Bayesian Network-Based Trust Model

Yao Wang, Julita Vassileva
University of Saskatchewan, Computer Science Department,
Saskatoon, Saskatchewan, S7N 5A9, Canada
{yaw181, jiv}@cs.usask.ca

Abstract

We review trust and reputation mechanisms applied in centralized and decentralized systems. Then we propose a Bayesian network-based trust model. Since trust is multi-faceted, even in the same context, agents still need to develop differentiated trust in different aspects of other agents' behaviors. The agent's needs are different in different situations. Depending on the situation, an agent may need to consider its trust in a specific aspect of another agent's capability or in a combination of multiple aspects. Bayesian networks provide a flexible method to present differentiated trust and combine different aspects of trust. A Bayesian network-based trust model is presented for a file sharing peer-to-peer application.

1. Introduction

Large distributed systems applied in the areas of e-commerce, web-services, distributed computing, and file sharing peer-to-peer (p2p) systems, consist of autonomous and heterogeneous agents, which behave on the behalf of users. Usually, agents play two roles, the role of service providers (sellers, servers) and the role of consumers (buyers, clients). Since agents are heterogeneous, some agents might be benevolent and provide high-quality services, others might be buggy and unable to provide high-quality services, and some might be even malicious by providing bad services or harming the consumers. Since there is no centralized node to serve as an authority to supervise agents' behaviors and punish agents that behave badly, malicious agents have an incentive to harm other agents to get more benefit because they can get away. Some traditional security techniques, such as service providers requiring access authorization, or consumers requiring server authentication, are used as protection from known malicious agents. However, they cannot prevent from agents providing variable-quality service, or agents that are unknown. Mechanisms for trust and reputation can be used to help agents distinguish good from bad partners. This paper describes a trust and reputation mechanism that allows agents to discover partners who meet their individual requirements, through individual experience and sharing experiences with other agents with similar preferences.

The rest of this paper is organized as follows: section 2 discusses some related issues about trust and reputation. Section 3 introduces our approach to developing a Bayesian network-based trust model. The experiment design and results are presented in Sections 4 and 5. Section 6 discusses related work on trust and reputation. In the last section, we present conclusions and directions for future work.

2. Trust and Reputation

Trust and reputation mechanisms have been proposed for large open environments in e-commerce, distributed computing, recommender systems. Agents are often used to manage and reason about trust and reputation. In this situation, trust is defined as an agent's belief in attributes such as reliability, honesty and competence of the trusted agent. The reputation of an agent defines an expectation about its behavior, which is based on other agents' observations or information about the agent's past behavior within a specific context at a given time. Suppose there are two agents, agent A and agent B. When agent A has no direct interaction with agent B or it is not sure about the trustworthiness of B, agent A can make decisions relying on the reputation of agent B (obtained through asking other agents). Once agent A has interactions with agent B, it can develop its trust in agent B according to its degree of satisfaction with the interactions and use this trust to make decisions for future interactions.

Some of the literature on trust and reputation treats the two concepts interchangeably or ambiguously, which sometimes causes confusion [12, 16]. The two concepts are related, but different. Agent A's trust in agent B is the accumulation of evaluations that agent A has of its past interactions with B. It reflects agent A's subjective viewpoint of B's capability. The reputation of agent B is an objective measure for agent B's capability, resulting from the evaluations of many other agents. There are two ways for agent A to learn agent B's reputation. One is to ask an authority, like a "better business bureau", which is responsible for accumulating evaluations of agents (including B) from other agents and calculating the reputation of the evaluated agents (e.g. B) based on these evaluations. In this case, the authority usually does not care who provides an evaluation, an honest agent or dishonest agent, but relies on the amount of data to make the effect of possible biased evaluations in-

significant. Centralized systems, such as eBay and onSale, use this way of building reputation.

The other way for A to obtain agent B's reputation is to proactively request and collect other agents' evaluations about B and to combine the evaluations together to form its own view of B' reputation. This way of computing reputation is adopted in decentralized systems [5].

Trust can be broadly categorized by the relationships between the two involved agents in the following categories [7].

- *Trust between a user and her agent(s).*

Although an agent behaves on its user's behalf, an agent might not act as its user expects. How much a user trusts her agent determines how she delegates her tasks to the agent [18].

- *Trust in service providers.*

It measures whether a service provider can provide trustworthy services.

- *Trust in references.*

References refer to the agents that make recommendations or share their trust values. It measures whether an agent can provide reliable recommendations.

- *Trust in groups.*

It is the trust that one agent has in a group of other agents. By modeling trust in different groups, an agent can decide to join a group that can bring it most benefit [19]. Hales [8] points that group reputation can be a powerful mechanism for the promotion of beneficent norms under the right condition. This kind of trust is also useful in helping an agent judge the other agent according to its trust in the group that the other agent belongs to [6, 8, 12, 19].

2.1 Centralized vs. Decentralized

Trust and reputation mechanisms have been implemented in many systems adopting either a centralized structure or a decentralized structure. Accordingly, the trust and reputation mechanisms used in the two kinds of systems are also different.

In centralized systems, such as in eBay and onSale, which are mainly seen in the area of e-commerce, the trust and reputation mechanisms used are relatively simple. There are some common characteristics in these systems.

- A centralized node acts as the system manager responsible for collecting ratings from both sides involved in an interaction.
- Agents' reputations are public and global. The reputation of an agent is visible to all the other agents.
- Agents' reputations are built by the system. There is no explicit trust model between agents.
- Less communication is required between agents. An agent only communicates with the centralized node to know other agents' reputations.

Despite of the simplicity of the centralized reputation, empirical results show these systems do encourage transactions between sells and buyers. But there are some problems. Agents are usually reluctant to give negative ratings because they can see each other's ratings and are afraid of revenges [15]. Another problem is that if an agent has a bad reputation, it can discard its old identity, choose a new one, start as a beginner and get rid of its poor reputation. The third problem is that agents can increase their reputations artificially by creating fake identities and having them to give themselves high ratings [24].

The trust and reputation mechanisms used in decentralized systems, for example, peer-to-peer networks, are more complex than those applied in centralized systems. They have the following characteristics [2, 3, 5]:

- There is no centralized system manager to govern trust and reputation.
- Subjective trust is explicitly developed by each agent. Each agent is responsible for developing its own trust in other agents based on their direct interactions.
- No global or public reputation exists. If agent A wants to know agent B's reputation, it has to proactively ask other agents for their evaluations of B, then synthesize the ratings together to compute agent B's reputation. The reputation of agent B developed by A is personalized because agent A can choose which agents it will ask for evaluations of B, its trustworthy friends or all known agents. Agent A can also decide how to combine the collected evaluations together to get agent B's reputation. For example, it can only combine the evaluations coming from trusted agents. Or it can weight differently the evaluations from trusted agents, unknown agents and even untrustworthy agents when it combines them together.
- A lot of communication is required between agents to exchange their evaluations.

In decentralized systems, agent A can get agent B's reputation based on its own knowledge of the truthfulness of agents that make recommendations for agent B. So it is difficult for agent B to increase its reputation artificially. Since only agent A can see the recommendations, the references can express their feelings truthfully, not worried about potential revenges. But the tradeoff is that agents have to conduct a lot of communication and computation.

3. Bayesian Network-Based Trust Model

Most current applications and experiments on trust and reputation only focus on one of them, either trust or reputation, although the idea of combining them together in one system has been well known in the literature [4, 21, 22, 23]. An agent broadly builds two kinds of trust in another agent. One is the trust in another agent's competence in providing services. The other is the trust in another agent's reliability in providing recommendations about other agents. Here the reliability includes two aspects: whether the agent is truthful in telling its information and whether the agent is trustworthy or not. Since agents are heterogeneous, they judge other agent's behaviour by different criteria. If their criteria are similar, one agent can trust another

agent. If their criteria are different, they cannot trust each other even if both of them tell the truth. In the implementation of such a system based on trust and reputation, some issues have to be considered.

1) How does an agent model its user? The user ultimately sets the criteria by which an agent evaluates other agents. Each user has different preferences and ways of judging the quality of interaction. In order to behave as its user wants, an agent has to keep learning its user's preferences and behaviors. If an agent fails to do as what its user expects, it will be useless.

2) How is an interaction to be evaluated? Trust is built based on an agent's direct interactions with other agents. For each interaction, an agent's degree of satisfaction of the interaction will directly influence its trust in the other agent involved in the interaction. Usually, an interaction has multiple aspects and can be judged from different points of view.

3) How does an agent represent and update its trust in another agent?

4) When will an agent ask for recommendations about another agent that it intends to interact with?

5) How does an agent combine together the recommendations for a given agent coming from different references? Since the recommendations might come from trusted agents, non-trusted agents or strangers, an agent has to decide how to deal with them.

6) How does an agent decide if another agent is trustworthy to interact with or not, according to its direct experiences or from the agent's reputation, or both?

7) How does an agent develop and update its trust in a reference agent that makes recommendations?

8) How many kinds of trust does an agent need to develop with another agent in a single context? Agents may need to develop multiple trust relationships with each other in order to evaluate each other from different perspectives and take different aspects of the agent's behavior into account. For example, agent A might trust agent B in providing music files with good quality. But agent A might not trust agent B in offering movie files with the same quality as music files.

Our approach will deal with all the issues above except the first one, which is beyond our scope, although it is extremely important. We will use a peer-to-peer file sharing application as an example in the discussion, however the method is general and can be applied to other applications, like web-services, e-commerce, recommender systems or peer-to-peer distributed computing.

3.1 Scenario

In the area of file sharing in peer-to-peer networks, all the peers are both providers and users of shared files. Each peer plays two roles, the role of file provider offering files to other peers and the role of user using files provided by other peers. In order to distinguish the two roles of each peer, in the rest of paper, when a peer acts as a file provider, we call it file provider; otherwise, we call it simply agent. Agents will develop two kinds of trust, the trust in file providers' competence (in providing files) and the trust in other agents' reliability in making recommendations. We assume all the agents are truthful in telling their evaluations. However, the agents may have different ways of evaluating other agent's performance, which reflect different user preferences.

3.2 Trust in a File provider's Competence

In a peer-to-peer network, file providers' capabilities are not uniform. For example, some file providers may be connecting through a high-speed network, while others connect through a slow modem. Some file providers might like music, so they share a lot of music files. Some may be interested in movies and share more movies. Some may be very picky about file quality, so they only keep and share files with high quality. Therefore, the file provider's capability can be presented in various aspects, such as the download speed, file quality and file type (see Figure 1).

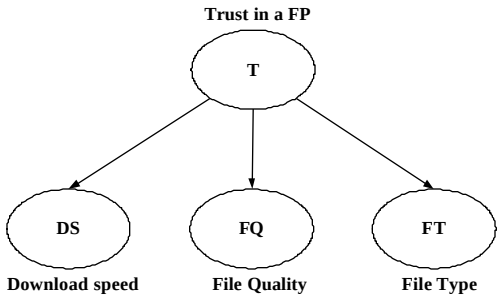

Figure 1. A Bayesian Network Model

The agent's needs are also different in different situations. Sometimes, it may want to know the file provider's overall capability. Sometimes it may only be interested in the file provider's capability in some particular aspect. For instance, an agent wants to download a music file from a file provider. At this time, knowing the file provider's capability in providing music files is more valuable for the agent than knowing the file provider's capability in providing movies. Agents also need to develop differentiated trust in file providers' capabilities. For example, the agent who wants to download a music file from the file provider cares about whether the file provider is able to provide a music file with good quality at a fast speed, which involves the file provider's capabilities in two aspects, quality and speed. How does the agent combine its two separated trusts together, the trust in the file provider's capability in providing music files with good quality and the trust in the file provider's capability in providing a fast download speed, in order to decide if the file provider is trustworthy or not? A Bayesian network provides a flexible method to solve the problem. A Bayesian network is a relationship network that uses statistic methods to represent probability relation-

ships between different agents. Its theoretical foundation is the Bayes rule [14].

$$p(h \mid e) = \frac{p(e \mid h).p(h)}{p(e)}$$

p(h) is the prior probability of hypothesis *h*; *p(e)* is the prior probability of evidence *e*; *p(h | e)* is the probability of h given *e*; *p(e | h)* is the probability of *e* given *h*.

A naïve Bayesian network is a simple Bayesian network. It is composed of a root node and several leaf nodes. We will use a naïve Bayesian network to represent the trust between an agent and a file provider.

Every agent develops a naive Bayesian network for each file provider that it has interacted with. Each Bayesian network has a root node *T*, which has two values, *"satisfying"* and *"unsatisfying"*, denoted by 1 and 0, respectively. *p(T=1)* represents the value of agent's overall trust in the file provider's competence in providing files. It is the percentage of interactions that are satisfying and measured by the number of satisfying interactions m divided by the total number of interactions *n*. *p(T = 0)* is the percentage of not satisfying interactions.

$$p(T=1) = \frac{m}{n} \qquad (1)$$

$$p(T=1) + p(T=0) = 1$$

The leaf nodes under the root node represent the file provider's capability in different aspects. Each leaf node is associated with a conditional probability table (CPT). The node, denoted by *FT*, represents the set of file types. Suppose it includes five values, *"Music"*, *"Movie"*, *"Document"*, *"Image"* and *"Software"*. Its CPT is showed in table 1. Each column follows one constraint, which corresponds to one value of the root node. The sum of values of each column is equal to 1.

Table 1. The CPT of Node FT

	T = 1	T = 0
Music	$p(FT = "Music" \mid T = 1)$	$p(FT = "Music" \mid T = 0)$
Movie	$p(FT = "Movie" \mid T = 1)$	$p(FT = "Movie" \mid T = 0)$
Document	$p(FT = "Docu" \mid T = 1)$	$p(FT = "Docu" \mid T = 0)$
Image	$p(FT = "\mathrm{Im}age" \mid T = 1)$	$p(FT = "\mathrm{Im}age" \mid T = 0)$
Software	$p(FT = "Soft" \mid T = 1)$	$p(FT = "Soft" \mid T = 0)$

$p(FT = "Music" \mid T = 1)$ is the conditional probability with the condition that an interaction is satisfying. It measures the probability that the file involved in an interaction is a music file, given the interaction is satisfying. It can be computed according to the following formula:

$$p(FT = "Music" \mid T = 1) = \frac{p(FT = "Music", T = 1)}{p(T = 1)}$$

$p(FT = "Music", T = 1)$ is the probability that interactions are satisfying and files involved are music files.

$$p(FT = "Music", T = 1) = \frac{m1}{n}$$

m1 is the number of satisfying interactions when files involved are music files .

$p(FT = "Music" \mid T = 0)$ denotes the probability that files are music files, given interactions are not satisfying. The probabilities for other file types in Table 1 are computed in a similar way.

Node *DS* denotes the set of download speeds. It has three items, *"Fast"*, *"Medium"* and *"Slow"*, each of which covers a range of download speed.

Node *FQ* denotes the set of file qualities. It also has three items, *"High"*, *"Medium"* and *"Low "*. Its CPT is similar to the one in table 1.

Here we only take three aspects of trust into account. More relevant aspects can be added in the Bayesian network later to account for user preferences with respect to service.

Once getting nodes' CPTs in a Bayesian network, an agent can compute the probabilities that the corresponding file provider is trustworthy in different aspects by using Bayes rules, such as $p(T = 1 \mid FT = "Music")$ – the probability that the file provider is trustworthy in providing music files, $p(T = 1 \mid FQ = "High")$ – the probability that the file provider is trustworthy in providing files with high quality, $p(T = 1 \mid FT = "Music", FQ = "High")$ – the probability that the file provider is trustworthy in providing music files with high quality. Agents can set various conditions according to their needs. Each probability represents trust in an aspect of the file provider's competence. With the Bayesian networks, agents can infer trust in the various aspects that they need from the corresponding probabilities. That will save agents much effort in building each trust separately, or developing new trust when conditions change. After each interaction, agents update their corresponding Bayesian networks.

3.3 Evaluation of an Interaction

Agents update their corresponding Bayesian networks after each interaction. If an interaction is satisfying, *m* and *n* are both increased by 1 in formula (1). If it is not satisfying, only *n* is increased by 1. Two main factors are considered when agents judge an interaction, the degree of their satisfaction with the download speed s_{ds} and the degree of their satisfaction with the quality of downloaded file s_{fq}. The overall degree of agents' satisfaction with an interaction s is computed as the following:

$$s = w_{ds} * s_{ds} + w_{fq} * s_{fq}, \quad \text{where } w_{ds} + w_{fq} = 1 \qquad (2)$$

w_{ds} and w_{fq} denote weights, which indicate the importance of download speed and the importance of file quality to a particular agent (depending on the user's preferences). Each agent has a satisfaction threshold s_t. If $s < s_t$, the interaction is unsatisfying; otherwise, it is satisfying.

3.4 Handling Other Agents' Recommendations

In current file sharing peer-to-peer application, users find files by using the search function. In most of situations, they get a long list of providers for an identical file. If a user happens to select an unsuitable provider, who provides files with bad quality or slow download speed, the user will waste time and effort. If this situation happens several times, the users will be frustrated. In order to solve the problem, we use the mechanism of trust and reputation. Once an agent receives a list of file providers for a given search, it can arrange the list according to its trust in these file providers. Then the agent chooses the most trusted file providers in the top of the list to download files from. If the agent has no experiences with the file provider, it can ask other agents to make recommendations for it. The agent can send various recommendation requests according to its needs. For example, if the agent is going to download a movie, it may care about the movie's quality. Another agent may care about the speed. So the request can be "Does the file provider provide movies with good qualities?" If the agent cares both about the quality and the download speed, the request will be something like "Does the file provider provide files with good quality at a fast download speed? ". When other agents receive these requests, they will check their trust-representations, i.e. their Bayesian networks, to see if they can answer such questions. If an agent has downloaded movies from the file provider before, it will send recommendation that contains the value $p(T=1 \mid FT="Music", FQ="High")$ to answer the first request or the value $p(T=1 \mid FT="Music", FQ="High", DS="Fast")$ to answer the second request. The agent might receive several such recommendations at the same time, which may come from the trustworthy acquaintances, untrustworthy acquaintances, or strangers.

If the references are untrustworthy, the agent can discard their recommendations immediately. Then the agent needs to combine the recommendations from trustworthy references and from unknown references to get the total recommendation for the file provider:

$$r_{ij} = w_t * \frac{\sum_{l=1}^{k} tr_{il} * t_{lj}}{\sum_{l=1}^{k} tr_{il}} + w_s * \frac{\sum_{z=1}^{g} t_{zj}}{g} \quad \text{,where } w_t + w_s = 1 \quad (3)$$

r_{ij} is the total recommendation value for the j^{th} file provider that the i^{th} agent gets. k and g are the number of trustworthy references and the number of unknown references, respectively. tr_{il} is the trust that the i^{th} user has in the l^{th} trustworthy reference. t_{lj} is the trust that the l^{th} trustworthy reference has in j^{th} file provider. t_{zj} is the trust that the z^{th} unknown reference has in j^{th} file provider. w_t and w_s are the weights to indicate how the user values the importance of the recommendation from trustworthy references and from unknown references. Since agents often have different preferences and points of view, the agent's trustworthy acquaintances are those agents that share similar preferences and viewpoints with the agent most of time. The agent should weight the recommendations from its trustworthy acquaintances higher than those recommendations from strangers. Given a threshold q, if the total recommendation value is greater than q, the agent will interact with the file provider; otherwise, not.

If the agent interacts with the file provider, it will not only update its trust in the file provider, i.e. its corresponding Bayesian network, but also update its trust in the agents that provide recommendations by the following reinforcement learning formula:

$$tr_{ij}^{n} = a * tr_{ij}^{o} + (1-a) * e_a \quad (4)$$

tr_{ij}^{n} denotes the new trust value that the i^{th} agent has in the j^{th} reference after the update; tr_{ij}^{o} denotes the old trust value. a is the learning rate – a real number in the interval [0,1]. e_a is the new evidence value, which can be -1 or 1. If the value of recommendation is greater than q and the interaction with the file provider afterwards is satisfying, e_a is equal to 1; in the other case, since there is a mismatch between the recommendation and the actual experience with the file provider, the evidence is negative, so e_a is -1.

Another way to find if an agent is trustworthy or not in telling the truth is the comparison between two agents' Bayesian networks relevant to an identical file provider. When agents are idle, they can "gossip" with each other periodically, exchange and compare their Bayesian networks. This can help them find other agents who share similar preferences more accurately and faster. After each comparison, the agents will update their trusts in each other according the formula:

$$tr_{ij}^{n} = b * tr_{ij}^{o} + (1-b) * e_b \quad (5)$$

The result of the comparison e_b is a number in the interval [-1, 1]. b is the learning rate – a real number in the interval [0,1] which follows the constraint $b > a$. This is because the Bayesian network collectively reflects an agent's preferences and viewpoints based on all its past interactions with a specific file provider. Comparing the two agents' Bayesian networks is tantamount to comparing all the past interactions of the two agents. The evidence e_a in formula (4) is only based on one interaction. The evidence e_b should affect the agent's trust in another agent more than e_a.

How do the agents compare their Bayesian networks and how is e_b computed? First, we assume the structures of Bayesian networks of all agents have the same structure.

We only compare the values in their Bayesian networks. Suppose agent 1 will compare its Bayesian network (see Figure 1) with the corresponding Bayesian network of agent 2. Agent 1 obtains the degree of similarity between the two Bayesian networks by computing the similarity of each pair of nodes (*T*, *DS*, *FQ* and *FT*), according to the similarity measure based on Clark's distance [12], and then combining the similarity results of each pair of nodes together.

$$e_b = 1 - 2 * \sum_{i=1}^{4} (w1_i * c_i), \text{ where}$$

$$w1_1 + w1_2 + w1_3 + w1_4 = 1 \quad (6)$$

$$c_1 = \sqrt{\frac{(v1_{11} - v2_{11})^2}{(v1_{11} + v2_{11})^2} + \frac{(v1_{12} - v2_{12})^2}{(v1_{12} + v2_{12})^2}} \quad (7)$$

$$c_i = \frac{\sum_{j=1}^{2} \sqrt{\sum_{l=1}^{h_i} \frac{(v1_{ijl} - v2_{ijl})^2}{(v1_{ijl} + v2_{ijl})^2}}}{2}, \quad \text{where i = 2, 3, 4} \quad (8)$$

$w1_1$, $w1_2$, $w1_3$ and $w1_4$ are the weights of the node *T*, *DS*, *FQ*, and *FT*, respectively, related to agent 1, which indicate the importance of these nodes in comparing two Bayesian networks. c_1, c_2, c_3 and c_4 are the results of comparing agent 1 and agent 2's CPTs about node *T*, *DS*, *FQ* and *FT*. Since the node T is the root node and it has only one column in its CPT, while other nodes (*DS*, *FQ*, *FT*) are the leaf nodes and have two columns of values in theirs CPTs, we compute c_1 differently from c_2, c_3, and c_4. h_i denotes the number of values in the corresponding node. $h_2 = 3$; $h_3 = 3$; $h_3 = 5$. $v1_{11}$ and $v1_{12}$ are the values of *p(T = 1)* and *p(T = 0)* related to agent 1. $v2_{11}$ and $v2_{12}$ are the values of *p(T = 1)* and *p(T = 0)* related to agent 2. $v1_{ijl}$ and $v2_{ijl}$ are the values in agent 1's CPTs and agent 2's CPTs, respectively.

The idea of this metric is that agents compute not only their trust values, their CPTs, but also take into account their preferences (encoded as the weights, $w1_1$, $w1_2$, $w1_3$, $w1_4$). So agents with similar preferences, such as the importance of file type, quality, download speed, will weight each other's opinions higher.

4. Experiments

In order to evaluate this approach, we developed a simulation of a file sharing system in a peer-to-peer network. The system is developed on the JADE 2.5. For the sake of simplicity, each node in our system plays only one role at a time, either the role of file provider or the role of an agent. Every agent only knows other agents directly connected with it and a few file providers at the beginning.

Every agent has an interest vector. The interest vector is composed of five elements: *music*, *movie*, *image*, *document* and *software*. The value of each element indicates the strength of the agent's interests in the corresponding file type. The files the agent wants to download are generated based on its interest vector. Every agent keeps two lists. One is the agent list that records all the other agents that the agent has interacted with and its trust values in these agents. The other is the file provider list that records the known file providers and the corresponding Bayesian networks representing the agent's trusts in these file providers. Each file provider has a capability vector showing its capabilities in different aspects, i.e. providing files with different types, qualities and download speeds.

Our experiments involve 10 different file providers and 40 agents. Each agent will gossip with other agents periodically to exchange their Bayesian networks. The period is 5, which means after each 5 interactions with other agents, the agent will gossip once. $w_{ds} = w_{fq} = 0.5$; $a = 0.3$; $b = 0.5$; $w1_1 = w1_2 = w1_3 = w1_4 = 0.25$. The total number of interactions is 1000. We run each configuration for 10 times and use the means for the evaluation criteria.

5. Results

The goal of the first experiment is to see if a Bayesian network-based trust model helps agents to select file providers that match better their preferences. Therefore we compare the performance (in terms of percentage of successful recommendations) of a system consisting of agents with Bayesian network-based trust models and a system consisting of agents (without Bayesian networks, BN) that represent general trust, not differentiated to different aspects. Successful recommendations are those positive recommendations (obtained based on formula 3) when agents are satisfied with interactions with recommended file providers. If an agent gets a negative recommendation for a file provider, it will not interact with the file provider. We have two configurations in this experiment:

- Trust and reputation system with BN: the system consists of agents with Bayesian networks-based trust models that exchange recommendations with each other;
- Trust and reputation system without BN: the system consists of agents that exchange recommendations, but don't model differentiated trust in file providers.

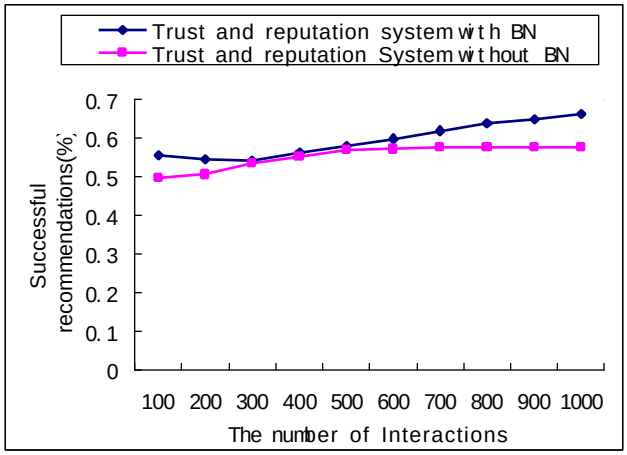

Figure 2. Trust and Reputation System with BN vs. Trust and Reputation System without BN

Figure 2 shows that the system using Bayesian networks performs slightly better than the system with general trust in terms of the percentage of successful recommendations.

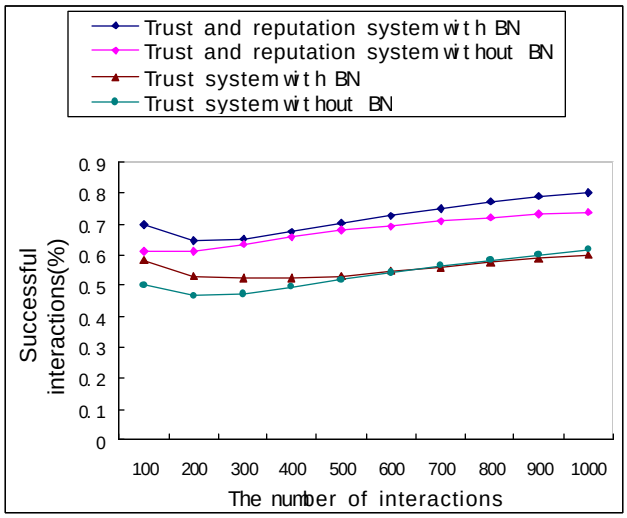

Figure 3. The Comparison of Four Systems

The goal of the second experiment is to see if exchanging recommendation values with other agents helps agents to achieve better performance (defined as the percentage of successful interactions with file provider). For the reason, we compare four configurations:

- Trust and reputation system with BN;
- Trust and reputation system without BN;
- Trust system with BN: the system consists of agents with Bayesian networks-based trust models, which don't exchange recommendations with each other;
- Trust system without BN: the system consists of agents that have no differentiated trust models and don't exchange recommendations with each other.

Figure 3 shows that the two systems, where agents share information with each other, outperform the systems, where agents do not share information. The trust system using Bayesian networks is slightly better than the trust system without using Bayesian networks. There is an anomaly in the case when agents do not share recommendations, since in the end of the curve, the system without BN perform better than the system with BN. This could be explained with an imprecise BN due to insufficient experience.

In some sense, an agent's Bayesian network can be viewed as the model of a specified file provider from the agent's personal perspective. In our experiments, we use a very simple naïve Bayesian network, which can not represent complex relationships. In the real file-sharing system, the model of file providers might be more complex and required the use of a more complex Bayesian network. Our Bayesian network only involves three factors. If we build a more complex Bayesian network and add more aspects into it, the system performance might be improved.

6. Discussion and Related work

How many Bayesian networks can an agent afford to maintain to represent its trust in other agents in the networks? It depends on the size of the network and the likelihood that agents have repeated interactions. Resnick [15] empirically shows that 89.0% of all seller-buyer pairs in eBay conducted just one transaction during a five-month period and 98.9% conducted no more than four. The interactions between the same seller and the same buyer are not repeatable. The buyer's trust in a seller is only based on one direct interaction. The seller's reputation is mostly built on the buyers' having a single experience with the seller. This situation often happens in a very large network or in large e-commence sites. Since there are a large number of sellers and buyers, the chance that a buyer meets the same seller is rare. But if the kind of goods being transacted is only interesting to a small group of people, for example, collectors of ancient coins, the interactions about this kind of goods happen almost exclusively in a small group. So the probability that sellers and buyers have repeated interactions will be high, and they will be able to build trust in each other by our method.

Our approach is useful in situations where two agents can repeatedly interact with each other. In a small-size network, there is no doubt that our approach is applicable. For a large network, our approach is still suitable under the condition that the small-world phenomenon happens. The small-world phenomenon was first discovered in the 1960ies by social scientists. Milgram's experiment showed that people in the U.S. are connected by a short (average length of 6) chain of intermediate acquaintances. Other studies have shown that people tend to interact with other people in their small world more frequently than with people outside. The phenomenon also happens in peer-to-peer networks. Jovanovic's work [9] proves that the small-world phenomenon occurs in Gnutella. It means that agents are inclined to get files from other agents from a small sub-

community. This small sub-community often consists of agents that have similar preferences and viewpoints.

Abdul-Rahman and Hailes [1] capture the most important characteristics of trust and reputation and propose the general structure for developing trust and reputation in a distributed system. Most of the later works in the area follow their ideas, but in different application domain, such as [3, 5, 11].

Sabater and Sierra's work [16] extends the notion of trust and reputation into social and ontological dimensions. Social dimension means that the reputation of the group that an individual belongs to also influences the reputation of the individual. Ontological dimension means that the reputation of an agent is compositional. The overall reputation is obtained as a result of the combination of the agent's reputation in each aspect. Our approach integrates these two previous works [1, 16], and applies them to file sharing system in peer-to-peer networks. Another difference between our work and Sabater and Sierra's work is that we use Bayesian networks to represent the differentiated trust at different aspects, other than the structure of ontology. Another difference is that we don't treat the differentiated trusts as compositional. Usually the relationship between different aspects of an agent is not just compositional, but complex and correlative. Our approach provides an easy way to present a complex and correlative relationship. Our approach is also flexible in inferring the trust of an agent for different needs. For example, sometimes we care about the overall trust. Sometimes we only need to know the trust in some specific aspect. This bears parallel with work on distributed user modeling and purpose-based user modeling [13, 20].

Cornelli's work [5], like ours, is in the area of file sharing in peer-to-peer networks. However, it concentrates on how to prevent the attacks to a reputation system and does not discuss how agents model and compute trust and reputation.

7. Conclusions

In this paper, we propose a Bayesian network-based trust model. Bayesian networks provide a flexible method to represent differentiated trust in different aspects of each other's capability and combine different aspects of trust. We evaluated our approach, in a simulation of a file sharing system in a peer-to-peer network. Our experiments show that the system where agents communicate their experiences (recommendations) outperforms the system where agents do not communicate with each other, and that a differentiated trust adds to the performance.

Future work includes adding more aspects in the Bayesian networks, trying to find the key parameters that influence the system performance, and testing the system under other performance measures, for example, how fast an agent can locate a trustworthy service provider. Applying this approach to distributed systems for computational services is particular promising.

References

[1] Abdul-Rahman A. and Hailes S. "Supporting trust in virtual communities". In Proceedings of the Hawai'i International Conference on System Sciences, Maui, Jan 2000.

[2] Abdul-Rahman A. and Hailes S. "A Distributed Trust Model". In Proceedings of the 1997 New Security Paradigms Workshop, 48-60. ACM, 1997.

[3] Azzedin F. and Maheswaran M. "Evolving and Managing Trust in Grid Computing Systems". IEEE Canadian Conference on Electrical & Computer Engineering (CCECE '02), May 2002.

[4] Carter J., Bitting E. and Ghorbani A. "Reputation Formalization for An Information-Sharing Multi-Agent System", 515-534. Computational Intelligence, Volume 18, Number 4, November 2002.

[5] Cornelli F. and Damiani E. "Implementing a Reputation-Aware Gnutella Servent". In Proceedings of the International Workshop on Peer-to-Peer Computing, Pisa, 2002.

[6] Esfandiari B. and Chandrasekharan S. "On how agents make friends: Mechanisms for trust acquisition". In 4th Workshop on Deception, Fraud and Trust in Agent Societies, Montreal, 2001.

[7] Falcone R. and Shehory O. "Trust Delegation and Autonomy: Foundations for Virtual Societies". AAMAS tutorial 12, July 16, 2002.

[8] Hales, D. "Group Reputation Supports Beneficent Norms". The Journal of Artificial Societies and Social Simulation (JASSS) vol. 5, no. 4, 2002.

[9] Jovanovic M. "Modeling Large-scale Peer-to-Peer Networks and a Case study of Gnutella", University of Cinicnnati, master thesis, April 2001.

[10] Milojicic D. S., Kalogeraki V. and Lukose R. "Peer-to-Peer Computing", Tech Report: HPL-2002-57, http://www.hpl.hp.com/techreports/2002/HPL-2002-57.pdf

[11] Montaner M. and L´opez B. "Opinion based filtering through trust". In Proceedings of the 6th International Workshop on Cooperative Information Agents (CIA'02), Madrid (Spain), September 18-20 2002.

[12] Mui L., Halberstadt A. and Mohtashemi M. "Notions of reputation in multi-agents systems: A review". In Proceedings of Autonomous Agents & Multiagent Systems (AAMAS'02), 280–287, Bologna, Italy, 2002..

[13] Niu X., McCalla G., Vassileva J. (to appear) "Purpose-based User Modelling in a Multi-agent Portfolio Management System". In Proceedings of User Modeling UM03, Johnstown, PA, June 22-26, 2003.

[14] Heckerman, D. "A Tutorial on Learning with Bayesian Networks", Microsoft Research report MSR-TR-95-06, 1995.

[15] Resnick P. and Zeckhauser R. "Trust Among Strangers in Internet Transactions: Empirical Analysis of eBay's Reputation System". NBER Workshop on Empirical Studies of Electronic Commerce, 2000.

[16] Sabater J. and Sierra C. “Regret: a reputation model for gregarious societies”. In 4thWorkshop on Deception, Fraud and Trust in Agent Societies, 2001.

[17] Schafer B.J. Konstan A, J, and Riedl J. "Recommender Systems in E-Commerce". ACM Conference on Electronic Commerce (EC-99), November 3-5, 1999, Denver, CO.

[18] Tang T.Y, Winoto P. and Niu X. “Who can i trust? Investigating trust between users and agents in a multi-agent portfolio management system ”. AAAI-2002 Workshop on Autonomy, Delegation, and Control: From Inter-agent to Groups. Edmonton, Canada.

[19] Vassileva J., Breban S. and Horsch M. “Agent Reasoning Mechanism for Long-Term Coalitions Based on Decision Making and Trust”. Computational Intelligence, Vol. 18, no. 4, 2002.

[20] Vassileva J., McCalla G. and Greer J. (accepted 17 October 2001) “Multi-Agent Multi-User Modeling”, to appear in User Modeling and User-Adapted Interaction.

[21] Yolum P. and Singh M. “Locating Trustworthy Services” In Proceedings of the First International Workshop on Agents and Peer-to-Peer Computing (AP2PC), 2002.

[22] Yu B. and Singh P. M. “A social mechanism of reputation management in electronic communities”. In Proceedings of Fourth International Workshop on Cooperative Information Agents, 154–165, 2000.

[23] Yu B. and Singh P. M. “An Evidential Model of Distributed Reputation Management”. In Autonomous Agents & Multiagent Systems (AAMAS’02), 294–301, Bologna, Italy, 2002.

[24] Zacharia G. Moukas A. and Maes P. “Collaborative Reputation Mechanisms in Electronic Marketplaces” In 32nd Annual Hawaii International Conference on System Science (HICSS-32), 1999.