

# A Probabilistic Approach to Predict Peers' Performance in P2P Networks\*

Zoran Despotovic and Karl Aberer

EPFL - Swiss Federal Institute of Technology  
Lausanne, Switzerland

**Abstract.** The problem of encouraging trustworthy behavior in P2P online communities by managing peers' reputations has drawn a lot of attention recently. However, most of the proposed solutions exhibit the following two problems: huge implementation overhead and unclear trust related model semantics. In this paper we show that a simple probabilistic technique, maximum likelihood estimation namely, can reduce these two problems substantially when employed as the feedback aggregation strategy. Thus, no complex exploration of the feedback is necessary. Instead, simple, intuitive and efficient probabilistic estimation methods suffice.

## 1 Introduction

Recent empirical studies have shown that much of eBay's commercial success can be attributed to its reputation mechanism (Feedback Forum) as a means of deterring dishonest behavior. Thus, [1] shows that "reputation profiles are predictive of future performance", while [2] and [3] come up with the conclusion that Feedback Forum completely fulfills its promises: the positive feedback of the sellers increases their price, while the negative one reduces it.

eBay's Feedback Forum is just a well known example of *reputation systems* [4] as informal social mechanisms for encouraging trustworthy behavior in online communities. Their key presumptions are that the participants of an online community engage in repeated interactions and that the information about their past doings is informative of their future performance and as such will influence it. Thus, collecting, processing, and disseminating the feedback about the participants' past behavior is expected to boost their trustworthiness. The mentioned eBay example confirms this expectation.

A huge body of work has appeared recently on managing online reputations. (For a comprehensive overview see [5] for instance). In this paper we will be more specific and consider only P2P networks. We will first review the relevant literature (Section 3) and offer a view on how various P2P reputation management approaches contribute to building trust. As we will see, most of the them

---

\* The work presented in this paper was partly carried out in the framework of the EPFL Center for Global Computing and supported by the Swiss National Funding Agency OFES as part of the European project Evergrow No 001935.

suffer from the following two problems: huge implementation overhead and unclear trust related model semantics. The main cause of the first problem lies in the necessity of aggregating the feedback about all peers in the network in order to assess the trustworthiness of a single peer, while the second problem is mainly caused by the counterintuitive feedback aggregation strategies resulting in the outputs that are hard to interpret. In this paper we show on two settings relevant for P2P communities that a simple probabilistic technique, maximum likelihood estimation namely, can reduce these two problems substantially when employed as the feedback aggregation strategy (Sections 4 and 5). Operating on a small fraction of the feedback available in the network, it lends itself to an efficient implementation. On the other hand, its outputs are probabilities of specific behaviors of the peers and as such have a clear and well founded interpretation. Finally, its ability to detect peers' misbehavior is as strong as that of the best ones of the existing approaches. Thus, we conclude that no complex exploration of the feedback is necessary. Instead, simple and efficient probabilistic estimation methods suffice.

Interestingly, the vast majority of the existing approaches (excluding game-theoretic works) evaluate their effectiveness assuming that the peers are associated with probability distributions determining their performance. However, this assumption on the peer behavior is not explicitly built into the solution methods they propose. On the other hand, we start with the same assumption and, instead of aggregating the feedback in an ad hoc manner, we just try to predict the peer probability distributions as well as we can. This is the key difference between our and the other solutions.

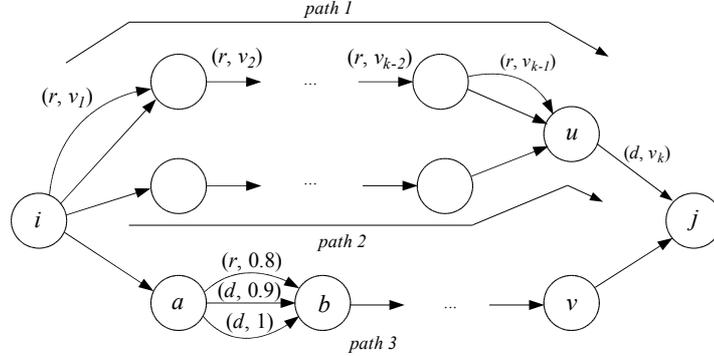
However, the exact setting in which the technique can be successfully used rules out forming huge collusive groups among peers. Instead, it implies their independent acting.

## 2 P2P Computational Models of Trust

In the following we introduce a general view on P2P computational models of trust, broad enough to cover all specific works we are aware of. The purpose of the section is to describe in general the problem the decentralized trust and reputation management considers. Section 3 then describes particular classes of solutions of the problem.

An underlying assumption of the computational models of trust is that the peers engage in bilateral interactions, which results in forming a directed weighted (trust) multigraph. Its node set coincides with the set of peers and the set of edges with the set of interactions between the peers. In a general form, the weight assigned to any edge consists of an ordered pair: a flag representing the context of the corresponding interaction and the interaction outcome as perceived by the source. The set of all interaction outcomes is assumed to be common knowledge among the peers. We assume at most two possible interaction contexts: (1) recommendations, when the destination node acts as a recommender of other nodes capable of performing a specific task or other rec-

ommenders, and (2) the task performances themselves. We also assume that the latter must be present while the former is optional.



**Fig. 1.** Computational model of trust

Figure 1 presents an example. The way we should understand this figure is as follows. Node  $a$  had three interactions with node  $b$ : once node  $b$  acted as a recommender of other entities (flag  $r$ ) and node  $a$ 's contention with the recommendation was evaluated 0.8 and twice node  $b$  provided the service in question to node  $a$  (flag  $d$ ) and  $a$ 's evaluations of the service provisions were 1 and 0.9 respectively.

The core of any computational model of trust is in the answer to the following question: how can a given peer use the information on direct and recommendation experiences between the peers, that it can retrieve from the network, to evaluate the trustworthiness of any other peer? A bit of caution is needed here: we do not say that every peer knows the whole trust multigraph. Instead, we assume that it is reconstructed (the whole graph or a part of it) by querying the other peers, which may misreport. Generally, the direct experiences of the nodes which interacted with the given node (nodes  $u$  and  $v$  in the case of node  $j$  from Figure 1) should be propagated through the graph down to the computation source (node  $i$ ) by using the recommendation experiences along the paths to filter them out. Different works propose different strategies for doing this. We classify them in Section 3. We stress that most of the existing works do not model explicitly the context of recommendations but rather use direct experiences as filters. This can be thought of as weighting one's reports by his trustworthiness rather than his ability to recommend. As well, we emphasize that most of the works use binary (zero-one) evaluations of the individual interactions. In the example of Figure 1 this would mean that all the numbers are zero or one.

It is important to note that the formed trust graph does not coincide with the underlying P2P network. We see the underlying P2P system as managing a distributed database of (*data key, data value*) pairs. The problem is how to distribute this database among the peers so that the basic database opera-

tions (e.g. search) are efficient and the storage space required at each peer is small in comparison with the size of the database. Two fundamental approaches exist to achieve this: (1) unstructured [6], in which the data is distributed randomly over the peers and broadcasting mechanisms are used for searching and (2) structured [7], [8], [9], that build up distributed, scalable data access structures to route search requests. Having clarified this, it should be clear that the trust graph can be actually stored in the underlying P2P system. In the case of an unstructured P2P overlay every peer can store its outgoing edges from the trust graph (the identifier of the destination node and possibly time stamp may act as the key), while in the case of a structured P2P overlay the triples (*destination, source, timestamp*) may act as the keys for the trust graph edges and be stored at peers just as dictated by the P2P network [10]. In both cases weights of the edges may act as the values. Thus, exploring the trust graph reduces actually to searching the underlying P2P network. More specifically, retrieving feedback about any specific peer is subdued to searching for the data items with the keys starting with that peer’s identifier. This can be done efficiently in a structured P2P network. In the rest of the paper we will assume this.

### 3 Trust Models Classification

#### 3.1 Classification Criteria

A clear categorization of P2P computational models of trust based on managing peers’ reputations must consider in the first place the models’ behavior with respect to the following three dimensions: the incurred implementation costs, the resistance to various attacks and the trust related model semantics.

As P2P networks normally involve millions of nodes particular attention should be paid to cutting down the total implementation overhead introduced by the employed reputation management solution. It consists of: the communication costs associated with the process of retrieving the necessary feedback, the involved storage costs, and the computation overhead related to the feedback aggregation. Resistance to attacks normally implies an analysis of the model responsiveness to various forms of misbehavior of the peers. The following two types of misbehavior have been established in the literature as relevant: independent cheating in interactions or badmouthing other peers and forming collusive groups to badmouth specific peers and boost trust values of some other ones. For example, peers  $u$  and  $v$  in Figure 1 may misreport their experiences with peer  $j$  independently or they may collude and misreport in correlated ways.

The last mentioned dimension deserves more explanation as it is accompanied by quite some disagreement in the literature. Deriving from [11], we view trust as being inseparable from vulnerability and opportunism associated with the interacting parties. Consequently, we say that peer A (trustor) trusts peer B (trustee) if the interaction generates a gain to be shared with and by peer B and exposes peer A to a risk of loss, if peer B takes a too large portion from the joint gain. Building on this, we see trust management as a set of actions related

to: 1) reducing the opportunism of the trustee, 2) reducing vulnerability of the trustor and, after these two issues have been properly addressed, 3) deciding if and when to enter an interaction.

Clearly, the main goal of any reputation management mechanism is partial or, if possible, complete reducing of the opportunism of the interacting parties. The degrees at which different mechanisms achieve this vary. We see the following classes: *social networks formation, probabilistic estimation techniques and game-theoretic models*, which are discussed next.

### 3.2 Social Networks

This class of approaches normally implies that the entire trust multigraph is reconstructed and that all reputation information available in it is aggregated (Figure 1). A natural interpretation of the aggregation process involves the following steps: 1) enumerating all paths from the trust computation source to the target node, 2) aggregating the trust values along the paths to give a path wide gossip and 3) merging these gossips into a final value.

Where is the exact position of this class with respect to the tree dimensions introduced above?

Normally, their implementation overhead is high. Because a trust-computing node has to retrieve the entire trust network, the communication costs are very high. Besides, because the number of paths between the trust-computing source and target can be exponential, the reputation aggregation process is too costly. These two problems are even more strongly emphasized if the context of recommendation is present - see [12] for an example.

[13], modeling only direct experiences, offers important theoretical insights on this issue by characterizing the combinations of path and across-path aggregation strategies that may lead to a non-exponential trust computation algorithm (we note that many other works use such combinations: e.g. [14], used for Web pages ranking, and [15]). Central to the approach is the claim that, for specific combinations of the aggregation strategies, exploring all the paths is equivalent to finding a convergent power of the trust matrix, derived naturally from the trust graph.<sup>1</sup> (Here, the matrix “multiplication” operation is derived from the path and across-path aggregation operations.) However, the proposed algorithm requires the synchronous participation of all peers, making it hardly implementable in a P2P network. Instead, we believe that an incremental computation is something worth further investigation.

[16] offers important insights with respect to this. The gist of this approach consists of computing the trustworthiness of a given node as the average of its performances as seen by its neighbors in the trust graph, weighted by the trustworthiness of the neighbors themselves. The authors also develop a simple caching scheme in which the trust values of the neighbors of the trust computation target are taken from a cache (default values are used in the case of cache

---

<sup>1</sup> It is assumed that the interaction multigraph is transformed into a graph by aggregating first the interaction outcomes between the pairs of nodes.

miss) and their computed trust values replace the corresponding values existing in the cache.

How robust is this class of approaches in presence of various misbehaviors? [15] and [16], the only works providing informative simulation results, report good performance of the corresponding approaches when the fraction of malicious peers is small (below 45% approximately) and the malicious peers independently cheat in the interactions and distort their ratings of other peers. [16] further reports the complete breakdown of the mechanism when the cheaters take more than a half of the overall population or when they collude. On the contrary, [15] claims almost full effectiveness of their mechanism when the malicious peers make the larger fraction of the population and collude in various ways. This results from, in our opinion, the fairly unrealistic assumption that a number of pretrusted peers exist each of whom is assigned some non-zero trust by the rest of the community, including the malicious peers. (This is so called “random walker” model used for Web pages ranking.)

The computed values have unclear semantics and are hard to interpret. They cannot be interpreted as the (estimated) probabilities of the trustworthy behavior of the target peers and the question what exactly they represent is left open. Let us also mention an interesting detail related to [13] and [15]: when the trust graph is irreducible and aperiodic the powers of the corresponding trust matrix converge to a matrix in which all the rows are the same and sum up to 1 (the primary eigenvector of the matrix). Thus the trust values of the peers have global meaning - they are independent of the computation source. On the other hand, because all the values sum up to 1, it seems as if the trust was distributed among the peers. But, if we have the values for all the peers and they are approximately close we are in doubt whether the whole network is trustworthy or it is malicious.

This leads us to conclude that the computed values lack a plausible interpretation on an absolute scale and that the only scenarios in which they can be used must involve ranking the trust values of many peers and selection of the most trustworthy one(s) among them.

### 3.3 Probabilistic Estimation

Probabilistic estimation techniques present certain improvement with respect to the meaningfulness of the computed values. Namely, they output probability distributions (or at least the most likely outcome) over the set of possible behaviors of the trusted agents. The importance of such models becomes clear if we recall the presented view on trust - if the opportunism of the trustee cannot be reduced completely then it becomes important for the trustor to be able to estimate the risks of the interaction and decides whether to enter it or not. If the individual outcomes of the interaction are assigned the probabilities and the trustor can assign them utilities as well then this task becomes easy - the trustor just needs to compute whether entering the interaction has a higher utility than staying out.

In principle, it is possible to construct a probabilistic model in which all the paths between the trust computation source and target. However, we are not aware of any such attempt and doubt that it would suffer from an exponential computation overhead, just as outlined previously. But, we believe that constructing such a model is unnecessary and that in most of the relevant settings it is sufficient to consider only two small fractions of the formed trust (multi)graph - those around the trust computation source and target. One of the goals of this paper is exactly to show this. Needless to say, another clear advantage of doing the trust computation this way is its implementation efficiency.

It is a bit surprising that very few works on using well known probabilistic estimation techniques for decentralized trust computation exist. [17] presents the well-known method of Bayesian estimation as the right probabilistic tool for assessing the future trusting performance based on past interactions. Only direct interactions were studied - the question of including recommendations was not considered.

[18] goes a step further by taking into account the "second-hand" opinions also. However, the strategy for merging own experiences with those of other witnesses is intuitive (giving more weight to own experiences, though plausible, is still intuitive) rather than theoretically founded.

### 3.4 Game-Theoretic Models

Game-theoretic reputation models make a further clarification in the interpretation of the agents' trustworthiness in the sense that, if the reputation system is designed properly, trust is encoded in the equilibria of the repeated game the agents are playing. Thus, for rational players trustworthy behavior is enforced. The real challenge here is how to define the feedback aggregation strategies that will lead to socially desirable outcomes carrying trust.

[19] presents the proper game-theoretic framework for analyzing reputations (repeated games with incomplete information), while [20] and [21] offer certain characterizations of the equilibria payoffs in the presence of reputation effects.

[22] focuses on a specific game and derives its equilibria. Apart from this the author also raises questions concerning the overall game-theoretic reputation systems design, such as incentivizing players to leave feedback, dealing with incomplete feedback etc. However, an underlying assumption of this work is that a central trusted authority does the feedback aggregation. We see this as a major obstacle to transferring game-theoretic models to decentralized environments.

## 4 Model I - Honest or Dishonest Peers

Let us now consider a P2P network consisting of peers having associated innate probabilities of performing honestly in their interactions with others. Let  $\theta_j$  denote the probability of peer  $j$ . Assume that peer  $j$  interacted with peers  $p_1, \dots, p_n$  and its performances in these interactions were  $x_1, \dots, x_n$ , where  $x_i \in \{0, 1\}$  (1

denoting the honest performance and 0 the dishonest one). When asked to report on peer  $j$ ' performances witnesses  $p_1, p_2, \dots, p_n$  may lie and misreport. Assuming that they lie with specific probabilities, say  $l_k$  for peer  $p_k$ , the probability of observing report  $y_k$  from peer  $p_k$  can be calculated as:

$$P[Y_k = y_k] = \begin{cases} l_k(1 - \theta_j) + (1 - l_k)\theta_j & \text{if } y_k = 1 \\ l_k\theta_j + (1 - l_k)(1 - \theta_j) & \text{if } y_k = 0. \end{cases} \quad (1)$$

Now, given a random sample of independent reports  $y_1, y_2, \dots, y_n$  we have that the likelihood function of this sample is

$$L(\theta_j) = P[Y_1 = y_1]P[Y_2 = y_2] \cdots P[Y_n = y_n]. \quad (2)$$

The maximum likelihood estimation procedure now implies simply finding  $\theta_j$  that maximizes this expression. This number is the maximum likelihood estimate of the unknown probability. Note also that the own experiences are seamlessly integrated into this model - the trust computing source peer  $i$  just has to put  $p_i = 1$  for his own experiences  $x_i$ .

Let us locate this model in the three-dimensional space introduced in Section 3.1.

First, referring back to Figure 1, if peer  $i$  is computing the trustworthiness of peer  $j$  then the model assumes the peer  $i$  first retrieves from peers  $u$  and  $v$  their reports on peer  $j$ 's performances with them. (This is the meaning of  $y_1, y_2, \dots, y_n$  in (2)). Thus, the necessary reputation information on which the model operates consists in this case of the edges entering node  $j$ . Keeping in mind what we said in Section 2 about how the overlay network is used to store the trust data we see that retrieving these edges (feedback) coincides with searching the overlay for the data items with the keys starting with  $j$ . Further, as we will see shortly, not all such data items have to be retrieved. Good predictions can be achieved even with 10-20 reports retrieved. We stress that this is a considerable improvement as compared to what most of the existing approaches do - we retrieve only a small fraction of the feedback about the trust computation target, while they retrieve the entire feedback about all the nodes.

On the other hand, we assume that peer  $i$  deduces the misreporting probabilities  $l_k$  from own interactions (by comparing its own performances with the reports about them). These "averages" can be maintained for specific peers ( $l_k$ s are different) or at the level of the whole network ( $l_k$ s are all same). In the simulations below we assume the second possibility. In this case the involved storage costs per peer are negligible - only a single value is kept and used to approximate the situation in the network.

Thus, in a word, the model incurs small communication overhead and virtually no storage costs.

Second, the output values of the model are probabilities and as such they do have a plausible interpretation on the absolute  $[0, 1]$  scale. Therefore, it is easy to interpret and use them without comparing with the other peers' values.

Third, Equation (2) implies the independence of the reports  $Y_1, Y_2, \dots, Y_n$ . Thus, the assumed setting is non-collusive and the simulation results we present

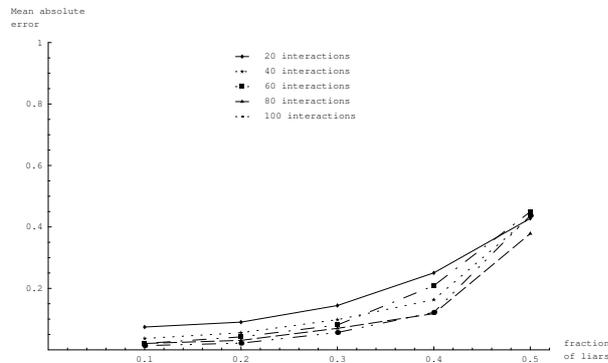
next hold for this setting. We note that extensions to collusive settings are possible by integrating the collusion possibilities into (2).

#### 4.1 Simulation Results

We checked the performance of the method in a variety of parameter settings. As the estimation quality measure we chose the mean absolute error of the estimated probabilities of the peers performing honestly and their actual values. The following parameters are considered: 1) the number of peers - 128 (constant throughout the simulations), 2) the number interactions per peer - varied at increments of 20 from 20 to 100, and 3) fraction of liars - varied at increments of 0.1 from 0.1 to 0.5. All the results are averaged across 20 simulation runs. The interactions among the peers were generated at random, we did not consider any particular structure of the resulting trust network.

Note that the second parameter, number of interactions per peer, is not correlated with the numbers of peers. For higher network sizes the same results would be obtained with the same numbers of interactions per peer. Put differently, the absolute amount of feedback is what determines the results, not its relative size as compared to the size of the network.

Figure 2 shows the results for the case when the peers' probabilities of performing honestly are generated at random in the interval  $[0, 1]$ . Note that the liars were generated so that they always lie ( $l_k$  from (1) equals the fraction of liars). We have also experimented with varying  $l_j$ s and did not observe any important difference.



**Fig. 2.** Simulation results - peers' trustworthiness drawn randomly from  $[0,1]$

We emphasize that the plot is symmetric across the line "fraction of liars = 0.5". This is simply a consequence of the introduced probabilistic assumptions and non-collusive peer behavior - if a peer believes that the majority of the peers

are liars then it should take the reverse of their reports as true. Interestingly, no existing approach exhibits this behavior.

## 5 Model II - Normally Distributed Services

There are settings in which the peers' behavior or their performances within certain contexts are characterized by a globally known finite number of types. An example of such a setting would be the speeds of the network connections of the peers. Namely, there are a fixed number of the connection types with prespecified speeds. (However, the speed of any given connection is only theoretically fixed, in practice it varies due to different conditions in the network.) In this section we present a general model of such settings and show how the maximum likelihood estimation can be used in this case. The simulation results, given in Section 5.2, show that the technique performs even better than in the setting of Section 4. Intuitively, this is a consequence of the stronger constraints imposed on the peers behavior.

Consider a P2P network consisting of  $N$  peers  $p_1, \dots, p_N$ . They provide services to one another and we are assuming that each observed service quality is distributed according to a normal distribution  $\mathcal{N}(\mu_i, \sigma)$ ,  $1 \leq i \leq N_s$ . Thus, there are  $N_s$  different distribution from which the service qualities are drawn, all of them having the same standard deviation  $\sigma$  but different means  $\mu_1, \dots, \mu_{N_s}$ . These distributions will be denoted by  $\mathcal{N}_i$  so that, for any  $1 \leq k \leq N_s$  the symbol  $\mathcal{N}_k$  will denote the normal distribution  $\mathcal{N}(\mu_k, \sigma)$ .

We further assume that the means are innate to the peers so that whenever a given peer, say peer  $p_j$ , provides a service then the service quality observed by its consumer will be a random variable distributed according to  $\mathcal{N}(\mu_k, \sigma)$ , where  $\mu_k$  is peer  $p_j$ 's mean.

As in the precedent model, we are assuming that a fraction of the peer population  $p_l$  are liars, while the rest of the populations are honest participants. The fraction of liars is learned by comparing the own performances and the reports about them. We will assume the following behavioral patterns of the two groups of peers. Consider an interaction between service provider  $p_i$  and consumer  $p_j$  in which  $p_j$  observes the quality level  $x$ . If peer  $p_j$  is honest then it will report value  $x$  when asked to report on peer  $p_i$ 's. On the other hand, if  $p_j$  is a liar then it will first choose a distribution  $\mathcal{N}_k$  at random and then report a random value generated from this distribution.

Our task is, just as before, to show that a maximum likelihood based estimation method can be used to accurately predict the future performance of the service providers given the reports of their past provided qualities. Note that a very similar setting was considered in [23], where a machine learning technique was used instead.

### 5.1 Estimation Derivation

Let  $\theta$  denote the unknown mean of a given peer. To determine the likelihood of any report on that peer's performance we must first determine the distribution

of the reports. Denoting by  $Y$  the corresponding random variable we have:

$$P[Y \leq y] = p_l P[Y \leq y | \text{false report}] + (1 - p_l) P[Y \leq y | \text{correct report}], \quad (3)$$

where

$$P[Y \leq y | \text{correct report}] = \sum_{i=1}^{N_s} P[Y \leq y | Y \text{ is } \mathcal{N}(\theta, \sigma)] \quad (4)$$

and

$$P[Y \leq y | \text{false report}] = \sum_{i=1}^{N_s} \frac{1}{N_s} P[Y \leq y | Y \text{ is } \mathcal{N}_i]. \quad (5)$$

Taking derivatives we obtain the following probability density function:

$$f_Y(y) = p_l \sum_{i=1}^{N_s} \frac{1}{N_s} \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(y-\mu_i)^2}{2\sigma^2}} + (1 - p_l) \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(y-\theta)^2}{2\sigma^2}}. \quad (6)$$

Now, having a random sample  $y_1, y_2, \dots, y_n$  from this distribution and assuming that the independence of the samples we compute the likelihood function:

$$L(\theta) = f_Y(y_1) f_Y(y_2) \cdots f_Y(y_n). \quad (7)$$

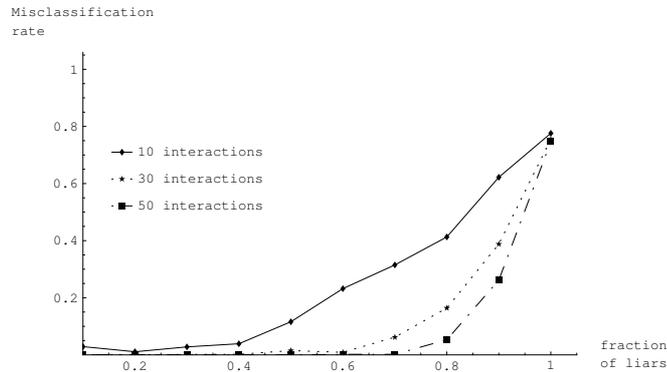
The final task is to select the unknown parameter  $\theta \in \{\mu_1, \mu_2, \dots, \mu_{N_s}\}$  such that this likelihood function is maximized.

We stress that whatever has been said in Section 4 on the the model's main properties applies also here. In short, low implementation overhead and meaningful interpretation of the output values remain the main properties of this model too. Its prediction quality is given next.

## 5.2 Simulation Results

We now present the results from a series of tests that show the performance of the method. We first generate peers, determine the liars and assign service distributions to the peers. The services are generated in such a way that, if there are  $N_s$  services then their means are random numbers between 1 and  $N_s$  (the standard deviation was varied between 0 and 1 as presented below). Being a liar and having any specific service are not correlated. Then we generate a number of interactions at random. The following parameters are considered in the simulations: 1) the number of peers - 128 (constant throughout the simulations), 2) the number interactions per peer - varied at increments of 20 from 10 to 50, 3) fraction of liars - varied at increments of 0.1 from 0.1 to 1.0, 4) number of services - 4 (constant throughout the simulations) and 5) standard deviations of the service distributions - varied at increments of 0.1 from 0.1 to 1.0 with the default value of 0.3. The fraction of incorrect guesses (misclassification rate) is used as the quality measure.

Figure 3 presents the dependency of the misclassification rate on the fraction of liars and the number of interactions per peer considered. The number of services was 4 and their means were all 0.3.



**Fig. 3.** misclassification rate of the expected service quality as a function of the fraction of liars. Standard deviations of the service distributions -  $\sigma = 0.3$ .

Figure 4 shows how the misclassification rate depends on the standard deviations of the services.

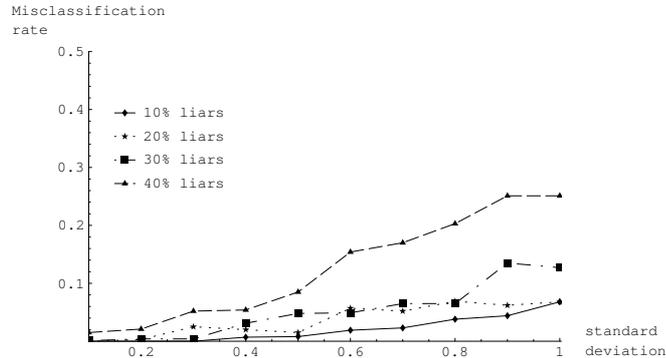
Let us comment on the presented results. First, because the number of the services is greater than two and the liars lie uncoordinatedly good predictions can be made even when the fraction of the liars is close to one. Second, the method becomes very robust as the number of interactions considered increases. When the influence of the "noise" (standard deviation  $\sigma$ ) is not too high, the method is able to separate the distributions quite well.

Though the simulation setting considered in this section is different from the setting of Section 4 we can still derive some conclusions by comparing the presented results. The main difference between the two models is as follows: the model in the previous section assumes infinitely many behavior types (probability distributions), while the model in this section implies a known finite number of behavior types. It is this difference that enables a better performance of the maximum likelihood estimation in the second case.

## 6 Discussion and Future Work

We propose in this paper a simple probabilistic method to assess peers' performance in a P2P network (or more specifically their trustworthiness). The method is based on the well known statistical estimation technique - maximum likelihood estimation. We test its quality on two settings relevant for P2P communities. From the test we derive the following as the main properties of the method.

1. The method is able to estimate peers' characteristics (i.e. their trustworthiness) quite well. For smaller fractions of liars, the method gives good estimates even when a small number of recent interactions is considered (around 20). This is particularly important when the assumed peer behavior



**Fig. 4.** misclassification rate of the expected service quality as a function of the standard deviation of the distributions of the services.

is time dependent. In this setting, our method is more capable of capturing the behavioral dynamics.

2. The implementation overhead of the method is as small as possible. Practically, it only implies the small communication overhead related to retrieving the direct experiences of the peers who interacted with the trust computation target.
3. The considered setting was non-collusive; it assumed that peers did not form collusive groups but rather acted independently. In this setting, the mechanism gives estimates of the peers' trustworthiness with errors within 5-10% even with 30% of liars.

To the best of our knowledge, no existing approach exhibits such properties. At best, they show similar quality of the trustworthiness estimation (similar or higher estimation error) but require substantially higher implementation overhead. On the other hand, most of them can be applied only in specific settings (e.g. file sharing) because they essentially need ranking of the trustworthiness values of various peers. Our comparison of the results given in [15], [16] and [23] and those presented here confirms these claims directly. There is a simple explanation of this. Namely, the simulation settings we consider coincide (or are very similar) with those presented in the mentioned works. This is true not only for the simulation parameters but also for an important aspect of the assumed behavior of the peers. Precisely, all the works assume that the peers are associated with specific probability distributions that determine their performance. The key difference between our and the other solutions is that we introduce this assumption explicitly into the trust computation method, while the other works do not.

However, we did not deal in this paper with the case in which these probability distributions are correlated among the peers. Put differently, we assumed

here independent acting of the peers, without forming collusive groups. Checking precisely the performance of the presented method in collusive settings and extending it to be as effective as possible under this assumption on the peers' behavior make the most important part of the future work. This can be done either by modeling collusions probabilistically (operating directly on Equations (2) and (7)) or learning the probabilities of misreporting for every peer in the network separately rather than for the network as a whole.

## References

1. Resnick, P., Zeckhauser, R.: Trust among strangers in internet transactions: Empirical analysis of ebay's reputation system. In Baye, M.R., ed.: *The Economics of the Internet and E-Commerce*. Volume 11 of *Advances in Applied Microeconomics*. Amsterdam, Elsevier Science (2002)
2. Houser, D., Wooders, J.: Reputation in auctions: Theory and evidence from ebay. Working paper, University of Arizona (2001)
3. Melnik, M.I., Alm, J.: Does a seller's ecommerce reputation matter? evidence from ebay auctions. *Journal of Industrial Economics* **50(3)** (2002) 337–349
4. Resnick, P., Zeckhauser, R., Friedman, E., Kuwabara, K.: Reputation systems. *Communications of the ACM* **43(12)** (2000) 45–48
5. Dellarocas, C.: The digitization of word-of-mouth: Promise and challenges of online reputation systems. Working paper, MIT (2002)
6. Clip2: The gnutella protocol specification v0.4 (document revision 1.2). [http://www9.limewire.com/developer/gnutella\\_protocol.0.4.pdf](http://www9.limewire.com/developer/gnutella_protocol.0.4.pdf) (2001)
7. Aberer, K.: P-grid: A self-organizing access structure for p2p information systems. In: *Proceedings of the Sixth International Conference on Cooperative Information Systems (CoopIS 2001)*, Trento, Italy (2001)
8. Stoica, I., Morris, R., Karger, D., Kaashoek, F., Balakrishnan, H.: Chord: A scalable peer-to-peer lookup service for internet applications. In: *Proceedings of the 2001 ACM SIGCOMM Conference*. (2001) 149–160
9. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Shenker, S.: A scalable content-addressable network. In: *Proceedings of ACM SIGCOMM '01*. (2001) 161–172
10. Aberer, K., Despotovic, Z.: Managing trust in a Peer-2-Peer information system. In: *Proc. of the IX International Conference on Information and Knowledge Management*, Atlanta, Georgia (2001)
11. Usunier, J.C.: Trust management in computer information systems. Working paper, IUMI, HEC, University of Lausanne, Switzerland (2001)
12. Beth, T., Borcharding, M., Klein, B.: Valuation of trust in open networks. In: *Proceedings of the European Symposium on Research in Computer Security (ESORICS)*, Brighton, UK, Springer-Verlag (1994) 3–18
13. Richardson, M., Agrawal, R., Domingos, P.: Trust management for the semantic web. In: *Proceedings of the Second International Semantic Web Conference*, Sanibel Island, FL (2003) 351–368
14. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank citation ranking: Bringing order to the web. Technical report, Stanford University, Stanford, CA (1998)
15. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: Eigenrep: Reputation management in p2p networks. In: *Proceedings of the World Wide Web Conference*, Budapest, Hungary (2003)

16. Xiong, L., Liu, L.: Peertrust: Supporting reputation-based trust in peer-to-peer communities. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, Special Issue on Peer-to-Peer Based Data Management (2004)
17. Mui, L., Mohtashemi, M., Halberstadt, A.: A computational model of trust and reputation. In: *Proceedings of the 35th Hawaii International Conference on System Science (HICSS)*. (2002)
18. Buchegger, S., Le Boudec, J.Y.: The effect of rumor spreading in reputation systems for mobile ad-hoc networks. In: *Proc. of WiOpt '03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, Sophia-Antipolis, France (2003)
19. Kreps, D., Wilson, R.: Reputation and imperfect information. *Journal of Economic Theory* **27** (1982) 253–279
20. Fudenberg, D., Levine, D.: Reputation and equilibrium selection in games with a patient player. *Econometrica* **57**(4) (1989) 759–778
21. Fudenberg, D., Levine, D.: Maintaining a reputation when strategies are imperfectly observed. *Review of Economic Studies* **59** (1992) 561–579
22. Dellarocas, C.: Efficiency and robustness of binary feedback mechanisms in trading environments with moral hazard. Working paper 4297-03, MIT (2003)
23. Sen, S., Sajja, N.: Robustness of reputation-based trust: Boolean case. In: *Proceedings of AAMAS02*, Bologna, July (2002)