Simulating a trust-based service recommender system for decentralised user modelling environment

Sabrina Nusrat* and Julita Vassileva

Department of Computer Science,
University of Saskatchewan,
176 Thorvaldson Building, 110 Science Place,
Saskatoon, SK S79 5C9, Canada
E-mail: san986@mail.usask.ca
E-mail: jiv@cs.usask.ca
*Corresponding author

Abstract: Trust and reputation mechanisms are often used in peer-to-peer networks, multi-agent systems and online communities to differentiate among members of the community as well as to recommend service providers. Although different users have different needs and expectations in different aspects of the service providers, few exiting trust models use differentiated trust values for judging different aspects of service providers. We have proposed a multi-aspect trust model where each user has two sets of trust values: 1) trust on different aspects of the quality of service providers; 2) differentiated trust on the recommendations provided by other users for each of these aspects. This trust model is used to recommend service providers in a decentralised user modelling system where agents have different preference weights in three different criteria of service providers. The paper focuses on the evaluation of the approach via a simulation on a large real social network. The results show that the trust model allows agents to learn from experience to find good service providers by using recommendations from their friends and that the model is robust with respect to colluding malicious agents.

Keywords: trust; reputation; user modelling; recommendation.

Reference to this paper should be made as follows: Nusrat, S. and Vassileva, J. (2013) 'Simulating a trust-based service recommender system for decentralised user modelling environment', *Int. J. Trust Management in Computing and Communications*, Vol. 1, No. 2, pp.121–139.

Biographical notes: Sabrina Nusrat received her Master in Computer Science in 2012 from the University for Saskatchewan, Canada. She did her Bachelor in Computer Science and Engineering from Bangladesh University of Engineering and Technology (BUET). Her research interests include trust and reputation mechanisms, social networking, user modelling systems, recommender systems and study of polyhedra.

Julita Vassileva is a Professor of Computer Science at the University of Saskatchewan, Canada and one of the directors of the MADMUC Lab. She received her PhD in Mathematics (Cybernetics and Control Theory) from the University of Sofia/Bulgarian Academy of Sciences. Her research areas involve human issues in decentralised software environments: user modelling and personalisation, designing incentive mechanisms for encouraging user participation and facilitating trust in decentralised software applications, such as online communities, social networks, open learning environments, multi-agent systems, and peer-to-peer systems.

1 Introduction

Searching for good services suitable to meet the preferences of users is an important problem both in everyday life (e.g., looking for a good doctor) and in online activities (e.g., finding suitable web services). In virtual environments, the anonymity and sparseness of networks expose users to significant risks in their interactions, since they can encounter unreliable, malicious or dishonest users. Trust and reputation mechanisms have been proposed to help users distinguish good members of the community from bad ones. Here, trust is one person's belief in another person's honesty, capability and reliability based on direct interaction while reputation, on the other hand, is the collective measure of trust. Usually, trust mechanisms model the processes of sharing information about services among people in a community, e.g., one can ask trusted others (acting as referees), who have experience with a given person or service provider, for recommendations (Yu and Singh, 2000).

One important feature of trust is that it has multiple aspects. Wang and Vassileva (2003a) considered three aspects of file providers' capabilities in a trust model in file-sharing P2P networks. Users have differentiated trust in the file provider according to multiple aspects of the service and share their trust values in these different aspects. A natural extension of this work would be that, in the case of asking for recommendation, users consider also multiple aspects of trust in the referees, depending on their competence in evaluating a particular aspect of the file provider. For example, one can trust his friend's judgment about a paper's readability but may not trust his judgment about the paper's contribution. A differentiated trust model in the referee according to the user's preference criteria would allow considering recommendations more adequately.

This paper presents a multi-faceted trust model where each user has two sets of trust values:

1 trust in different aspects of the quality of service providers

2 differentiated trust in the recommendations provided by other users (referees) for each of these aspects.

The trust in different aspects of the quality of service providers corresponds to a model of the users' preferences (i.e., which aspects are relevant to the user), in order to find the best possible service providers according to the users' needs. As the multi-faceted trust model is presented in Nusrat and Vassileva (2012) and presented only briefly here in Section 3 for completeness, the main focus of this paper is the evaluation of the differentiated trust model and mechanism and an analysis of its resistance to manipulation. The results show that with growing number of requests in the system, users learn from their friends about service providers that closely match their expectation and preferences. The paper is organised as follows: Section 2 presents a brief review of related previous research, Section 4 presents the design of the Erlang simulation environment used to evaluate the model, Section 5 presents the experimental setup and results and Section 6 presents an evaluation of the robustness of the approach with respect to shill attacks.

2 Related work

In this section, we present a brief review of related approaches for recommending resources and services in the field of trust and reputation mechanisms and user modelling.

2.1 Trust and reputation

Trust is defined as one peer's belief in another peer's capabilities based on its direct interaction with that peer (Wang and Vassileva, 2003b). Reputation is defined as one peer's belief in another peer's capabilities based on the recommendations received from other peers (Wang and Vassileva, 2003b).

Wang and Vassileva (2007) divide trust and reputation models into two distinct classes: centralised vs. decentralised. In a centralised approach, there is a central server which stores the trust values of all agents and calculates the reputation for each agent or service provider by aggregating these values. In a decentralised environment, on the other hand, agents can use social information exchange mechanisms. They share information – references, or 'gossip' (Yu and Singh, 2000) among each other and each calculates the reputation of a subset of other agents. Therefore, in decentralised systems there is no 'agreed-upon' reputation of agents, but the reputation of each agent is subjective and depends on the preferences and social network of the agent who calculates it.

In traditional models of trust and reputation, trust is represented as single value and multiple aspects of trust are not considered. An extension to the single-value trust model was proposed by Wang and Vassileva (2003a) in the context of a file-sharing peer-to-peer network, where an agent may consider different aspects of trust in file providers according to its own preferences. The Bayesian network-based trust model by Wang and Vassileva (2003a) allows agents to combine trust in different aspects to calculate the total trust in service providers. Unfortunately, this approach has not been followed up by other researchers in the area of trust and reputation, with the exception of Hang et al. (2009) who describe propagation of trust by three operators (concatenation, aggregation and selection) and claim that their approach can accommodate multidimensionality of trust. However, their approach does not model the user's preference over these multiple aspects of trust. Modelling user's preferences is a problem addressed in the area of user modelling which we present briefly below.

2.2 User modelling and recommender systems

A user modelling system aims to represent and reason about the goals, preferences, behaviours and information needs of human users (Kobsa, 1992). User models are deployed in a wide range of applications and systems aiming to personalise their services or information to the needs of the user. For example, an expert finding system (Yimam-Seid and Kobsa, 2003) needs to know the goals and preferences of the user, as well as the expertise and certain features of experts in order to generate good matches that fit the needs and the preference criteria of users.

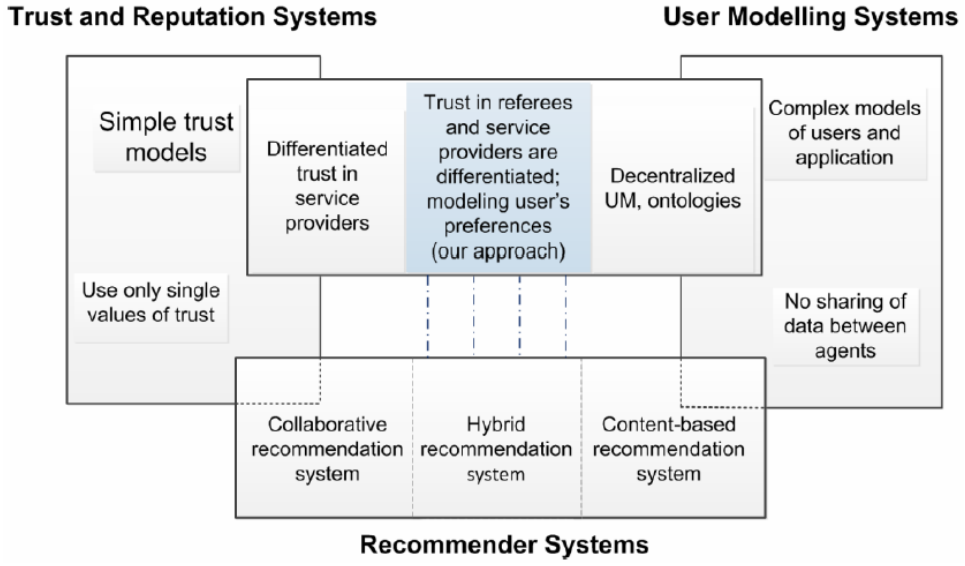
Most user modelling systems use centralised architectures, where information about the user (e.g., preferences, skills, etc.), and features of the available services, experts, or items are stored and matched by a central server (Fink and Kobsa, 2000; Kay et al.,

2002). In decentralised user modelling architectures, which are becoming increasingly popular with the proliferation of social networking sites, service-oriented architectures, cloud-based and mobile applications, user and expert information is stored by autonomous applications, and is shared and combined by these applications to make recommendation decisions depending on the purpose and context at hand (Vassileva et al., 2003; Heckmann and Krueger, 2003; Niu et al., 2004).

The problem of generating recommendation for a specific user can be seen as the problem of estimating the user's ratings of the items that have not yet been seen by the user (Adomavicius and Tuzhilin, 2005). The estimation is usually based on correlating the ratings given by this user to items in the past with the ratings of other users who have rated the same items. Once the estimated ratings of unrated items are calculated, the system recommends to the user the item(s) with the highest estimated rating(s). This type of recommender systems is called collaborative (Shoham and Balabanovic, 1997). Another type of recommender systems is content-based, where the users are recommended items similar to the ones they preferred in the past (Shoham and Balabanovic, 1997).

There is a similarity between users' trust and service ratings (Matsuo and Yamamoto, 2009). Users put trust in other users with whom their preferences match (demonstrated by a similar history of ratings) and the ratings of users are also influenced by the ratings of other users that they trust. The opinions of trusted partners can be used to create recommendations (Dell'Amico and Capra, 2008). Thus, the recommendation problem can be addressed more efficiently by combining trust mechanisms with user models since user models can store the information about users' needs and preferences and trust mechanisms can help to identify trustworthy referees.

Figure 1 Bridging trust mechanisms with user modelling (see online version for colours)



To summarise, the two research areas trust and reputation and user modelling have been developing independently so far and the approaches are somewhat complementary (see Figure 1). Traditional trust models (with some exceptions, e.g., Wang and Vassileva, 2003a) are typically simple and use single trust values that one agent/user holds about another one. They do not consider differentiated trust values on the recommendations received. Also, most approaches do not utilise information about users, for example, about their preferences. In contrast, user modelling has focused on modelling features of users, including user preferences, and has used these models to generate recommendations, but typically this is done within centralised architectures and without consideration of the fact that different sources of user data can vary in their reliability. More recently, decentralised user modelling approaches have emerged, which allow to share user models among different applications, but this is done without considering that applications may not be equally trusted. There is an evident gap between the research areas of trust and reputation mechanisms and decentralised user modelling. Augmenting trust mechanisms with user modelling concepts has the potential of improving service/expert recommendation systems. This paper presents an approach bridging these two areas. It uses explicit models of user preferences in combination with a differentiated trust mechanism in a decentralised environment to recommend service providers to users, thus.

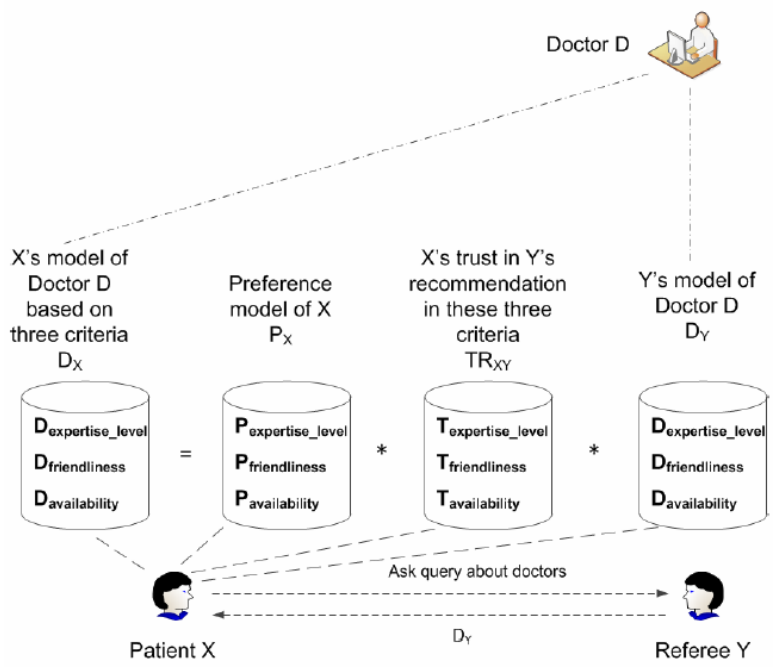
3 A differentiated trust model in referees

The main assumption underlying the approach is that the process of finding the right expert/service provider in a social network needs to take into account explicitly the user's needs and preferences (i.e., it is a content-based, rather than a collaborative recommender approach). As an example of a scenario for service recommendation, consider patients (users) looking for good doctors. The following assumptions are needed for the model.

Users rate doctors (service providers, in general) according to a finite set of aspects (we will use the term 'criteria' along with 'aspects' in the remainder of the paper); the ratings express the users' trust in the server provider. In the example scenario the patients rate doctors according to three criteria: the doctor's expertise level, how approachable/friendly the doctor is, and the doctor's availability. The users participate in a social network. Each user has a set of neighbours (friends) in the network. Friends act as referees and help each other by sharing information (references) about service providers. Each reference contains the referee's trust in each of the aspects of the service provider. However, the user who is receiving this information may not have the same level of trust in the referee for judging all the aspects. Again, not all the criteria have the same weight in making the decision. A user may give more preference weight to the doctor's 'availability' than to the other aspects.

After receiving recommendations about a service provider from his neighbours, a user (the user's agent) creates its own model of this service provider by considering its own preference model, its differentiated trust in the referee's ability to evaluate each criterion and the referee's model of the service provider (the referee's rating of the provider regarding each criterion). Suppose two agents X and Y represent two users who are friends or are connected in a social network. X queries Y about its models of doctors. Y has a model of one doctor D. The model contains three values, reflecting Y's trust in D with respect to the three different criteria mentioned above (see Figure 2).

Figure 2 Trust and preference models between two agents (see online version for colours)



After being queried, Y sends its model D_Y of doctor D, back to X. X has a differentiated trust model in Y's recommendation, TR_{XY}. X's preference model, P_X has different preference weights in the three aspects/criteria for judging doctors. Taking all these into account, X creates its own model of D by computing (1).

$$D_X = D_Y * TR_{XY} * P_X \tag{1}$$

Here, '*' denotes element-by-element multiplication of the models, i.e., the i^{th} element of D_Y is multiplied by i^{th} element of both TR_{XY} and P_X and the same goes for other elements of the models.

If there are more than one friends of X (referees) who know or have a model of doctor D, X will calculate D_X by taking the average of all their recommendations about D. Therefore, if 'Δ' is used to denote the set of all agents who provide references about the doctor 'D'; and $\| \Delta \|$ denotes the cardinality of this set, then X's model of D is calculated using (2).

$$D_X = \Sigma_{Y \in \Delta} \left(D_Y * TR_{XY} * PX \right) / \| \Delta \| . \tag{2}$$

More details about the approach and a preliminary evaluation results in a small social network can be found in Nusrat and Vassileva (2012).

4 Simulation design

In order to evaluate the approach, we developed a simulation. The objective of the experiment, input configuration and computational steps are described below.

4.1 Objective

The focus of the experiments is to see how effectively the trust-based recommender system helps an agent find an appropriate expert matching its preferences. We expect that the average performance of the agents in the system (in terms of their average satisfaction with their choices of experts) will increase with the number of queries.

4.2 Simulation language

Erlang Simulation Language (2012) is chosen as implementation language of the simulation because it is a functional language that is well-suited to building large-scale distributed systems, while allowing a full control over the design of agent functionality and interactions. This is not easily achieved in existing multi-agent simulation environments (e.g., AnyLogic). The runtime system of Erlang has built-in support for concurrency, distribution and fault-tolerance.

4.3 Input parameters

Each user is represented in the simulation by an agent that has a set of differentiated trust models K in different doctors and a set of preferences P.

The service providers are characterised with three features, according to the patient-doctor scenario discussed above. While there are certainly many other features that can be included, for the purpose of the evaluation, it is sufficient to consider these three. Agents maintain differentiated trust models (e.g., user models) of doctors consisting of trust values for each of the three aspects:

1 expertise level of the doctor

2 how approachable the doctor is (friendliness)

3 availability of the doctor

If K is a differentiated model representing a doctor's actual level in these three criteria, then,

$$K = \{K_1, K_2, K_3\}$$

Here, K_1, K_2 and K_3 respectively denote doctor's level of expertise, friendliness and availability and these values range from 0.1 to 1.0.

P represents a user's preferences with respect to the three features of doctors:

$$P = \{P_1, P_2, P_3\}.$$

The values for P_1 and P_3 range from 0.1 to 1.0, whereas value of P_2 range from 0.1 to 0.5 only, based on a-priori assumption that patients will give more priority on the doctor's expertise and availability than on the doctor's friendliness.

Each agent has also differentiated trust in other agents (neighbours in the social network representing the user's friends) that are part of its social network and are therefore potential referees. These trust values indicate how much the user trusts the referee to provide fair judgment with respect to the different features. These trust values also range from 0.1 to 1.0.

4.4 Performance metric

The best possible doctor for an agent is found by computing a score for each doctor according to the agent's preferences and sorting the scores. If agent X knew objectively (i.e., from an oracle) the actual abilities of a doctor d, represented in his model K defined above, the score (A) calculated by the agent for this doctor d is defined as:

$$A_x^d = \Sigma_i K_i * P_i, \text{ where } i \in \{1, 2, 3\} \quad (3)$$

These 'ideal' scores are computed for all the agents at the beginning of the simulation and the matches with the best possible scores are identified. This is only necessary in the simulation in order to identify an overall measure of 'goodness' of every possible match and thus, to be able to evaluate if agents are improving their ability to find good matches by interacting with their neighbours. In a real scenario of course this is not possible since the agents do not know the real capabilities of the doctors, K, and calculate their scores using the values of trust T in the doctors, supplied by their friends. These values can be considered as approximations (subjective evaluations) of the real capabilities of the doctor.

We want to find whether our model allows agents to gradually find more suitable doctors, even though they do not know the real capabilities of the doctors.

With each request it sends, the agent learns more about doctors from its friends/referees, and finds a better doctor according to its preferences. At any point of time, the percentage of satisfaction (P_{st}) for an agent is determined by equation (4).

$$P_{st} = \left(A_{current} / A_{best} \right) * 100 \quad (4)$$

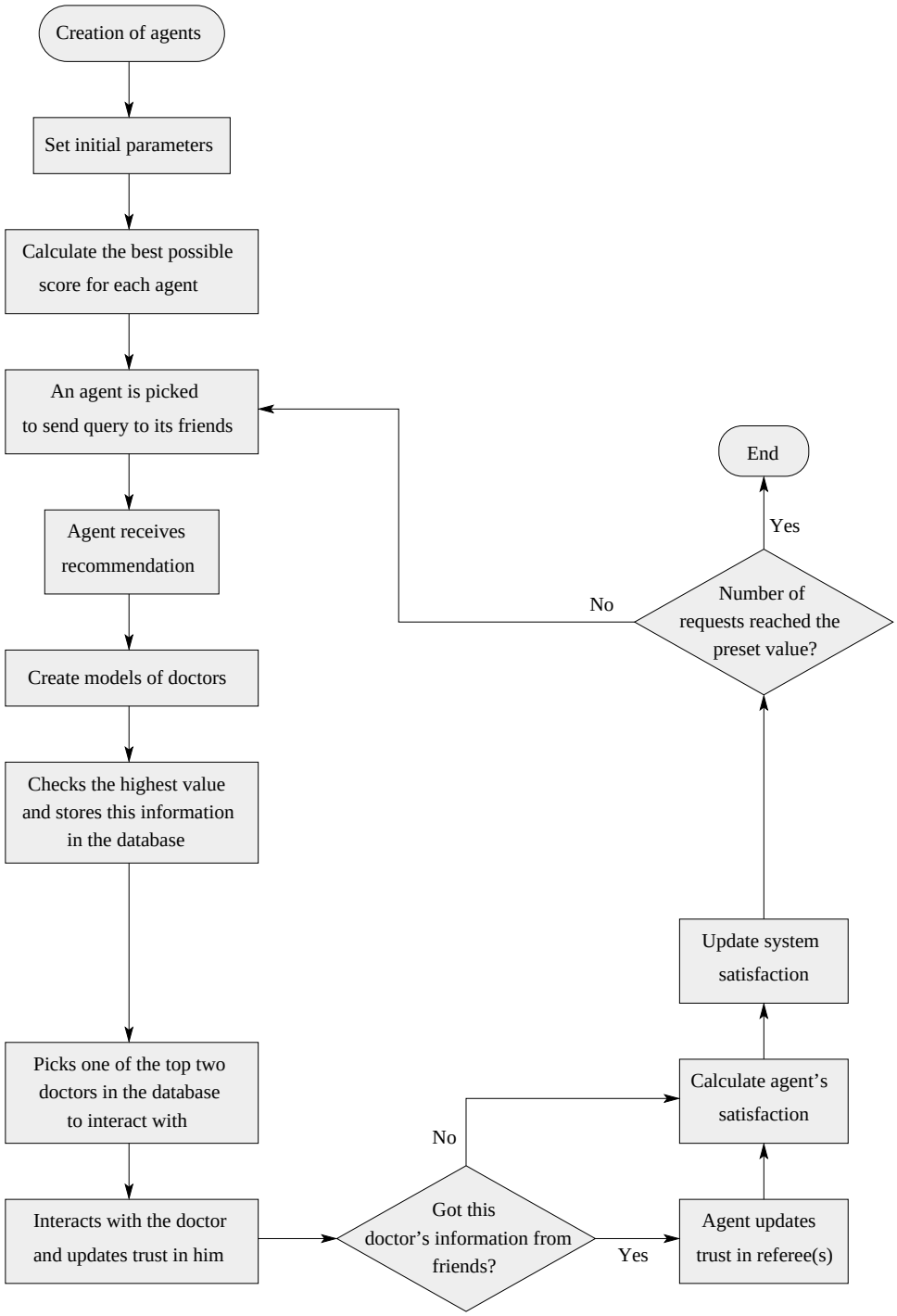
Here, $A_{current}$ represents the current score for present choice of doctor (described later) and A_{best} represents best score achievable for the patients in the set of all doctors in the system. A_{best} can be calculated in the simulation, but of course will not be known in the real world as there is no centralised user modelling system collecting all user preferences and doctors' features measured 'objectively'.

An increase of P_{st} indicates that the user is getting closer to the best possible match in the network. As more requests are generated in the simulation, we observe how close the agents get to their best matches.

The experiments were run using the following workflow (see Figure 3).

- *Defining the optimality criterion:* In the first step of the simulation, the best match for each of the agents in the system is calculated. This indicates the agent's optimal choice of service provider (doctor) from the entire pool of service providers in the simulation. In order to find the best match, all possible scores of the agent are calculated using equation (3). Then the service provider with the highest score is identified. This is the agent's best choice.
- *Creation of agents:* A separate Erlang process was created for each of the agents in the model. These processes (agents) were distributed in different Erlang runtime systems for the sake of scalability. The runtime systems were connected with each other and an agent could communicate with any other agent through the Erlang's message passing feature.
- *Query processing:* Each agent, after being created, waits to send messages or to process information. An agent picked randomly queries its friends about doctors.

Figure 3 Flowchart showing the steps in the simulation



- *Processing doctor's information:* After receiving the recommendation about a doctor from its friends, an agent creates its model of the doctor according to its trust and preference models by computing equation (2). If it is the first reference about this doctor then this data is stored in database of the agent. If there are existing models about the same doctor, then the old models are combined with the new model, the average of the corresponding values is calculated and stored in the database.

  Figure 4 shows a scenario where agent $a1$ asks its friends about their models of doctors and receive information about doctor $d21$, $d31$, $d32$, $d41$, $d42$ and $d43$. After getting all these recommendations, agent $a1$ checks which doctor has the highest trust value and also which referee provided this information and stores this information in its own database. Suppose the trust value of doctor $d42$ was the highest among all the references agent $a1$ received. Then doctor $d42$'s information will be saved in a1's database ($d42_{a1}$). This is shown in Figure 5.

Figure 4 Agents with doctor's information in their own databases (see online version for colours)

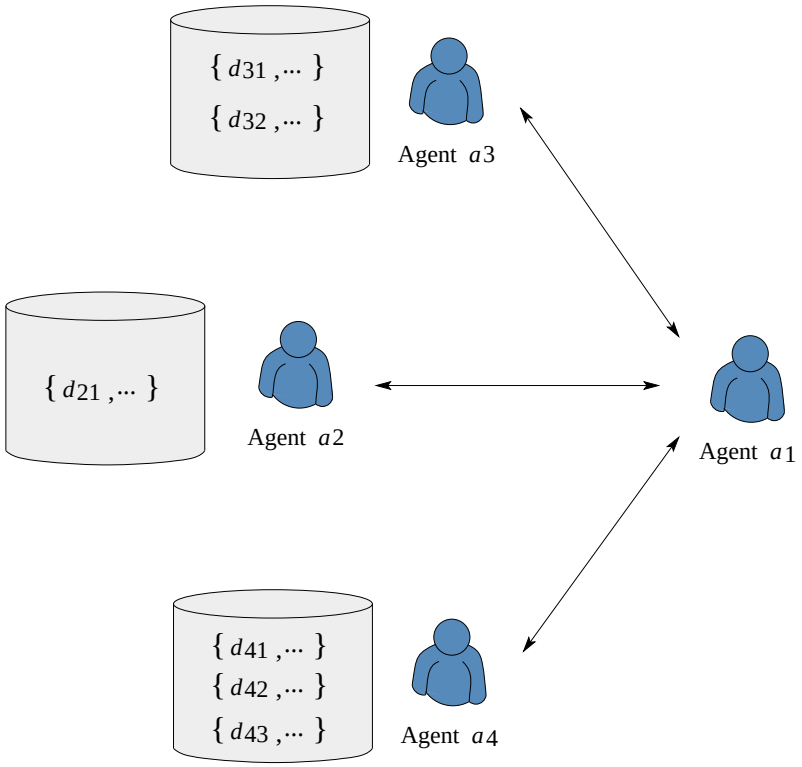
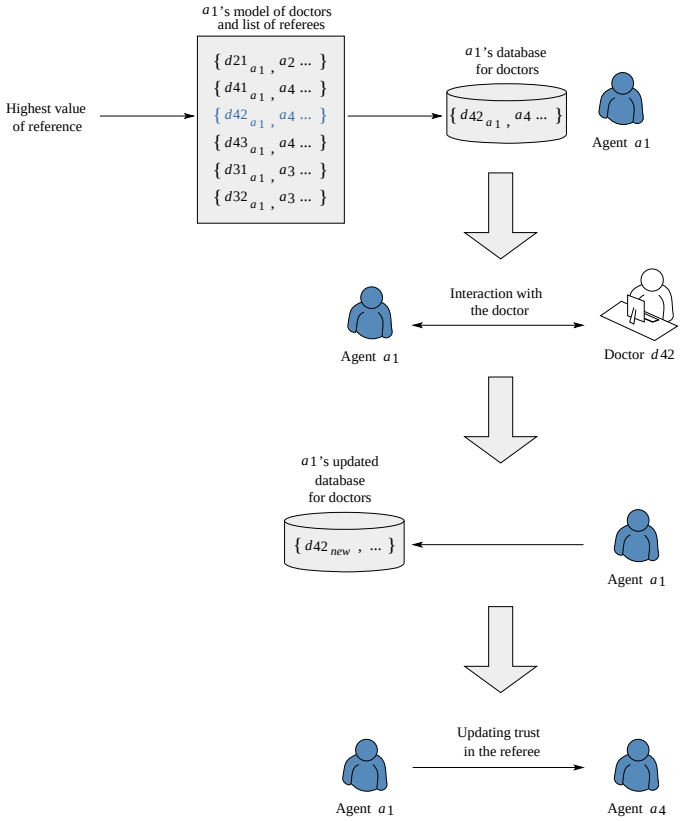


Figure 5 Storing doctor's recommendation and updating trust in the referee (see online version for colours)



- *Choosing the doctor to interact with:* After all the above steps are done, the agent has information about at least one doctor in its database. However, it may have other doctors' information as well from previous interactions. Therefore, it sorts all the models of doctors based on the trust in them and picks one randomly from the top two to interact with. This is done in order to allow for exploration, and avoid picking the top-trusted doctor all the time. If a user interacts directly with a doctor, her trust in the doctor will usually have a higher value than if she only got recommendation about the doctor from friends. But the recommended doctor may actually be better suited for the agent. Picking randomly one of the two top-trusted doctors allows exploring from time to time the unknown but highly recommended doctors.
- *Updating trust in the doctor:* Before interacting with the chosen doctor, the agent had some trust in the doctor based on either
  1 recommendation from its friends
  2 its previous direct interaction with the doctor.

After the interaction the agent will have to update this value based on its satisfaction with the interaction. In the first case, the agent will discard the previous recommendation value in the doctor and store the new value of trust after its own direct interaction. In the second case, where the agent's old trust value in the doctor was based on direct interaction it will combine the previous trust with the trust value generated recently based on the evidence from the referees and it will calculate a new trust value using reinforcement learning formula (4):

$$t_{new} = t_{old} * \alpha + (1 - \alpha) * evidence \tag{5}$$

Here, α is the learning rate of the agent and evidence denotes the interaction feedback (in the simulation this value is skewed towards doctor's skill level). For example, if the doctor's skill level on some criteria is 0.6 (in a scale between 0.1 and 1), after interacting with the doctor, the evidence will be a random number between 0.4 and 0.8.

- *Updating trust in referees:* After interacting with the doctor and updating its trust in the doctor, the agent updates its trust in the referees who gave recommendations about this doctor (Figure 5). If the agent's new model of the doctor matches closely with that of referee, then agent's trust in referee will increase, otherwise, it decreases. This new trust value is stored for future reference.
- *Calculating the satisfaction of the agent:* If R is a differentiated model to represent trust in the doctor after interaction and P represents the agent's preferences, then the score (A) of the agent for the current choice of doctor is calculated as follows:

$$A_{current} = \Sigma_i R_i * P_i, \text{ where } i \in \{1, 2, 3\} \tag{6}$$

After that, the satisfaction of the agent is calculated by equation (4). As more requests are generated in the simulation, we expect to see how close the agents get to their best matches (computed in the first step of the simulation).

All the above steps are done after each simulated request by an agent for finding a doctor. After that, the system's average satisfaction is calculated by averaging the satisfactions of all the agents in the system. The hypothesis is that the system satisfaction increases with the number of requests being processed (i.e., agents interacting with each other and exchanging references about doctors collectively learn to find better suited doctors to their preferences). To test this hypothesis, we experimented on a large scale real social network. The results of the experiments and analysis are presented in the next section.

5 Experiments and analysis

5.1 Network data

The social network graph used in the simulation is from the 'Wikipedia Vote Network', a large scale social network dataset available from Stanford Network Analysis Platform (SNAP, 2012). The nodes in the network denote Wikipedia users and a directed edge

from node *i* to node *j* represents that user *i* voted on user *j*. There were 7,115 nodes in the directed graph. The network topology is static during the simulation, i.e., agents do not make new friends or lose friends. While this assumption may not be realistic, modelling a dynamic network would have introduced too much complexity in the simulation and would have obscured the effect of our model and mechanism that we aim to evaluate.

5.2 Simulation setup

The experiments use an Intel(R) Xeon(R) machine with two 2.33 GHz processors and 16 GB RAM.

At first, the input graph of 7,115 agents is fed into the simulation. Our experiments involve a population with eight doctors (service providers). This was done to match real world data, since there are 1.03 general/family physicians per 1,000 people according Statistics Canada (2012).

The initial trust values between any two agents, the trust values of agents in various doctors and the preference weights over the quality of service criteria for all the agents were preset and were read from a file at the start of the simulation to ensure that the experiments can be repeated. These values, between 0.1 and 1, were created in Erlang random number generator.

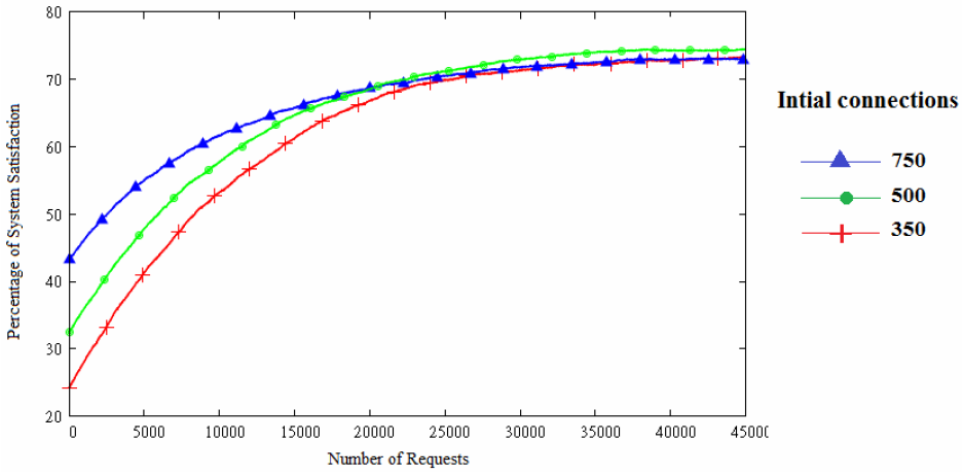
5.3 Initial distribution of patients among doctors

As we were not able to find any real world statistical data about the distribution of patients over doctors, to adjust appropriately the network topology when inserting the doctors in the network, we used anecdotal evidence. An estimate (around 500 patients per family doctor) was given independently from two healthcare workers we were acquainted with. A normal distribution of the number of connections over the eight doctors was used since this is one of the most commonly found distributions among statistical and natural phenomena. Three experimental setups were evaluated which used normal distribution of patients for each of the eight doctors with three different means 350 patients, 500 patients and 750 patients.

5.4 Results

The goal of our experiment was to see whether using our approach of recommending services in a trust-based network helps agents to achieve better satisfaction. The results are plotted in Figure 6. The result of the setup with initial mean connection of 350 is drawn with a red line and '+' symbol in every 2,000th point. The green line with '●' symbol in every 2,000th point indicates the result the setup with initial mean doctor-patient connection of 500 and the blue line with '▲' symbol shows the result of the setup with initial mean connection of 750. In Figure 6, we can see the evolution of the overall level of satisfaction for all agents in the system with the number of requests for all three setups.

Figure 6 Percentage of system satisfaction vs. number of requests for the three different initial setups of 'Wikipedia Vote Network' (see online version for colours)



The purpose of the evaluation is to check how close the matches found with this approach fit the agents' preferences and whether agents gradually increase their satisfaction with their interactions. From Figure 6, we can see that the system satisfaction increases very quickly with the number of requests, until the satisfaction converges to a final value. For the setup with 500 median initial connections per doctor, the system satisfaction is around 32% in the beginning of the simulation and it grows quickly until 35,000 requests (among all agents in the system), then it converges slowly to around 75%. For the setups with 350 and 750 median initial connections per doctor, the initial satisfaction values are 24% and 43% respectively, and both these curves converge to the satisfaction value of around 74%. Therefore, even with different initial configurations, in all experimental settings the system reaches a convergence value of system satisfaction about 75%.

5.5 Analysis

As more requests are generated in the system, some relations become more trusted, and some less-trusted. If an agent is satisfied with a doctor, its trust values in the referees who provided references about this doctor also increase, otherwise they decrease. Therefore, the trust in friends who have different preference criteria decreases with time and has less effect in decision making. As agents update their trust models with each request, they learn to find better matches according to their preferences, and their level of connectedness. The system satisfaction does not reach 100%, as the simulation uses a very large and realistic social network. An agent 'on one side of the network' may not get information about a well matching doctor who is connected with the agents 'on the other side' of the network. As we are checking how close an agent gets to the best possible doctor in a large network and as the trustworthiness of a message decreases with the number of hops propagated, the recommendation value of that doctor might be too low for the agent to actually choose to interact with the doctor.

Another interesting point is, even after starting with a lower initial satisfaction value (32%), the final satisfaction value (74.5%) for the setup with 500 connections median is slightly higher than that of the setup with 750 connections median (73.3%).

Our explanation for this is based on two important features of our model: an agent considers one of the two top-trusted doctors to interact with, and the trust of an agent in a doctor known from direct interaction is usually higher than any recommendation value. Suppose, for agent X doctor $d1$ is better than doctor $d2$.

Consider two different experiments. For the first experiment, at the time of request $n - 1$, the agent has only information about $d2$ in its database and at the time of request n it gets a recommendation about $d1$. X may choose to interact with $d2$ again, not taking the recommendation into consideration. Therefore, the satisfaction of agent X increases slightly after run n, resulting in very small increase in the overall satisfaction.

In experiment two, the agent may not have any information about doctors at the time of request $n - 1$. The system satisfaction in this experiment at the time of this request is lower than in experiment one, where agent x had some satisfaction value, contributing in the overall system satisfaction. However, after the agent gets a recommendation about doctor $d1$ in run n, it chooses to interact with this doctor and gets a higher satisfaction from him than it could get from doctor $d2$. This contributes to improving the overall system satisfaction. Therefore, even after starting with a lower system satisfaction, the rate of increase can be higher depending on the information an agent has and the decision it makes. The growing curve for system satisfaction implies that agents collectively are able to find good matches for themselves in a trust-based social network without the help of any central server.

6 Evaluating robustness of the approach

In a real network there can be malicious agents who try to mislead other agents by providing unfair ratings. A good trust and reputation approach, or a level 2 approach, according to the classification in Sabater and Sierra (2005) should be able to allow agents to quickly identify these undesirable agents and avoid them, choosing more trustworthy referees. In order to check the robustness of our approach against malicious agents, an experiment is conducted, introducing some malicious agents in the system.

6.1 Main idea

The main idea of this experiment is to test whether the reputation of malicious agents decreases with the number of interactions in the system. 'Reputation' is defined as the average trust that all contacts of a particular agent have in it, i.e., it is an aggregation of their trust values in that agent. A certain number of malicious agents are introduced randomly in the network, who always provide top trust values for a particular doctor, whose capabilities are low. In this way, a shilling attack by malicious agents is simulated who pretend to be objective in their references but aim to promote a particular bad service provider. The hypothesis is that as new requests are generated, and agents share trust information among each other, the reputation of the malicious agents will decrease, whereas the average reputation of the honest referees will remain the same or increase.

6.2 Experimental setting and measure

To simplify the calculation of reputation and the analysis of the experimental results, we revert back to non-differentiated trust models, by summing up the three trust values of the

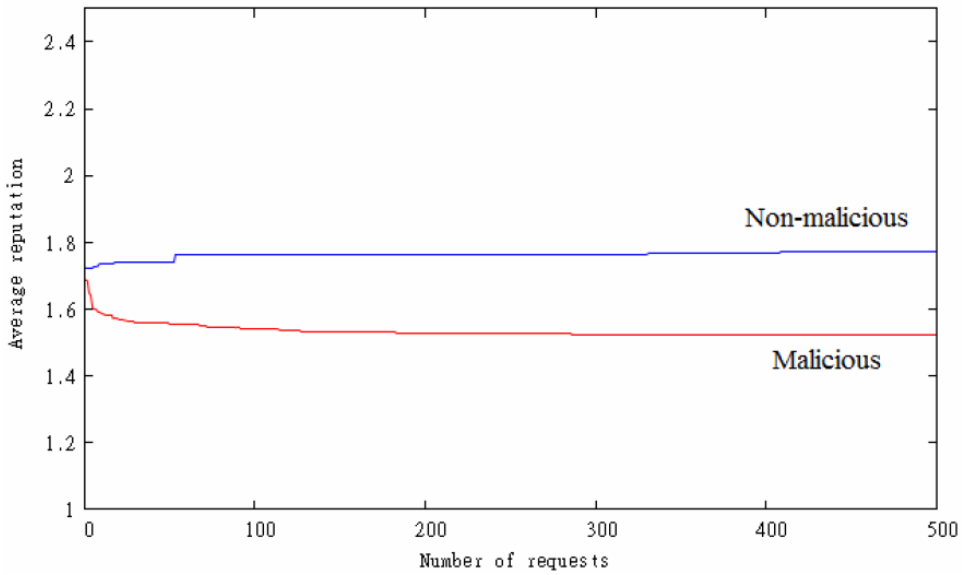
differentiated trust model that each agent X has in its neighbour Y, thus creating a general (non-differentiated) trust value $TG_{XY} = TR^1_{XY} + TR^2_{XY} + TR^3_{XY}$, where TR^i_{XY} is X's trust in Y as a referee about aspect i. TG_{XY} can take values in the interval [0.3, 3]. This is because in our model, the highest level of trust that an agent can have in another as a referee is 1 and the lowest level of trust can be 0.1 in each of the three criteria, so the sum of the three values can vary between 0.3 and 3. Therefore, the initial reputation of the agents can be a number between 0.3 and 3. The general trust values by all neighbours of each agent are aggregated by averaging them, and in this way the reputation of each agent is calculated. This is equivalent to calculating agent reputation by a central entity or reputation service similar to the 'better business bureau'.

The simulation compares the reputation of the group of malicious agents with the reputation of the group consisting of all other agents who provide honest recommendations. The reputation of a group is calculated by summing up the reputation of all agents in a group and dividing it by the number of agents. There were 15 malicious agents introduced in the system. Since the reputation of an agent is a number between 0.3 and 3.0, the average reputation of 15 agents is also a number in the range [0.3, 3.0]. The objective of the experiment presented in this section is to check how the average reputation of malicious agents changes with the growing number of interactions in the simulation.

The malicious agents provide dishonest recommendations about one particular bad doctor (suppose, 'B'), giving very high trust values in all three criteria, although this doctor's actual skill levels are very low. As the friends of these malicious agents query them about doctors, the malicious agents provide dishonest recommendations. We assume also that, initially, only the malicious agents know about the bad doctor B. So all the recommendations received about B at first will be dishonest (i.e., with artificially high trust values). In the course of the simulation, some of the non-malicious agents will select this doctor and will interact with him, resulting in dissatisfaction. This agent will develop its own model of B, and it will share its model with its friends when asked for recommendations. In this way, truthful references about B will start spreading in the system and competing with the dishonest references provided by the malicious agents. We want to check how quickly the friends (neighbour nodes) of the malicious agents will decrease their trust in them, i.e., how quickly the reputation of the malicious agents will drop.

To have a baseline for comparison, we run the simulation with the same setup, but without malicious agents, i.e., all agents are truthful in giving recommendations. The same agents who are friends of the malicious 15 agents (but in the baseline setup these 15 agents are truthful) query about doctors and receive recommendations. Figure 7 shows the results of both simulations for 500 queries. The average reputation of good agents is drawn with a blue line and the average reputation of bad agents is drawn with a red line.

Figure 7 Comparison between the reputation of good and bad agents (see online version for colours)



6.3 Analysis

According to our trust model, after receiving recommendation from a referee, an agent increases its trust in the referee, if it finds the recommendation useful; otherwise it decreases its trust in the referee. This updated trust value is stored in the agent's database and it determines how future references from this referee agent will be evaluated. If the agents are satisfied with the recommendations received from these referee agents, their reputation will increase (in case of positive experience with a doctor chosen based on the reference) or, at least, will stay the same (in case the agent did not choose the doctor about which it received reference and thus has no direct evidence).

As we can see from Figure 7, the average reputations of the group of good agents and the group of malicious agents start with almost the same value. But, within the first few interactions in the system, the average reputation of the malicious agents group decreases, and then it stays steady at a lower value. The stabilisation of the reputation score implies that the recommendations from the malicious agents are not considered anymore and nobody chooses doctor *B* after a certain number of requests. Therefore, the trust values in these malicious agents cannot change anymore. Hence, the bad agents cannot have much influence on their friends in decision making after the first approximately 25 requests. On the other side the reputation of the group of honest agents increases slightly and also stabilises at a certain level that depends on the distribution of preferences among the agents (agents with different preferences cannot trust each other even though they are honest). Consequently, only references from the trustworthy agents are considered, nullifying the effect of the malicious agents.

7 Conclusions

We have proposed and evaluated a differentiated trust model in referees that can be used to recommend suitable service providers to the users depending on their preferences. The approach is decentralised, scalable and robust with respect to shilling attacks by groups of malicious agents.

While not able to achieve the maximum possible satisfaction, which could have been achieved in a centralised system, provided that all users report truthfully their ratings of individual features of service providers and their preferences, this decentralised approach does not require agents to reveal their preferences to a centralised component, thus preserving their privacy. It is easily extendible by adding new features of service providers that can be included in the user models.

Our approach enriches three different research areas: trust and reputation mechanisms, decentralised user modelling and recommender systems. This approach has an evolutionary aspect: agents representing users in a decentralised environment will learn from user's feedback to improve their user models and the trust models in other agents representing users. Thus, starting with an existing real social network, certain relationships will become more trusted and some – less trusted, depending on the preferences of the users and the usefulness of recommendations exchanged. We are optimistic that this approach, when implemented in a real expert finding system, will prove to be efficient, effective and psychologically acceptable to users, as it exploits trusted social networks of friends to generate recommendations.

References

Adomavicius, G. and Tuzhilin, A. (2005) 'Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions', *IEEE Transactions on Knowledge and Data Engineering*, Vol. 17, No. 6, pp.734–749.

Dell'Amico, M. and Capra, L. (2008) 'Sofia: social filtering for robust recommendations', *Trust Management II*, pp.135–150.

Erlang Simulation Language (2012) available at http://www.erlang.org/ (accessed on 29 March 2012).

Fink, J. and Kobsa, A. (2000) 'A review and analysis of commercial user modeling servers for personalization on the World Wide Web', *User Modeling and User-Adapted Interaction*, Vol. 10, No. 2, pp.209–249.

Hang, C.W., Wang, Y. and Singh, M.P. (2009) 'Operators for propagating trust and their evaluation in social networks', *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems*, Vol. 2, International Foundation for Autonomous Agents and Multiagent Systems, pp.1025–1032.

Heckmann, D. and Krueger, A. (2003) 'A user modeling markup language (UserML) for ubiquitous computing', *9th International Conference on User Modeling (UM'2003)*, Springer, pp.393–397.

Kay, J., Kummerfeld, B. and Lauder, P. (2002) 'Personis: a server for user models', *Proceedings of the Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, pp.203–212.

Kobsa, A. (1992) *User Modeling and User-Adapted Interaction 1*, pp.v–viii, 1991 Kluwer Academic Publishers, The Netherlands.

Matsuo, Y. and Yamamoto, H. (2009) 'Community gravity: measuring bidirectional effects by trust and rating on online social networks', *Proceedings of the 18th International Conference on World Wide Web*, ACM, pp.751–760.

Niu, X., McCalla, G. and Vassileva, J. (2004) 'Purpose-based expert finding in a portfolio management system', *Computational Intelligence*, Vol. 20, No. 4, pp.548–561.

Nusrat, S. and Vassileva, J. (2012) 'Recommending services in a trust-based decentralized user modeling system', *Advances in User Modeling*, Revised selected papers from *UMAP 2011 Workshops*, Springer Verlag, LNCS7138, pp.230–242.

Sabater, J. and Sierra, C. (2005) 'Review on computational trust and reputation models', *Artificial Intelligence Review*, Vol. 24, No. 1, pp.33–60.

Shoham, Y. and Balabanovic, M. (1997) 'Fab: content-based, collaborative recommendation', *Communications of the ACM*, Vol. 40, No. 3, pp.66–72.

SNAP (2012) available at http://snap.stanford.edu/data/ (accessed on 29 March 2012).

Statistics Canada (2012) available at http://www.statcan.gc.ca (accessed on 29 March 2012).

Vassileva, J., McCalla, G. and Greer, J. (2003) 'Multi-agent multi-user modelling in I-Help', *User Modeling and User-Adapted Interaction*, Vol. 13, No. 1, pp.179–210.

Wang, Y. and Vassileva, J. (2003a) 'Bayesian network-based trust model', *Proceedings of the IEEE/WIC International Conference on Web Intelligence, 2003 (WI 2003)*, IEEE, pp.372–378.

Wang, Y. and Vassileva, J. (2003b) 'Trust and reputation model in peer-to-peer networks', *Proceedings of the Third International Conference on Peer-to-Peer Computing (P2P2003)*, September, pp.150–157.

Wang, Y. and Vassileva, J. (2007) 'Toward trust and reputation based web service selection: a survey', *International Transactions on Systems Science and Applications*, Vol. 3, No. 2, pp.118–132.

Yimam-Seid, D. and Kobsa, A. (2003) 'Expert-finding systems for organizations: problem and domain analysis and the DEMOIR approach', *Journal of Organizational Computing and Electronic Commerce*, Vol. 13, No. 1, pp.1–24.

Yu, B. and Singh, M. (2000) 'A social mechanism of reputation management in electronic communities', *Cooperative Information Agents IV – The Future of Information Agents in Cyberspace*, Vol. 1860, pp.355–393, Springer Berlin/Heidelberg.