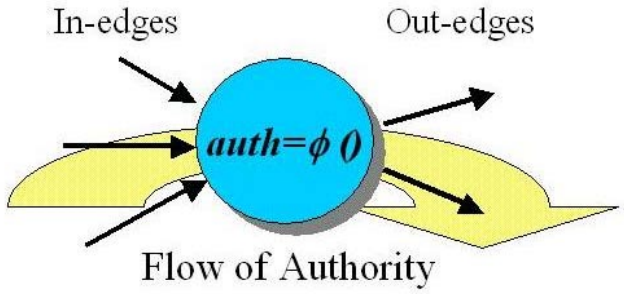

Figure 3. Flow of authority in *NodeRanking*

The authority of a node depends on the authority of its in-nodes. The authority of one of the out-nodes in figure 3 node depends in part on the authority of this node. Cycles in the graph can produce critical deadlocks in the calculations. The *NodeRanking* algorithm overcomes these problems and insures convergence.

4.1 The *NodeRanking* Algorithm

NodeRanking follows essentially the *random walker strategy* to explore the graph. It starts in a randomly selected node, and proceeds by selecting one of the nodes that can be reached through out-edges.

```
do
 n ← getNode(v)
 do
  passAuthority(n)
  nnew ← getNextNode(n)
  n ← nnew
 while (∃ nnew)
while (¬ converge( ))
```

Figure 4. *NodeRanking* algorithm

We have used some functions that require additional explanation.

getNode (): returns a randomly chosen node of the graph. The probability is uniform.

getNextNode(node x): returns one of the out-neighbors nodes of node *x*. Each node has a set, that can be empty, of out-edges that points to other nodes. This function can return a null node to stop the path of the random walker by introducing some elements of randomness. When *getNextNode()* returns a null node, in fact, the path is broken. There are two cases where the path is broken. The first one is when the algorithm arrives at a node that has been already visited in the previous *k* steps. The algorithm keeps a window of *k* elements and breaks the path if a new node already exists in the window. The second case is when a certain value of jumping probability is reached. In effect, the random walker evaluates the jumping probability whenever it reaches a node. This probability is a function of the connectivity of the node. The walker skips the node with a probability Pr_{jump} *(a)*.

$$\Pr_{jump}(a) = \frac{1}{\#out-edges(a)+1} \quad (6)$$

Nodes with fewer out-edges have a greater probability of breaking the path. This could be seen as a walker that gets bored because of the reduce range of choices. *getNextNode( )* returns the next node *b* to be visited from *a*. This node is selected with a probability that is calculated as a function of the weight of the edge between *a* and *b*, the corresponding density probability function is shown below.

$$\Pr_{choose}(a \to b) = \frac{w(a \to b)}{\sum_{\forall \alpha \in out-nodes(a)} w(a \to \alpha)} \quad (7)$$

Where *w (a -> b)* is the weight of the link connecting *a* and *b*.

passAuthority(node x): this function assigns part of the authority of node *x* to all the nodes that are pointed by *x*. The next equation shows the change of authority between a node *x* and a node *y*.

$$auth(y) = auth(y) + \left[\frac{\Pr_{choose}(x \to y) auth(x)}{F_y} \right] \quad (8)$$

Where *auth(y)* is the authority of the node *y* and F_y is a factor to maintain the value of authority within a limited range of values. Without this factor, values calculated with equation 8 would tend to infinity because the authority of a node gets higher and higher as the method proceeds. *F* controls the growth of authority and also eliminates the effects of randomness introduced by the asynchronicity in the authority updating. Each node has its own *F* factor that remembers the state of the total authority value of the graph the last time that the node was involved in a *passAuthority* function call. These factors depend on the authority value of all the nodes in the graph at the time that the authority of node *y* was updated. The factors of more frequently visited nodes grow faster than the values of the less visited ones. The growth of these factor's values is monotonously increasing. Thus, we can insure convergence towards a finite value. Without this factor, nodes that had been last in the random selection would have an advantage over the other ones, because the graph is accumulating more and more authority as the algorithm proceeds. Factor *F* is initialized for every node as the sum of the authority of all nodes in the graph. The initial authority of a node has to be positive, and factor *F* has to be bigger or equal than 1.

converge( )*:* this function can be evaluated anytime, it's a test on all the nodes in the graph. Each node remembers its last increment in authority. The increment in authority tends to 0 because of the *F* factor, as can be seen in equation 8. The function *converge()* tests the state of each node. If the increment is less than a given threshold, φ, the node will be considered as stationary. When all the nodes of the graph are stationary the algorithm ends. Actually the convergence function does not test all the nodes because it would not be very efficient. The event of becoming stationary is notified by the nodes themselves.

4.2 Differences with *Pagerank* and *HITS*

HITS [16] and *Pagerank* [17] algorithms are the reference for web page ranking algorithms. The second one is the core of the ranking performed by the commercial *Google* search engine. They are based on finding out the stationary state of a matrix, the variance-covariance matrix in the case of *HITS* and the transition probability matrix for *Pagerank*. In order to do this, the adjacency matrix of the graph has to be known. Moreover, when the principal eigenvector is calculated using the technique known as "iterative product" the vector that contains the principal eigen vector, has to be normalized frequently, so the whole vector has to be known. *HITS* and *Pagerank* use global information of the graph. That means that it is difficult to use directly these algorithms to rank huge graphs. Let's think, for example about a not so big graph, let's say a graph of 25000 nodes. Its adjacency matrix will have $6.25*10^8$ positions. Because of the authority updating mechanism that *Pagerank* and *HITS* use, there is a synchronization step in the process of authority transfer since the state of all nodes has to be know in order to update the state of a single one of them. On the contrary, our algorithm, *NodeRanking*, uses only local information. Each node *x* only needs to know the about nodes that it points to and these other ones have to be aware of *x*'s convergence towards the stationary state. In order to retrieve the results of the ranking, a centralization point is required, but even in this process the communication proceeds unidirectionally (from the nodes to the controller) so it is not necessary at all to have information about the whole graph.

Pagerank and *NodeRanking* are almost identical in the underlying idea, both of them follow the same random walker strategy. Actually the transition probability matrix of a synchronous version of *NodeRanking* can be defined as follows:

$$M = J\frac{1}{n}11^T + (11^T - J)P \quad (9)$$

Where 1 is a vector of 1s, *n* is the number of nodes, *P* is the adjacency matrix normalized by rows, *J* is the jumping probability matrix defined as a *0* matrix where the diagonal contains the jumping probability of a node, i.e. J_{ii} contains node *i* jumping probability (as defined in equation 6). The results obtained by finding out the stationary state of the Markov chain defined by the transition probability matrix *M*, i.e finding out the principal eigenvector of the matrix M^T, are equivalent to the results obtained by the original *NodeRanking* algorithm. The advantage of the original *NodeRanking* is that it is not necessary to know the graph's adjacency matrix as it happens in techniques based on transition probability or variance-covariance matrices.

As it can be seen in equation 9, matrix *J* contains the jumping probability of a node. Each node has a different jumping probability, calculated by applying equation 6. So, the jumping probability only depends on local information. The average jumping probability depends on the distribution of the nodes' out-degrees. Thus *NodeRanking* is able to adapt dynamically to graphs with different topologies because the jumping probability depends only on the node's out-edges. This is clearly in contrast with *Pagerank* where the jumping probability is not adaptative, and it is fixed to values in the [0.1..0.2] range, usually 0.15.

5. EXPERIMENTS ABOUT RANKING AND REPUTATION

In order to validate the obtained rankings as a correct reputation measure they had to be compared against a real and accepted measure of reputation. This test community being a research community, a good way to establish a comparison was to resort to citation index impact values for each one of the 34 members calculated by an independent scientific publication ranking agency. That is, citation indexes for each of the 34 randomly selected members of the Software Department were compared against the rating that *NodeRaking* yielded, $Rank_{NodeRanking}$, and also against the ratings that *Pagerank* yielded. Citation indexes are a clear indicator of reputation in scientific communities. *CiteSeer* [6] was used as a source for citation index values. In *CiteSeer*, the papers of each researcher can be retrieved together with the corresponding number of citations and self-citations for each of his or her papers. We have built two rankings for the members of our test community. $Rank_{cite}$, sorts researchers by number of citations and $Rank_{cite\text{-}self}$, sorts researchers by number of citations without counting self-citations. These rankings can be considered as reference rankings, the closest ones to real reputation in the scientific community.

NodeRanking was applied to the social network of the experimental community formed by the members of the UPC Software Department. The parameters that have been used along all the experiment are k=4 and φ=10^{-6}. Afterwards, a fragment of the community, 34 members, was selected randomly to become the test set. The ranking obtained by means of *NodeRanking* was called $Rank_{NodeRanking}$. The ranking obtained by *NodeRanking* was compared against the results obtained by applying *Pagerank* with the e=0.15 and also against the results of the *HITS* algorithm.

Pagerank and *HITS* work with no weighted graphs. Weighted edges can be useful, if they are available, because the normalized weight of an edge contributes with more information than the fact that an edge exists or not. The UPC Social Network *has* weighted edges, so two rankings instead of one were built for each algorithm and one was calculated without taking the weights into account. It was called $Rank_{PageRank}$ or $Rank_{HitsAuthority}$ The second ranking, $Rank_{PageRank(w)}$, was calculated by having weights into account.

To compare the quality of the ratings obtained by *NodeRanking*, *Pagerank, HITS* and the reference rankings built using *Citeseer* the correlation coefficient between rankings was used as a similarity measure.

Table 2 shows the correlation values between reference rankings $Rank_{cite}$, $Rank_{cite\text{-}self}$, considered as the desired rankings, and the rest. The ranking $Rank_{NodeRanking}$ is the average of twenty executions of the algorithm, the mean and the standard deviation is given. That variability is due to the asynchronicity of the authority transfer process within *NodeRanking*. Independently of the reference ranking, the same order among the generated rankings is obtained. $Rank_{NodeRanking}$ values are always a little better than $Rank_{PageRank}$ and $Rank_{PageRank(w)}$ values and these ones are better than the $Rank_{HitsAuth(w)}$ and $Rank_{HitsAuth.}$ It is interesting to remark that the rankings obtained by *HITS* algorithm are better if the edges' weights are taken into account. However, in any case, the correlation is very strong. Correlation coefficient goes from [-1..1], where 0 means that there is no correlation, the correlation between two random rankings is close to 0. Rankings with a correlation of 0.621 or 0.687 against the reference ranking can not be considered perfect even though they are similar enough to be interesting.

Correlation Coefficient	$Rank_{cite}$	$Rank_{cite\text{-}self}$
$Rank_{cite}$	1.0	0.983
$Rank_{cite\text{-}self}$	0.983	1.0
$Rank_{NodeRanking}$	0.687, s=8.6*10^{-4}	0.621, s=0.011
$Rank_{PageRank(w)}$	0.535	0.486
$Rank_{PageRank}$	0.521	0.495
$Rank_{HitsAuth(w)}$	0.412	0.383
$Rank_{HitsAuth}$	0.342	0.323

Table 2. Correlation values between rankings

Figure 5 shows the correlation between $Rank_{cite}$ as a desired ranking and the rest of rankings

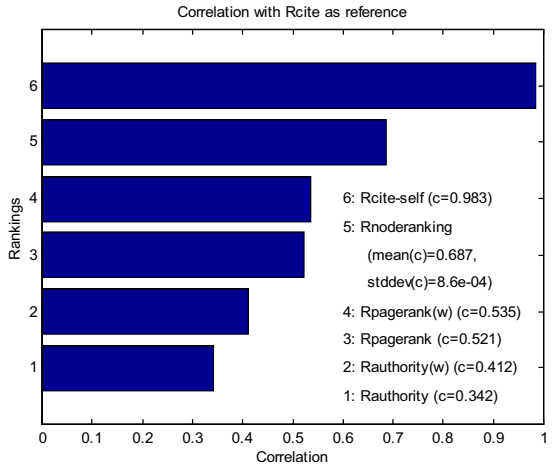

Figure 5. Correlation with $Rank_{cite}$ as reference

Figure 6 shows the correlation between $Rank_{cite\text{-}self}$ as a desired ranking and all the rest.

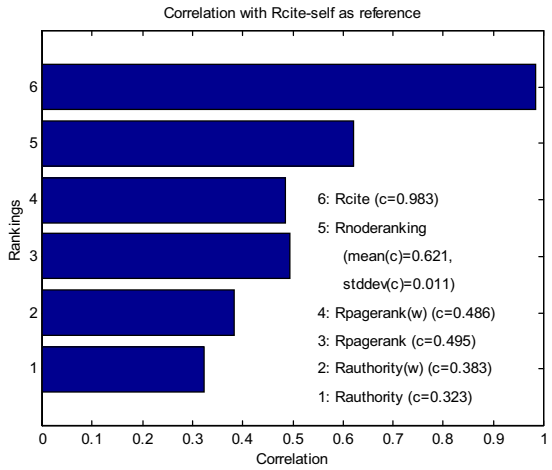

Figure 6. Correlation with $Rank_{cite\text{-}self}$ as reference

As it can be observed, the rankings obtained by *NodeRanking* are better than the ones obtained by *Pagerank*. Both algorithms follow the same random walker strategy with the difference of the authority updating mechanism, which is asynchronous in *NodeRanking* and synchronous in *Pagerank*. So, the results should be similar, the reason of the best performance of *NodeRanking* may be due to its ability to adapt itself to the graph's topology.

In figure 2 the distribution of nodes out-edges degrees can be observed. It follows an exponential distribution that is characteristic of the graphs that are small worlds such as the Social Network of our case of study is. The average jumping probability can be calculated using this distribution that we call $d_{out\text{-}connectivity}$.

$$\overline{\Pr_{jump}} = \sum_i d_{out-connectivity}(i)\Pr_{jump}(i) \quad (10)$$

The average jumping probability of our case of study social network is 0.5314. So *NodeRanking* with its dynamical adjustment of the jumping probability is able to adapt to different graph topologies. In contrast, *Pagerank* is not able to calculate such good rankings for social networks nodes because of the exponential distribution of the connectivity in the graphs. There is a lot of nodes with no out-edges or with very few out-edges. So, a jumping probability of 0.15 (the one used in *Pagerank*) is too small and *Pagerank* falls into the *rank sink* problem. If *Pagerank* were set to work with a jumping probability of 0.5414 the results would be equivalent to the obtained by *NodeRanking*. This is to be expected and perfectly coherent. However, the fact that *NodeRanking* only uses local information is an advantage over *Pagerank* in order to extract reputation and being more general extracting ranks.

6. DISCUSSION AND FURTHER RESEARCH

The values of the ratings calculated by the topological information of a community's social network has been used as a measure of each member's reputation within a community. A new algorithm *NodeRanking* has been devised to obtain such measures. An experiment has been performed on a real community and the results compared against a well-known valid measure of reputation for that type of communities. The results seem to indicate that *NodeRanking* values are a good approximation of reputation measures. The well known *Pagerank* and *HITS* algorithms have been also used to find out the member's reputation, the *Pagerank* results has been slightly worse than the results yielded by *NodeRanking*. It can be concluded that NodeRanking builds good approximation of reputation. However *NodeRanking* is able to assess the reputation only by using local information. It is a distributed algorithm. *NodeRanking* does not need to know the entire graph to operate which *Pagerank* does.

The proposed criterion for reputation measurement in a collaborative multi agent system and the corresponding method have as an advantage with respect to other ones [29] the fact that it does not require to have users continuously and explicitly issuing ratings, a method that is seen as a burden on users and eventually a reason for poor performance of collaborative systems. With our proposal a quite approximate reputation ranking can be calculated a priori without the typical and annoying feedback request used in collaborative systems.

Another advantage of *NodeRanking* is that the proposed method is solely based on topological information, thus making complete abstraction of any other information. On the other hand its success hinges critically on the quality of fit between the social network representation and the real community structure. In the case of the social network of the *Collaboratory* it seems that the information used to build the social network is well suited to research communities.

Other multi agent system methods that use social networks either do not use them for reputation measurement, as is the case of *Referralweb* [15] or still rely exclusively on rating feedback from users as Yu [28] does. This last one has only been tested on a simulated community as opposed to the test we carried on a real one (another example of this tests on simulated communities is Zacharias and Maes [29]).

Further experimentation will be carried on with other types of knowledge sharing communities in order to test the dependence between the information used in building the social network and the final quality of the reputation measurements obtained from it.

Finally, and as another interesting field, *NodeRanking* could be tested as a generic and fully distributed algorithm to rank nodes of any kind of graph, such as the Web. Following this line some interesting results has been obtained applying *NodeRanking* to web-like graphs. Rankings that are equivalent to the reference ones, such as *Pagerank* and *HITS,* has been obtained using only local information.

7. ACKNOWLEDGEMENTS

The research leading to these results have been partially supported by the Catalan Autonomous Government DURSI project i2CAT.

8. REFERENCES

[1] Ackerman, M.S., and McDonald, D.W. Answer Garden 2: Merging organizational memory with collaborative help. In Computer Supported Cooperative Work, pages 97–105, 1996.

[2] Adamic, L.A., and Adar, E. Friends and neighbors on the web, 2000.

[3] Adamic, L.A., The small world web. In S. Abiteboul and A.-M. Vercoustre, editors, Proc. 3rd European Conf. Research and Advanced Technology for Digital Libraries, ECDL, number1696. Springer-Verlag, 1999.

[4] AskMe. Available on http://www.askme.com/.

[5] Borghoff, U.M. and Pareschi, R.(Eds.), Information Technology Support for Knowledge Management Springer-Verlag. Berlin.1998.

[6] CiteSeer. Available on http://www.citeseer.com/.

[7] Contractor, N.S., O'Keefe, B.J. and Jones, P.M. IKNOW: Inquiring knowledge networks on the web. Computer Software Dept., University of Illinois, 1997.

[8] eBay. Available on http://www.ebay.com

[9] Experts Exchange. Available on http://www.experts-exchange.com/.

[10] Exp.com. Available on http://www.exp.com/.

[11] Firefly. Available on http://www.firefly.net/.

[12] Foner, L. Yenta: A multi-agent, referral-based matchmaking system, 1997.

[13] Garcia, R. Extensió col.laborativa del servei de localització d'expertesa. Master's thesis, Technical University of Catalonia, 2001.

[14] Grasso, A. Mixing Cognitive and Collaborative Filtering. Proceedings of the I3net Community of the Future Conference. Sienna, Italy, October 1999.

[15] Kautz, H., Selman, B., and Shah, M. The hidden web. AI Magazine, (18), 1997.

[16] Kleinberg, J. Authoritative sources in a hyperlinked environment. Technical Report RJ 10076, IBM, May 1997.

[17] Page, L., Brin, S., R. Motwani, T. Winograd, The PageRank citation ranking: Bringing order to the Web, submitted for publication.

[18] Pujol, J.M. Netexpert: Localitzador d'expertesa. Master's thesis, Technical University of Catalonia, 2000.

[19] Sangüesa, R., Vázquez, A. and Vázquez, J. Ace: A multiagent recommender system using mixed collaborative and cognitive filtering. In Proceedings of the WARS 2000, Workshop on Agent-Based Recommender Systems, 2000.

[20] Sangüesa, R. and Pujol, J.M. NetExpert: A multiagent system for expertise location. Accepted to IJCAI'01 Workshop on Organizational Memories and Knowledge Management, 2001.

[21] Schillo, M., Funk, P. and Rovatsos, M. Who can you Trust: Dealing with Deception. In C. Castelfranchi, Y. Tan, R. Falcone and B. S. Firozabadi (ed.), Proceedings of the Workshop "Deception, Fraud and Trust in Agent Societies" of the Autonomous Agents Conference, 1999.

[22] Vázquez, A., Vázquez, J., Barrio, I., Pujol, J. M. Pujol, and Sangüesa, R. The collaboratory. Technical Report LSI-01-22-R., Technical University of Catalonia, 2001.

[23] Vázquez, A., Barrio, I., Vázquez-Salceda, J., Pujol, J.M. and Sangüesa, R. An agent-based Collaboratory. Accepted for publication at ACAI'01, Advanced Course on Artificial Intelligence, Prague, July 2001.

[24] Vivacqua, A.S. Agents for expertise location. In Proceedings of the 1999 AAAI Spring Symposium on Intelligent Agents in Cyberspace, 1999.

[25] Walsh, T. Search in a small world. In Proceedings of the 16th International Joint Conference on AI (IJCAI-99-Vol2), pages 1172--1177, S.F., July.

[26] Wasserman, S. and Glaskiewics, J. Advances in Social Networks Analysis. Sage Publications, 1994.

[27] Watts, D.J., and Strogatz, S.H., Collective dynamics of 'small-world' networks. Nature, (393), 1998.

[28] Yu, B. and Singh, M.P, A Social Mechanism of Reputation Management in Electronic Communities. Proceedings of Fourth International Workshop on Cooperative Information Agents, pages 154-165, 2000.

[29] Zacharias, G. and Maes, P., Trust Management Through Reputation Mechanisms, Applied Artificial Intelligence 14, pp 881-907, 2000.