PET: A PErsonalized Trust Model with Reputation and Risk Evaluation for P2P Resource Sharing

Zhengqiang Liang and Weisong Shi
Wayne State University
{*sean,weisong*}@*wayne.edu*

***Abstract*— Building a good cooperation in the P2P resource sharing is a fundamental and challenging research topic because of peer anonymity, peer independence, high dynamics of peer behaviors and network conditions, and the absence of an effective security mechanism. In this paper, we propose PET, a personalized trust model, to help the construction of a good cooperation, especially in the context of economic-based solutions for the P2P resource sharing. The trust model consists of two parts: reputation evaluation and risk evaluation. Reputation is the accumulative assessment of the long-term behavior, while the risk evaluation is the opinion of the short-term behavior. The risk part is employed to deal with the dramatic spoiling of peers, which makes PET differ from other trust models that based on the reputation only. This paper contributes to first modeling the risk as the opinion of short-term trustworthiness and combining with traditional reputation evaluation to derive the trustworthiness in this field.**

I. Introduction

Economic model based resource management has been widely proposed and proves a good try in the past for the Peer-to-Peer (P2P) resource sharing. Numerous economic models including microeconomics and macroeconomics principles for resource management have been proposed in the literature [6], [14], [30], and various criteria are used for judging the effectiveness of an economic model, including social welfare, stability, and computation efficiency. However, for the work targeting for P2P environments, the trustworthiness of peers was either neglected or treated as an optional factor. The previous work also separates computational economies from the trustworthiness of participating peers, which is a necessary component to make the economic model feasible and reliable in an open environment. We envision that trustworthiness should be fundamental to the design of P2P resource management based on economic models.

A. Background

There are several definitions about trust in the literature. In this paper, we define trust as *the subjective probability by which an individual A expects that another individual B performs a given action as good as expected*. Trust evaluation provides an intuitive way to build the cooperation and brings the reciprocity for the resource management, but it is difficult to be achieved in the P2P environment due to peer anonymity, peer independence, high dynamics of the peer behaviors and conditions, and the absence of an effective security mechanism. There are several philosophic questions necessary to be addressed to design a trust model:

- Is trust relationship transitive? That is, how much is the local trustworthiness calculation affected by recommendations from others?
- Is trustworthiness a global value or a local value?
- What are the factors affecting the trustworthiness value?
- Is the reputation enough to build a good cooperation?

Different answers to these questions generate different trust models. Traditionally there are two major classes of trust models. The first class is the *Central* model (CM), which has a central trust point. Every entity in the *Central* model has the same opinion as what the central trust point has. Reputation-based systems [8], e.g., eBay [9], are the typical examples of the CM model. The certificate authority (CA) based trust model, which has been widely deployed in e-commerce [1], [26], [32], is another typical example. The CM model works fine if the central point is reliable and trustable, and provides only one type of service, but it is not a good choice in case of a large number of peers and multiple services coexisting. The second class is the *Transitive* model (TM) [3] which has a transitive trust chain. In TM, the recommendation from the recommender is highly emphasized for the trustworthiness. Actually, the CM model can be seen as a special case of the TM model with just one transitive relationship and the recommendation being totally trusted. Perils including collusion and wrong recommendations are big threats for the TM model. However, making use of the recommendation is also the merit for the TM model when the recommendation is good, which is helpful to discover the quality of other peers even without any mutual interaction. It deserves further study to find the suitable tradeoff.

B. Our Contribution

In this paper we propose PET, a personalized trust model in the context of economic-based solutions for the P2P resource sharing. Differing from the former two models, PET is an intermediate model in which the recommendation plays a moderate role as one of the many factors to derive local trustworthiness values.

PET models the reputation as the accumulative assessment for the long-term behavior, treats the risk as the opinion of the short-term behavior, and makes both of them quantified. The weights of the reputation and risk are adjustable according to different environments and requirements, which distinguishes PET from the previous work. In addition to developing a theoretical model for the PET model, we also conduct a comprehensive performance analysis in terms of four metrics: *sensitiveness, hit ratio, effectiveness*, and *applicability*, by

0-7695-2268-8/05/$20.00 (C) 2005 IEEE

applying the model to a Peer-to-Peer Web server sharing application [25]. Our evaluation results show that both reputation (long-term behavior assessment) and risk (shot-term behavior assessment) are important in designing a personalized trust model. The results also show that the PET model is flexible enough to adapt to different applications by adjusting the weights of the reputation and the risk.

The rest of the paper is organized as follows. We will give the details of the PET model in Section II. The simulation methodology and results are presented in Section III. In Section IV we discuss the related work. We summarize the paper in Section V.

II. Design of PET

Before depicting the PET model, we list four principles for the design:

- **P.1** Peer will always trust itself.
- **P.2** Bad behavior makes the trustworthiness value drop faster and good behavior increases the value slower.
- **P.3** If a peer continually behaves badly, it will be bad peer prone.
- **P.4** The recommendations from others will not dominate the calculation of the trustworthiness value, but it will gain more weight when no direct interactions happen before.

A. **Trustworthiness**

In PET the trustworthiness T is directly derived from two parts: reputation R_e and risk R_i, as shown in Figure 1. W_{Re} and W_{Ri} are the weights of R_e and R_i respectively. There are also two parts for the derivation of the reputation: recommendation (E_r, also called epidemic) and interaction-derived information (I_r). Recommendation is the opinions of other peers towards the target peer, which is collected by the *feedback collecting* component in PET. Interaction-derived information is the self-opinion from the direct interaction, which is reliable and self-determined. W_{Er} and W_{Ir} are their corresponding weights. The interaction-derived information is also the base of the risk calculation. I_r can be obtained from the feedback of the agent [25] (also collected by the feedback collecting component) or self-observation. There are a wide range of resource categories in P2P resource sharing such as CPU, content, and so on. In the context of multiple resource sharing, another program component *resource classifying* in PET is employed to identify the resource category which the feedback and self-observation information belong to, then adopts different strategies to process these information. In addition, we abstract four general behaviors which are common in the resource sharing (see Section II-B), so that PET can be applied for most resource sharing cases by just modifying its feedback collecting and resource classifying components. Some existing methods [10], [17], [28] will be helpful to implement these components, but it is out of the scope of this paper. In Subsection II-B, more details will be discussed.

According to different requirements, we can assign different weights to the reputation and risk, through which PET can meet most demands, no matter preferring to the long-term assessment or caring for the short-term assessment. Equation 1 describes the derivation of the trustworthiness T:

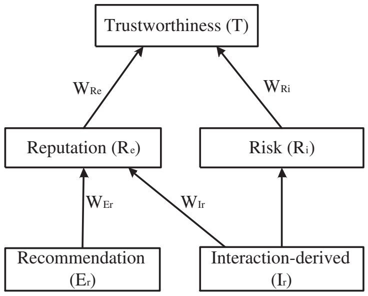


Fig. 1. Derivation of the trustworthiness.

$$T = \begin{cases} (\alpha, 1-\alpha) \times (R_e, (1-R_i))^T, & 0 \leqslant \alpha \leqslant 1 \\ R_e, \quad \text{if } R_i = 0 \\ R_i, \quad \text{if } R_e = 0 \end{cases} \tag{1}$$

where R_e is the reputation value, and R_i is the risk value. The values of *T*, R_e, and R_i are all from 0 to 1. Here W_{Re}= α and $W_{Ri} = 1 - \alpha$. If we set $\alpha = 1$, that means the weight of the risk is 0, then PET will degenerate to the traditional reputation system. However our simulation results show that risk evaluation is a very helpful component to build the trust model. Normally when the system is highly dynamic and most nodes are not good, it is recommended to set the risk with a high weight, that is, set a lower value to α, which is supported by the simulation results in Section III. How to adaptively determine the value of α is itself an interesting problem, which is our ongoing follow-up work. The basic idea is using differentiate of each component to adjust the coefficient α.

B. **Reputation Model**

In the following, we call the peer to evaluate other peers the *valuer*, the peer to be evaluated the *valuee*, and the peer that sends the trustworthiness value of the known peers to others the *recommender*. For example, when peer A tells peer C the trustworthiness value of peer B, A will be the *valuee* of B, and the recommender of C.

According to the quality of services provided by cooperating peers, we classify services into four categories, as shown in Table I. We formalize the quality set as Q = {G, L, N, B}. This coarse-grain classification is flexible enough to apply to any resource sharing. More subclasses can be introduced if necessary. All three L, N, and B services are bad services, and will cause the *valuer* to decrease the *valuee*'s score. It is worth noting that PET treats *No Response* as a bad action, so that peers joining and leaving the system highly dynamically will get low trustworthiness because the dynamics decreases the probability of the response. This is helpful to protect PET from the churn of the P2P system. For the *valuer* there is a map function h that maps from Q to a score for one cooperation:

$$h(x) = \begin{cases} S_1 & , \quad x = G, \ S_1 > 0 \\ S_2 & , \quad x = L, \ S_2 < 0 \text{ and } |S_2| > S_1 \\ S_3 & , \quad x = N, \ S_3 < S_2 \\ S_4 & , \quad x = B, \ S_4 < S_3 \end{cases} \tag{2}$$

Service Quality	Description
Good (G)	The service is as good as expected.
Low Grade (L)	The service is correct, but with some degradation, e.g., delay for service.
No Response (N)	Under this category the incoming service request is rejected.
Byzantine Behavior (B)	Giving the wrong or even malicious response for the incoming requests.

TABLE I

FOUR DIFFERENT SERVICE QUALITIES USED IN THE PET MODEL.

The score is used to calculate the reputation, as seen in Equation 3. Simply we can choose a constant value for S_1 to S_4 (Note, S_2, S_3, and S_4 are negative. In our simulation, the values of S_1 to S_4 are 1, -2, -3, -4 separately). For example, let $h(B) = -6$, which means when the *valuee* i's service category is known as `Byzantine` behavior, i's total score S will be dropped by six. However, we can also adjust these values with an appropriate adaptive mechanism.

Reputation value is the historical accumulation for *valuee*'s past behavior from the *valuer*'s viewpoint. It will reflect the overall quality of the peer for a long time period. Sometimes some good peers will misbehave because of nonsubjective factors. For example, a good peer will reject a service request due to the breakdown of the physical link, but after recovery, it will provide good service continually. If we want to forgive the occasional nonsubjective misbehavior, we can set a high value to α ($\alpha > 0.5$ for example) to amortize the temporary service outage, that is, to emphasize the reputation (long-term accumulation) in the derivation of the trustworthiness. Reputation is derived from E_r and I_r, as shown in Equation 3:

$$R_e = \quad (\beta, 1-\beta) \times (E_r, I_r)^T, \; 0 \leqslant \beta \leqslant 1 \tag{3}$$

where

$$E_r = \frac{\sum T_i}{N_e}, \quad I_r = \begin{cases} 1, & S \geqslant T_{good} \\ \frac{S}{T_{good}}, & 0 < S < T_{good} \\ 0, & S \leqslant 0 \end{cases}$$

Here W_{Er}= β and W_{Ir} = $1-\beta$. T_i stands for the recommendation coming from peer `i`, and N_e is the number of the recommendations. So E_r is the average value of recommendations. In order to prevent the malicious recommenders from repeating their malicious recommendations to mislead the recommendation calculation, only the newest opinion from the recommender is saved and used for the calculation. Another solution is based on the social network [27], [31], [33], [34], [35] to assign different weights to the recommendations according to their corresponding trustworthiness values. However, we do not adopt this approach because of two reasons. First, in PET personalization is of primary concern. In other words, the recommendation from a trustable peer is more trustable does not necessarily hold. For example, for peer A, peer B is trustable, and for peer C, peer A is trustable, however, B can be a bad peer for C because B is distant from C and can not afford good quality services to C. In this case, A's recommendation is not helpful but harmful for C. Second, this solution will increase the load of the system considerably, especially in the context of heterogeneous resource sharing. So, considering the simplicity and the actual effect, averaging the recommendation and viewing every recommendation neutral are reasonable from the view point of the workload and the indeterminate reliability of the recommendation.

T_{good} is the score threshold to normalize the score. S is the total score. For example, in our simulation T_{good} is set to 100. If the valuee's score is higher or equal to 100, its I_r will be set to one.

Currently, based on our simulation results in Section III, setting a low value for β (the weight of the recommendation) statically, e.g., 0.2, is recommended. Of course, if the environment is known to be highly trustable, assigning a high weight to the recommendation can efficiently improve the convergence speed of the trustworthiness. But since PET aims to deploy in the P2P community with different kinds of peers, it is good to lower the role of the recommendation. The reasons are:

- Different peers may have different views on the same resource provider because different peers may have different situation-specific criteria and requirements for the sharing.
- Peer's behavior can change dynamically, which implies that we can not compute the trustworthiness relying much on the recommendations from others.
- Fraudulent recommendations, especially the collusive recommendations, are very difficult to handle if the trustworthiness calculation relies too much on the recommendation.

However, as mentioned before, it is not a good answer to ignore the recommendation. Assigning it a lower weight β is reasonable solution, which is supported by the simulation results in Section III.

C. Risk Model

Reputation is an accumulative value for the past behavior and reflects the overall evaluation on the *valuee*. However, it is not sensitive enough to perceive the suddenly spoiling peer because it needs time to decrease the accumulative score. Risk evaluation can help to solve this problem.

The risk value R is normalized to the worse case, i.e., the ratio of the effect of all bad services received by the peer during this time interval over the worst effect when all services received in this time interval are Byzantine service, as shown in Equation 4, where `B`, `N` and `L` are the service qualities defined in Table II and $h(i)$ is the score for the cooperation with service quality `i` defined in Equation 2. N_i is the number of services with quality `i`. As mentioned before, every peer has its own personalized views about the community, so the recommendations are not reliable even they are from the trustable peers. Therefore, to make the personalized trust more precise, we only use the interaction-derived information to calculate the risk value.

$$R = \frac{\sum_{i=B,N,L}(N_i * h(i))}{h(B) * \sum_{j=G,B,N,L}(N_j)} \tag{4}$$

Trustworthiness is a temporal value, because the behavior of the peer will change dynamically. The old trustworthiness value may totally misrepresent one peer's quality after some time. To solve this problem, decay function is used in [2]. However, it is difficult to choose a unique decay function for all peers, because different peers have different behavior patterns. In PET, every *valuee* has its individual risk value. Risk window is employed for the risk calculation. Only the behaviors of the *valuee* inside the window are taken into consideration. With the window shifting forward, the risk value reflects the fresh statistics of the *valuee*'s recent behaviors. The window size plays an important role in the risk calculation. The smaller the window size is, the more the shorter-term assessment is favorite by the trustworthiness calculation. To reduce the risk from the cooperation, users can focus more on the risk value R by assigning it a high weight. Yet this will decrease the availability of the resources, because the less risk for the cooperation is requested, the less peers are qualified to be cooperated. The user can make a tradeoff between the risk and the resource availability by adjust the weight of the risk. The effects of risk and the change of the window size on the trustworthiness can be found in Section III-D.

Using risk evaluation, the risk-sensitive users can find the bad peers much earlier than only using the reputation value, which is illustrated in Figure 2. In this figure, x-axis represents the time, and y-axis shows the behavior of the peer. The line reflects the variation of the peer's behavior. At time t_0, the peer starts to behave badly. The bad peer will not be discovered until time t_2 when only the reputation value is considered. If the risk evaluation is federated, the time will be efficiently shortened to time t_1 due to the high risk value.

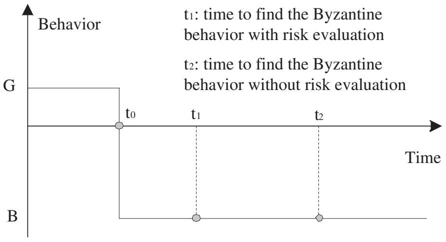

Behavior
t1: time to find the Byzantine behavior with risk evaluation
t2: time to find the Byzantine behavior without risk evaluation
G
t0
t1
t2
Time
B

Fig. 2. Risk evaluation helps to find the peer with dramatic spoiling earlier.

III. Experimental Analysis

In this section, we will explore the feasibility of the PET model by simulation. We start with an introduction of the application scenario used in the simulation, followed by a description of some crucial concepts. Then, we analyze the simulation results and summarize them.

A. Peer-to-Peer Web Sharing

We use simulation to assess the effects of different components under various environment options in the context of P2P Web server sharing application [25], which is a new content delivery mechanism for both static and dynamic Web content by federating participating Web servers together in a P2P fashion. It empowers the individual peer which is autonomous with respect to managing the resources and replica placement. Each Web server is a peer and serves a bound of clients. The peers pool their resources to help each other during individual peer's peak loads and/or system failures. It is worth noting that in this application only the servers providing the service form the P2P sharing system. The clients here are just requesting services and providing feedbacks, and do not belong to the P2P system we consider. The main concept behind the workability of this arrangement is an understanding that not all companies which form the P2P network will have peak loads on their web sites simultaneously.

In the simulation, PET is integrated with the **M**ultiple **CU**rrency **B**ased **E**conomic model (M-CUBE) [16]. The M-CUBE model is motivated by several features of current world economic models, in which there are multiple currencies, and the currencies are associated with application-level services directly. Each peer issues its own currency and self determines the price of the resource. When one peer needs the resource (service) from others, it must get the currencies of others first through currency exchange protocol using its own currencies. Initially the currency ratio between any peer pair is set to be equal (1:1), then the ratio will be self-adjusted automatically based on the mutual trustworthiness derived from PET.

In the following subsections, we will present the related concepts at first; then we depict the simulation settings; after that the discussions on the results are presented; finally we list the summary of the results.

B. Concepts

Before presenting the experimental results, it is necessary to make some concepts clear.

Cooperation: When peer A uses B's currency to ask for B's service, and B satisfies A's request, we say A has one cooperation with B, or A cooperates with B.

Active Cooperator: The cooperators which are chosen to be ready for the cooperation are called active cooperators. When the cooperation is needed, active cooperators will be considered first.

Inactive Cooperator: The cooperators which are not ready for the cooperation are called inactive cooperators. They may be the peers exchanged the currency before, or the peers heard from others through the recommendations. They can also be the active cooperators before, but now are purged because of their bad trustworthiness values.

Cooperated Cooperator: When A has cooperated with B, B will be A's cooperated cooperator. A cooperated cooperator can be either an active cooperator or an inactive cooperator.

Good-Known-Cooperator: When the trustworthiness value of one cooperator is over a certain threshold (the value is set to 0.7 in our simulation), the cooperator will be called the good-known-cooperator.

Active Cooperator Table: The information of the active cooperator will be stored in this table. The main contents

Settings			Illustrations
Client Number	**C1**	4700 Clients	Small-size population.
	C2	9400 Clients	Large-scale population.
The Proportion of Peer with Different Quality (G:L:N:B:D)	**P1**	20% : 10% : 10% : 30% : 30%	To simulate the community with less good peers and all kinds of peers coexist.
	P2	20% : 0% : 0% : 0% : 80%	To simulate high dynamic community with many dynamic peers.
	P3	20% : 20% : 20% : 40% : 0%	To simulate the stable community without dynamic peers.
	P4	50% : 10% : 20% : 10% : 10%	To simulate a half-good community.
	P5	80% : 5% : 5% : 5% : 5%	To simulate a terrific community.
Malicious Recommendation	**M1**	Malicious recommendation	Spreading the distorted facts.
	M2	Correct recommendation	Spreading the true facts.
Weight of Different Components	**W1**	$\alpha = 0.3, \beta = 0.2$	Emphasizing R_i and I_r.
	W2	$\alpha = 0.3, \beta = 0.5$	Emphasizing R_i and relying more on E_r.
	W3	$\alpha = 0.7, \beta = 0$	Emphasizing R_e and ignoring E_r.
	W4	$\alpha = 0.7, \beta = 0.2$	Emphasizing R_e and I_r.
	W5	$\alpha = 0.7, \beta = 0.5$	Emphasizing R_e and relying more on E_r.
	W6	$\alpha = 1, \beta = 0.2$	Ignoring R_i and Emphasizing I_r.
Size of Risk Window	**S1**	4	Small window size. Based on last four services.
	S2	32	Large window size. Based on last 32 services.

TABLE II

SIMULATION SETTINGS AND THEIR ILLUSTRATIONS.

include the cooperator ID and its corresponding number of the currency.

History Table: The information of all cooperators, including the active cooperators and the inactive cooperators, will be stored in this table. The main contents include the cooperator ID, trustworthiness, valuee's quality, the number of the currency, etc. When one peer receives the recommendation from other peers, it will store the recommendation information into this table. When an active cooperator is purged, its information will be kept inside this table also, and a reselection of active cooperator will be based on this table in priority.

C. Experiment Design and Settings

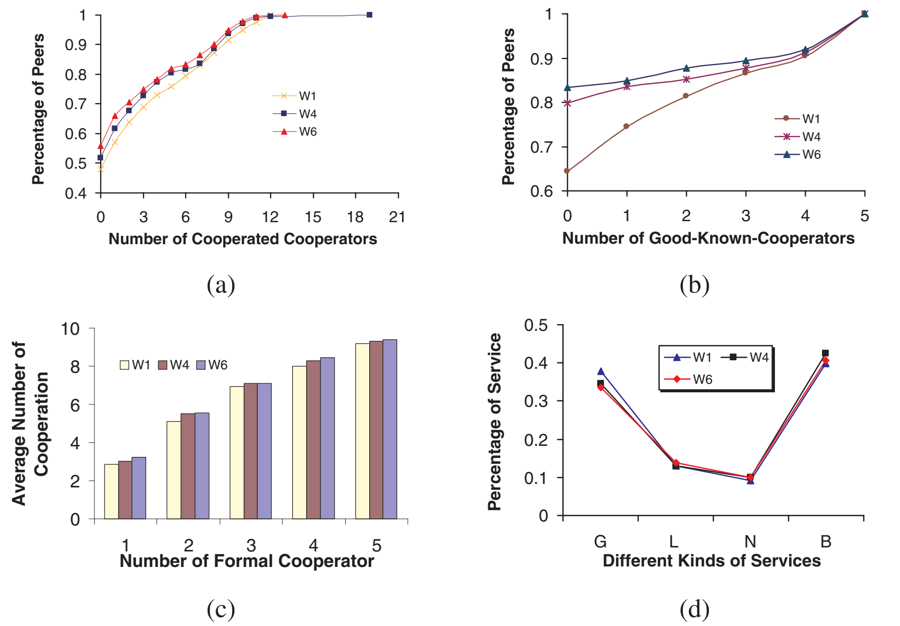
The simulation is thread-based and written in Perl language. Table II gives the details of the settings of the simulation. There are 500 peer servers to be simulated. To show the scalability of the model, two sizes of clients are used: 4,700 clients (**C1**) and 9,400 clients (**C2**). The peer servers need to cooperate with each other to make full use of the spare (computing) resource to serve the clients. HTTP requests from clients are generated using SURGE [4]. The total number of requests in the simulation is about 300,000 when using 4,700 clients, and 600,000 when using 9,400 clients.

Considering the dynamic behaviors of the peers in the real P2P community, we introduce the Dynamic quality (`D`) in the simulation in addition to the four qualities (`G, L, N, B`) described in Table I. For the peer with dynamic quality, it will change its behavior among `G`, `L`, `N`, and `B` repeatedly. Five configurations (from P1-P5) are used to simulate different P2P communities as listed in Table II. In order to simulate the malicious recommenders, peers will also have a secondary role: sending out the correct recommendation (**M1**) or malicious recommendation (**M2**). In our simulation, the malicious recommendation will rate the good peers as bad, and rate bad peers as good. Changing the weights of different model components can adjust the model to different environments. Finding some good weight settings through the simulation is one of our goals as well. To achieve this goal, six weight combinations (from W1-W6) are used as shown in Table II.

D. Results and Analysis

In the following subsections, we will present the simulation results with different experiment options. Before we start analyzing the results it is important to understand four metrics used to evaluate our model. For each evaluation sub-scenario (Section III-D.1—-Section III-D.3), we group the results of these four metrics into four graphs, with the exception for three graphs in Section III-D.5.

- **Sensitiveness:** This metric is implied by the total number of cooperated cooperators in the history table, which will increase as time goes on, because bad cooperators will be purged and new cooperators will be chosen and known until all active cooperators are good, and these peers will be added into the history table. Generally speaking, high sensitiveness is favorite for the model, because it shows that the model is active. The sensitiveness will be studied in the sub-figure (a) in every figure of this section, in which the x-axis is the number of the cooperated cooperators, and the y-axis is the cumulative distribution of the percentage of the peers which have corresponding amount of cooperated cooperators in the x-axis.
- **Hit Ratio:** The hit ratio metric is reflected by the number of the good-known-cooperators. It is worth noting that, even some peers are the peers always providing the service with good quality, we can not say they are good-known-cooperators until they are discovered to be good (the trustworthiness value is over the threshold). Even without any direct cooperation, the good peers still can be perceived through the recommendation from others. In order to prevent malicious recommendations,



(a)

(b)

(c)

(d)

Fig. 3. Risk evaluation with different weights. Other setting options of this experiment group are: C1, M2, P1, and S2. (a) CDF of the total number of cooperated cooperators in the history table, (b) CDF of the number of good-known-cooperators, (c) Number of good-known-cooperators versus related average number of cooperations, and (d) Distribution of served services.

the threshold to be a good cooperator is high so that it is very difficult for one malicious peer to fool other peers to take the bad peers as the good ones. The more the good-known-cooperators, the higher the hit ratio is. This is important because picking up the good cooperators from the community is the goal of our trust model. The hit ratio will be studied in the sub-figure (b) of every figure of this section, in which the x-axis is the number of the good-known-cooperators, and the y-axis is the cumulative distribution of the percentage of the peers which have corresponding amount of good-known-cooperators in the x-axis.

- **Effectiveness:** Effectiveness is implied by the average number of cooperations required to discover certain number of the good-known-cooperators. The lower number of this value, the more effective the model is. Except Figure 6, all the (c) sub-figures in this section show the effectiveness, in which the x-axis is the number of good-known-cooperators, and the y-axis lists the average number of cooperations to find the corresponding number of good-known-cooperators of the x-axis.
- **Applicability:** Applicability is reflected by the percentage of good services received by clients. The more good services the clients get, the more applicability the model has. The applicability will be studied in the sub-figure (d) of every figure from Figure 3 to Figure 6, and Figure 7(c), in which the x-axis lists the four service categories (`G, L, N, B`), and the y-axis presents the percentage of the requests receiving the corresponding service category.

1) **Effect of Risk Evaluation:** There are two ways to change the effect of the risk value: changing the weight of the risk W_{Ri} and adjusting the risk window size S_w. This group of experiments explore how the change of W_{Ri} influences the model while remaining S_w the same.

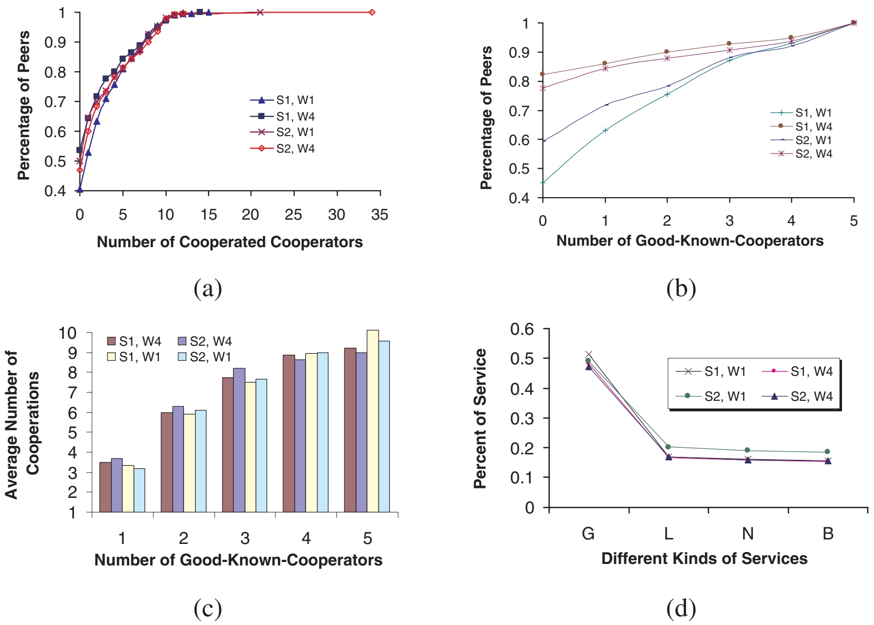
Setup: In order to compare the result more precisely, we fix other options and just change W_{Ri}. Three values of W_{Ri} are used here: W1(0.7), W4(0.3), and W6(0). Other fixed options are: C1, M2, P1 and S2.

Discussion: Intuitively, *when there are bad peers* (`L,N,B,D`), *high value of W_{Ri} will help to get more satisfactory results compared with the model with lower value of W_{Ri}*. In Figure 3(a) and Figure 3(b), the model with W1, the highest value of W_{Ri} in the three experiments, shows more sensitive and has more hit ratio than other two, which meets our expectation; we can also see that *high value of W_{Ri} will make the model more effective* from Figure 3(c). In Figure 3(d), the improvement for the applicability by the model with W1 is also the highest. All the results show that, *in the P2P community where most peers are not good, the high value of W_{Ri} is helpful to improve the model.*

2) **Effect of Risk Window Size:** Next, we investigate the risk window size S_w, the second factor to adjust the effect of the risk. Considering the weight of the risk will disturb the results, we also combine W_{Ri} into the experiments.

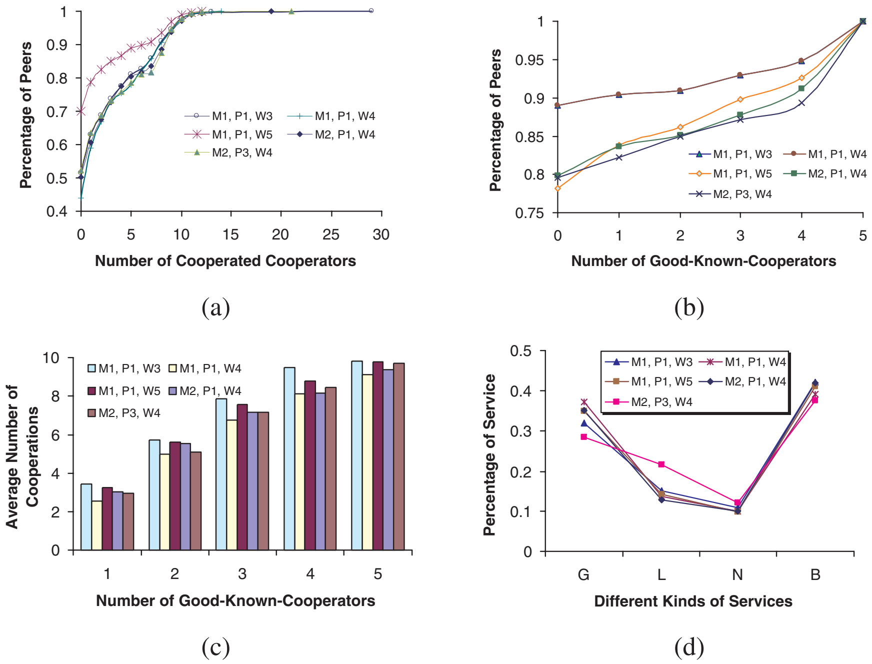
Setup: Two values of W_{Ri}, W1(0.7) and W4(0.3), and two value of S_w, S1(4) and S2(32), are chosen here. Other fixed options are: C1, M2, and P1.

Discussion: In Figure 4(a), comparing the line with options (S1, W1) with the one with options (S1, W4), we can find that in the former one about 30% peers have more than three active cooperators, while the percentage drops to 23% for the latter. This shows that with small value of S_w and high value of W_{Ri}, the model is more sensitive. This is because when the risk window size is small, the model will be more sensitive to catch the variation of the actual behavior pattern. Since in this group of experiments there are 50% bad peers and 30% dynamic peers, small window is easy to catch the



(a) (b) (c) (d)

Fig. 4. Results with different risk window sizes and different weights. Other setting options of this experiment group are: C1, M2, and P2. (a) CDF of the total number of cooperated cooperators in the history table, (b) CDF of the number of good-known-cooperators, (c) Number of good-known-cooperators versus related average number of cooperations, and (d) Distribution of served services.



(a) (b) (c) (d)

Fig. 5. Effect of recommendations. Other setting options of this experiment group are: C1 and S1. (a) CDF of the total number of cooperated cooperators in the history table, (b) CDF of the number of good-known-cooperators, (c) Number of good-known-cooperators versus related average number of cooperations, and (d) Distribution of served services.

bad peers. With higher weight than the reputation, the risk dominates the trustworthiness calculation and makes it more precise. However, when the weight drops to W4 and with the same risk window size S1, the sensitiveness drops to the least in the group. Normally the experiment with options (S1,W4) should be more effective than the experiment with the options S2 (big window size) and W4 if the risk still dominates the trustworthiness, but now it is not, which shows the reputation dominates the trustworthiness with this setting. This shows that *S_w plays an important role in improving the sensitiveness of the model when W_{Ri} is high, and plays a weak role when W_{Ri} is low*. It gives us a hint that when applying this model in the less-risk community, we can set the risk window a small value to save some storage.

Focusing on Figure 4(b), we can observe that two clusters: one cluster with option W1 and one with option W4, and for the cluster with option W4 (W_{Ri} is lower), about 10% peers have chosen and known more than two good cooperators as their active cooperators, while for the cluster with option W1 (W_{Ri} is higher) this percentage increases to more than 20%. This result indicates that *when there are a large amount of bad peers in the community (but all peers give the correct recommendations), W_{Ri} has more importance in improving the hit ratio than S_w*, otherwise the two clusters will be grouped by S1 and S2, instead of W1 and W4. This conclusion also holds true for the effectiveness in the prophase of the simulation from Figure 4(c).

Considering Figure 4(c) and Figure 4(d), we can find that, with the same W_{Ri} the small value of S_w leads to more satisfactory effectiveness, but the improvement is limited. In Figure 4(d) it is observed that 53% requests have been well served with the options (S1, W1),which is more than two times of the percentage of good peers (20%). The number told us the facts that *when most of the peers change their behaviors dynamically (here the peer configuration is P2, that is G:L:N:B:D=20%:0%:0%:0%:80%), setting higher weight to risk and the small size of the risk window will great improve the applicability*.

3) **Selecting the Weight of Recommendations:** Since selecting a suitable weight of recommendation is important for PET, we conduct this group experiments. Through the results, we will have the idea how to select the weight of the recommendation W_{Er} (defined in Equation 3) to decrease negative effects of malicious recommendations.

Setup: We will mix M1 (Malicious recommendation) and M2 (Correct recommendation), P1 (20% good peers and 30% dynamic peers) and P2 (20% good peers and 80% dynamic peers), and W3 (W_{Er} is 0), W4 (W_{Er} is 0.2) and W5 (W_{Er} is 0.5) to build different experiments. Other fixed options include C1 and S1.

Discussion: Let's focus on the lines and bars with M1 option in the following discussion. In Figure 5(a), when W_{Er} is set to W3 or W4, more than 25% peers will have three cooperated cooperators. However, when W_{Er} is set to W5, the percentage will drop to about 15%. So *the model will gain more sensitiveness when W_{Er} is set to be low*. The effect is even much close to the model with correct recommendation. In Figure 5(b), it shows that setting W_{Er} higher will make the model's hit ratio increase: the number of peers having more than two good-known-cooperators increases from 10% to 15%. Figure 5(c) shows that *with the lower value of W_{Er} (W4), PET can gain the better effectiveness than the case ignoring the recommendation (W1) and the case with higher W_{Er} (W5)*. Finally in Figure 5(d), with the option W4, among all the requests, 36% of them are served as good, higher than 32% from the case with the option W3 (ignore the recommendation). The result is even a little bit higher than the case without malicious recommendations, which implies that *to resist the malicious recommendations and find the good cooperators to serve, a lower value of W_{Er} is better than the case that ignores the recommendation (W3, W_{Er}=0) and the case that relies more on the recommendation (W5,W_{Er}=0.5)*. From the above results, we can conclude that *in a community with malicious recommenders, just ignoring others' recommendations is not a good strategy, even it improves the hit ratio. The right solution is assigning it a low weight to make a tradeoff*. From the simulation results, the tradeoff can lead to a good solution.

4) **Risk against Malicious Recommendations:** Risk model is very important in the PET model. Next, e will see the ability of the risk model against malicious recommendations from the following group of experiments.

Setup: We choose two weights of the risk W_{Ri}, W1(0.7) and W4(0.3), and take the malicious recommendation into the consideration. The other fixed options include: C1, P1 and S2.

Discussion: From Figure 6(a), Figure 6(b), and Figure 6(d) just with the option M1, it can be seen that *if we increase the value of W_{Ri}, PET has high resistance to the malicious recommendations for the sensitiveness, hit ratio, and the applicability*. Considering the sensitiveness, about 25% peers have more than four cooperated cooperators with options (M1, W1), while with options (M1, W4) this number drops to less than 15%. For the hit ratio, about 18% peers pick up more than two good-known-cooperators with options (M1, W1), while with option (M1, W4), the number drops to 9%. The result of the experiment with options (M1, W1) is even better than the case without malicious recommendations with options (M2, W4) (the number is 16%). For the applicability, the options (M1, W1) brings 36% good services, much better than 31% with the options (M1, W4), and it is even better than 35% with options (M2, W4), the case without malicious recommendations. The merit also appears in the anaphase considering the effectiveness from the Figure 6(c). In summary, *when the malicious recommendations exist, setting W_{Ri} with a high value is great helpful to resist the malicious recommendation*.

5) **Long Range Effect:** Now, we are in a position to investigate the long range effect of our model. In other words, what will happen if the scale of the system increases.

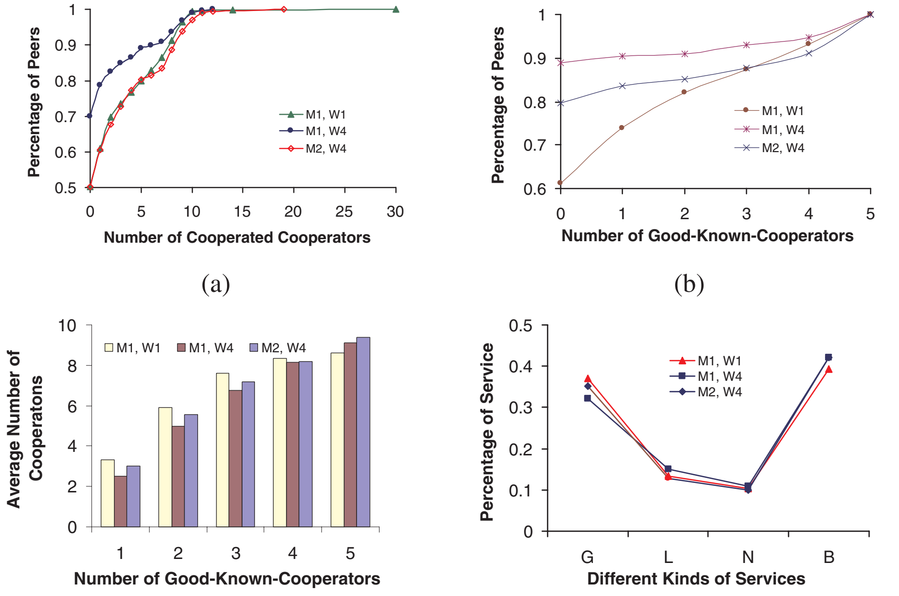
Setup: Here we will combine two groups of options, (C1, C2) and (P1, P4), to proceed the simulation. Other options are the same: M2, S1, and W4.

Discussion: All the sub-figures show that the more clients, which means the longer the execution time and the larger the scale, the more effective and applicable the PET model will be. In Figure 7(b), the number of good-known-cooperators with C2 option is seven, 40% more than the one with C1 option. From the Figure 7(c), it can be seen that with the increasing of the client number from C1 to C2, the good service percentage increases from 35% to 48%. All these imply that the experiment results will be better if the scale of experiment increases. Thus we expected that PET is promising in the large scale P2P community.

E. Summary

We summarize the major observations in the following:

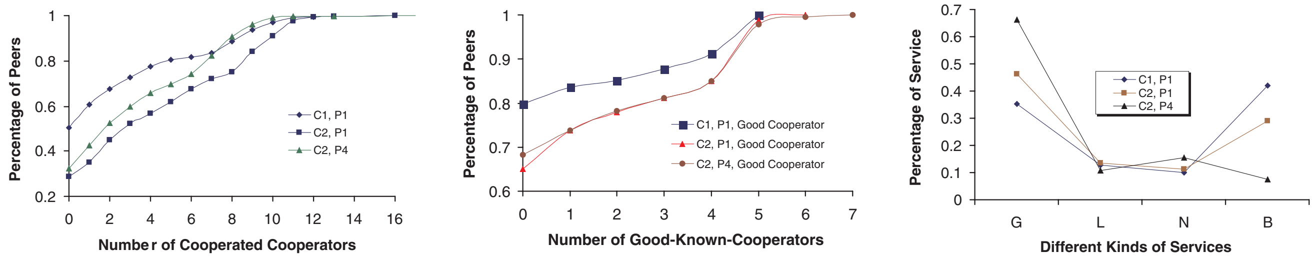
- **High weight of the risk** is much more helpful to improve the performance of the model, including the sensitiveness, effectiveness, hit ratio and applicability when more peers in the community are not good. It is also very helpful to resist the negative effect of malicious recommendations.
- **Small risk window size** is helpful to improve the sensitiveness of the model when the weight of the risk is set to high.



(a) (b)

(c) (d)

Fig. 6. Risk evaluation with the effect of malicious recommendations. Other setting options of this experiment group: C1, P1, and S2. (a) CDF of the total number of cooperated cooperators in the history table, (b) CDF of the number of good-known-cooperators, (c) Number of good-known-cooperators versus related average number of cooperations, and (d) Distribution of served services.



(a) (b) (c)

Fig. 7. Long-range effect of the Model. Other setting options of this experiment group: M2, S1, and W4. (a)CDF of total number of cooperated cooperators in the history table, (b) CDF of the number of good-known-cooperators, and (c) Distribution of served services.

- **Setting the recommendation a low weight** is a good tradeoff to improve the performance while keeping the ability of resistance to the malicious recommendations.
- The model is promising in large scale P2P resource sharing systems with suitable settings; however, how to intelligently determine these weights need further study.

IV. Related Work

Our work is built upon a great deal of previous work. Instead of describing all of them, we cluster them into three groups that are specifically related to our work: *trust management*, *reputation-based system*, and *cooperative Web caching*.

a) Trust Management: The notion of "trust management" was first coined by Blaze, Feigenbaum, and Lacy in their seminal paper on decentralized trust management [5], which addresses the authentication of each client request from the perspective of servers (service provider) in terms of security policies, credentials, and trust relationship. This is different from what we proposed, where the trustworthiness of both sides are considered in general, rather than on each individual service request. In the computer science literature, Marsh (1994) is the first one to introduce a computational model for trust in the distributed artificial intelligence (DAI) community [19]. However, he did not model reputation in his work. Mui [21] gives a detailed computational model of trust and reputation. In Mui's model, reputation is well modeled, but it doesn't take the risk into consideration. [12], [24] consider risk assessment for the trust management. Different from these solutions, we make risk as the assessment of the short-term behaviors and treat it as part of the trustworthiness.

b) Reputation-based System: Centralized reputation systems is a very hot topic and has been widely deployed in e-commerce [1], [26], [32], such as eBay (an online auction site) and slashdot.com (an online tech-guru site). Recently, in the P2P domain many decentralized reputation management schemes like P2Prep [7], EigenTrust [13], and NICE project [15] emerge. P2Prep provides a protocol complementing existing P2P protocols. EigenTrust assumes that trust is transitive and addresses the weakness of the assumption and the collusion problem by assuming there are pre-trusted nodes in the system. NICE project [15] discusses the trust inference problems, and [29] proposes a model to build trustworthy software agent. However, the objectives of these reputation-based systems are different from that of our effort, which focuses on the self-policing trustworthiness over other peers, rather than obtaining a global consistent trust value for each peer. But we believe that our work will benefit from these reputation-based systems. In [20], referral (recommendation) is treated as the challenge in the reputation system. Many solutions have been proposed in this field [27], [31], [33], [34], [35], which are good directions to improve the effect of the referral if we want to dig out more potential of the referral.

c) Cooperative Web Caching: The P2P Web server sharing is chosen as a case study to evaluate the efficacy and performance of the proposed personalized trust model. However, this idea is similar to cooperative Web caching, which has been extensively studied in recent years [11], [18], [22]. Different from these previous work, which is from the perspective of client caching (passive mode), P2P Web server sharing is a proactive approach from the perspective of Web servers. More detailed comparison can be found in [25].

V. Conclusions

A novel trust model combining the reputation value calculation and the risk evaluation is proposed here. The preliminary results show that the PET model is promising for resource sharing in the P2P community with large amount of dynamic and noncooperative peers, and malicious recommenders, by assigning a high weight to the risk and a low recommendation weight. Determining these weights intelligently is our next step. Exploring how the dynamic behaviors of peers affect the trustworthiness value and the convergence of the trustworthiness value are also interested research topics. We plan to apply this model to a resource management middleware and deploy it on the Planetlab platform [23].

References

[1] K. Aberer and Z. Despotovic. Managing trust in a peer-to-peer inforamtion systems. *Proceedings of the 10th International Conference on Information and Knowledge Management (CIKM'01)*, 2001.

[2] F. Azzedin and M. Maheswaran. Evolving and managing trust in grid computing systems. *IEEE Canadian Conference on Electrical and Computer Engineering (CCECE '02)*, May 2002.

[3] M. A. Jøsang, E.Gray. Analysing topologies of transitive trust. *In Proceedings of the Workshop of Formal Aspects of Security and Trust (FAST) 2003*, Sept. 2003.

[4] P. Barford and M. E. Crovella. Generating representative web workloads for network and server performance evaluation. *Proceedings of Performance '98/ACM SIGMETRICS '98*, July 1998.

[5] M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized trust management. *IEEE Symposium on Security and Privacy*, May 1996.

[6] R. Buyya, D. Abramson, and J. Giddy. A case for economy grid architecture for service-oriented grid computing. *Proceedings of the 10th IEEE International Heterogeneous Computing Workshop*, Apr. 2001.

[7] F. Cornelli, E. Damiani, S. D. C. Vimercati, S. Paraboschi, and P. Samarati. Choosing reputable servents in a p2p network. *Proc. of the 11th International World Wide Web Conference (2002)*, May 2002.

[8] R. Dingledine, N. Mathewson, and P. Syverson. Reputation in p2p anonymity systems. *Proc. of the 1st Workshop on Economics of Peer-to-PeerSystems*, June 2003.

[9] B. Gross and A. Acquisti. Balances of power on ebay: Peers or unequals. *Workshop on Economics of Peer-to-Peer Systems*, June 2003.

[10] N. Habra, B. L. Chalier, A. Mounji, and I. Math-ieu. Asax: Software architecture and rule-based language for universal audit trail analysis. *European Symposium on Research in Computer Security (ESORIC92)*, 1992.

[11] S. Iyer, A. Rowstron, and P. Druschel. SQUIRREL: A decentralized, peer-to-peer web cache. *Proceedings of the 12th ACM Symposium on Principles of Distributed Computing (PODC 2002)*, July 2002.

[12] A. Jøsang and S. L. Presti. Analysing the relationship between risk and trust. *In Proceedings of the Second International Conference on Trust Management*, Apr. 2004.

[13] S. Kamvar, M. T. Schlosser, and H. Gacia-Molina. The eigentrust algorithm for reputation management in p2p networks. *Proc. of the 12th International World Wide Web Conference (2003)*, May 2003.

[14] A. Lazar and N. Semret. Auctions for network resource sharing. Tech. Rep. TR 467-97-02, Computer Science Department, Columbia University, Feb. 1997.

[15] S. Lee, R. Sherwood, and B. Bhattacharjee. Cooperative peer groups in nice. *Proc. of IEEE Conference on Computer Communications (INFOCOM'03)*, Mar. 2003.

[16] Z. Liang and W. Shi. M-CUBE: A novel economic model for trusted P2P resource sharing. Tech. Rep. MIST-TR-2004-005, Department of Computer Science, Wayne State University, Feb. 2004.

[17] T. F. Lunt. Detecting intruders in computer sys-tems. *Conference on auditing and computer*, 1993.

[18] Y. Mao, Z. Zhu, and W. Shi. Peer-to-peer web caching: Hype or reality? *Proceedings of the tenth International Conferences on Parallel and Distributed Systems*, July 2004.

[19] S. Marsh. *Formalising Trust as a Computational Concept*. Ph.D. thesis, University of Stirling, 1994.

[20] M.P.Singh. Trustworthy service composition: Challenges and research questions. *Proceedings of the 1st International Joint Conference on Autonomous Agents and MultiAgent System (AAMAS)*, July 2002.

[21] L. Mui. *Computational Models of Trust and Reputation: Agents, Evolutionary Games, and Social Networks*. Ph.D. thesis, Massachusetts Institute of Technology, Dec. 2002.

[22] V. Padmanabhan and K. Srpanidkulchai. The case for cooperative networking. *Proc. of the 1st International Workshop on Peer-to-Peer Systems (IPTPS'02)*, Feb. 2002.

[23] L. Peterson, T. Anderson, D. Culler, and T. Roscoe. A blueprint for introducing disruptive technology into the internet. *ACM First Workshop on Hot Topics in Networks (HotNets-I)*, Oct. 2002, `http://www.planet-lab.org/pubs/hotnets.pdf`.

[24] A. A. Rahman and S. Hailes. Supporting trust in virtual communities. *In Proceedings of 33rd Hawaii International Conference on System Sciences*, Jan. 2000.

[25] J. Ravi, Z. Liang, and W. Shi. A case for peer-to-peer web server sharing. Tech. Rep. MIST-TR-2003-010, Department of Computer Science, Wayne State University, Nov. 2003.

[26] P. Resnick, R. Zeckhauser, E. Friedman, and K. Kuwabara. Reputation systems. *Communications of the ACM* 43(12):45–48, 2001.

[27] J. Sabater and C. Sierra. Regret: Reputation in gregarious societies. *ACM SIGecom Exchanges* 3, 2002.

[28] S. E. Smaha and J. Winslow. Misuse detection tools. *Journal of Computer Security* no. 1, p. 3949, 1994.

[29] A. S.Patrick. Building trustworthy software agents. *IEEE INTERNET COMPUTING* pp. 46-53, Nov. 2002.

[30] C. Waldspurger, T. Hogg, B. Huberman, J. Kephart, and W. Stornetta. Spawn: A distributed computation economy. *IEEE Transactions on Software Engineering* 18(2):103–117, 1992.

[31] Y. Wang and J. Vassileva. Trust and reputation model in peer-to-peer networks. *Third International Conference on Peer-to-Peer Computing (P2P'03)*, Sept. 2003.

[32] L. Xiong and L. Liu. A reputation-based trust model for peer-to-peer ecommerce communities. *Proceedings of the IEEE Conference on E-Commerce*, June 2003.

[33] B. Yu and M.P.Singh. Searching social networks. *Proceedings of the 1st International Joint Conference on Autonomous Agents and MultiAgent System (AAMAS)*, July 2002.

[34] B. Yu and M.P.Singh. Searching social networks. *Proceedings of the 2nd International Joint Conference on Autonomous Agents and MultiAgent System (AAMAS)*, July 2003.

[35] B. Yu and M. P. Singh. A social mechanism of reputation management in electronic communities. *In Proceedings of Fourth International Workshop on Cooperative Information Agents*, 2000.