A Simulation Model for Trust and Reputation System Evaluation in a P2P Network

Roberto Aringhieri and Daniele Bonomi

Summary. A peer-to-peer (P2P) network is an exchange community of anonymous peers or individuals. The evolution of P2P network has determined the need of trust and reputation systems (TRS) in order to improve the reliability of local interactions: collecting in some way the local experiences, the TRS assesses the possibility that an individual has a malicious behavior, i.e., he cheats other individuals.

In this paper we present an agent-based simulation model for the evaluation of a generic TRS within a decentralized P2P network. We describe some minimal requirements that, in our opinion, every P2P simulators should have. Moreover, we propose a complete model of peers in which the behavior of both good and malicious peers is accurately defined.

Key words: Agent-based simulation, Trust and reputation system, P2P network.

A peer-to-peer (P2P) network is an exchange community of anonymous peers or individuals. Each individual can join or leave the community freely. When joined, it plays the role of client and server at the same time. Such a community is characterized by the absence of a central authority (decentralized environment) and by the fact that each individual has a local view of the whole community. A global behavior can emerge from local interactions, i.e., exchanges between pairs of individuals.

The evolution of P2P network has determined the need of trust and reputation systems (TRS) in order to improve the reliability of local interactions: collecting *in some way* the local experiences, the TRS assesses the possibility that an individual has a *malicious* behavior, i.e., he cheats other individuals. The way of collecting or aggregating the local experiences defines different TRS (see e.g. [5, 9, 11, 14, 18]).

In this paper we present a simulation model for the evaluation of a generic TRS within a decentralized P2P network. In Sect. 1 we review four different TRSs, for which a simulation model or a numerical evaluation is reported, in order to define some minimal requirements for a generic simulation model. From this review, three main components of a simulation model are highlighted, that is the network model, the content distribution model and the peer behavior model. The latter is discussed in Sect. 2. The simulation model is described in Sect. 3. Conclusions are discussed in Sect. 4.

1 Model Requirements

Reputation has been extensively studied in economics especially using game theory to model the behavior of economic agents acting in a market to maximize their utilities. Usually, the agents work under the incomplete information and the "looking forward" assumptions. The latter consists in maximizing the agent utility considering a long time period using past events to predict the future needs of the agent thus modifying the current utility.

The need of improving the reliability of local interactions among peers determines the increasing relevance of trust and reputation topic in the field of P2P research. A typical successful example is the "Feedback Forum", the eBay reputation mechanism [1], which is deeply analyzed in literature [16,18].

The problem of managing trust in a decentralized environment is formally described in [2]. Here, we simply describe the basic TRS environment. Each peer stores its *local reputations*, i.e., the result of the interactions with other peers. These reputations are usually represented by values in $[0, 1]$, where 0 and 1 represent, respectively, the worst and the best reputation. To decide if an offerer peer j is trustworthy or not, the peer i uses its local reputation about j or, when i has not previous experience with j or the local reputation is still considered not reliable enough, it tries to obtain the *global reputation* of j from the community. The global reputation is usually a value obtained by aggregating the local reputation of other peers about j.

In [10], the authors review and describe several tools to manage TRS. They identify three broad classes of tools: social network formation, probabilistic estimation techniques and game-theoretic reputation models. For each tools they analyze, when possible, the "trust related model semantics" and the "incurred implementation costs." In particular, the latter concerns the performance analysis and the implementation overhead.

In our review, we consider the "EigenTrust" algorithm [14], the "Maximum Likelihood Estimation" method [9, 11], the "P2PRep" protocol with fuzzy aggregation [4, 5] and the method based on fuzzy logic inference proposed in [18]. All these papers report numerical results to evaluate the goodness of the proposed method. These results are usually obtained by running a simulation model. For each paper, we briefly describe the method and then we analyze the main characteristic of the simulation model. Our concern is to define some minimal requirements of a generic simulation model for the evaluation of a TRS.

1.1 EigenTrust Algorithm

In EigenTrust, each peer i rates another peer j from which it tries to download a file by keeping track of the numbers of successful *sat*(*i,j*) and unsuccessful *unsat*(*i,j*) downloads. A *local trust value* s_{ij} is then defined as the difference between *sat*(*i,j*) and *unsat*(*i,j*). To aggregate these local trust values around the P2P network, they are normalized so that malicious peers will not be able to assign arbitrarily high trust values to other malicious peers and subvert the EigenTrust algorithm. The normalized local trust value c_{ij} is defined as:

$$c_{ij} = \frac{\max(s_{ij}, 0)}{\sum_j \max(s_{ij}, 0)}$$

The normalized trust values are aggregated using the concept of *transitive trust*: peer i can know about the trust of peer k by asking all peers j with which peer i has interacted. However, since not all peers j are trustworthy, their opinions is weighed with the trust peer i places in them:

$$t_{ik} = \sum_j c_{ij} c_{jk}$$

To compute the trust value, the authors proposed the distributed EigenTrust algorithm which works as follows.

```
Foreach peer $i$ do:
  1. Query all peers $j$ who have downloaded files from $i$ for their opinions
     about him ($t_j^{(0)} = p_j$)
  2. Repeat
       2a. Compute $i$'s current global trust value $t_i^{(k+1)}$
       2b. Send opinion $c_{ij} t_i^{(k+1)}$ to $j$ from which $i$ has downloaded files
       2c. Wait for all $j$ to send their updated trust values $c_{ji} t_j^{(k+1)}$
     until $| t_i^{(k+1)} - t_i^{(k)} | < \epsilon$.
```

In order to evaluate the EigenTrust method, the authors proposed the following simulation model. They consider a typical P2P network in which a query is propagated by broadcast with hop-count horizon throughout the network as done in a Gnutella network [17].

For each malicious node, the authors proposed several *threat model* in which several aspects are taken into account, i.e., individual malicious or collectives malicious, camouflage behavior, malicious spies and so on. Note that malicious peers connect to the most highly connected peers and they are supposed to have a large bandwidth allowing them to answer to the top 20% of queries received. In order to guarantee the convergence of the method, the authors introduce the concept of pretrusted peers, i.e., peers that are known to be trustworthy.

The author also provide a content distribution model in which each peer shares a subset of content types (categories) in which the popularity of single content is governed by a Zipf distribution. An experiment consists of a certain number of repeated query cycles; each cycle terminates when the peer retrieves the required content, otherwise it continues to query the network. A simulation is composed of a number of repeated experiments depending on the scenario evaluated.

1.2 Maximum Likelihood Estimation

The "Maximum Likelihood Estimation" is a probabilistic technique whose main concern is to reduce the implementation overhead. The authors assumes to consider a P2P network composed of peers having high probabilities of performing honestly during their transactions.

Let θ_j be the probability of peer j to act honestly. A peer j interacts with peers $p_1, p_2, \ldots, p_n$ and the variables $x_1, x_2, \ldots, x_n \in \{0, 1\}$ denote the honest performance ($x_j = 1$) and the dishonest one ($x_j = 0$). Assuming that peers $p_1, p_2, \ldots, p_n$ can lie with specific probability ℓ_k for peer p_k, the probability of observing report y_k from peer p_k can be calculated as:

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

Given a random sample of independent reports $y_1, y_2, \ldots, y_n$, the likelihood function of this sample is

$$L(\theta_j) = \mathbb{P}[Y_1 = y_1]\, \mathbb{P}[Y_2 = y_2] \ldots \mathbb{P}[Y_n = y_n].$$

The maximum likelihood estimation procedure requires to find the value of θ_j maximizing $L(\theta_j)$.

In [9], the authors reports the results obtained by some simulation experiments. First of all, note that the function $L(\theta_j)$ implies the independence of reports $Y_1, \ldots, Y_n$. Thus in their simulation they assumed a noncollusive behavior of peers. They do not consider any particular structure of the network in which the interactions among peers were generated at random. Other settings are the constant number of peers (128), the number of interaction per peer varying in $\{20, 40, 60, 80, 100\}$ and the fraction of liars varying in $\{0.1, 0.2, 0.3, 0.4, 0.5\}$. All the results is the average value of 20 repeated experiments.

1.3 P2PRep and Fuzzy Aggregation

P2PRep is a reputation-based protocol which formalizes the way of each peer stores and shares with the community the reputation of other peers [6, 8, 12]. It runs in a fully anonymous and decentralized P2P environment. A more detailed description of the protocol is given in [5]. Here, we report only the reputation model.

Let $r_{i,j}$ be the local reputation resulting from direct interactions between peer i and peer j. A fuzzy value expresses local reputations to take into consideration the fact that transactions can be heterogeneous for importance, resource value, and so on. At any time $n > 1$, based on the outcome of the nth transaction ($t_{i,j}^{(n)} = 1$ if the outcome was satisfactory, $t_{i,j}^{(n)} = 0$ otherwise), the local reputation is updated as follows

$$r_{i,j}^{(n)} = \alpha^{(n)} r_{i,j}^{(n-1)} + (1 - \alpha^{(n)}) t_{i,j}^{(n)} \quad \text{and} \quad r_{i,j}^{(1)} = t_{i,j}^{(1)}.$$

The value of $\alpha^{(n)} \in [0, 1]$ is a feedback measure varying during the time following a well-known technique for feedback control, that quickly stabilizes to a fair and efficient setting [13].

The global reputation of peer j can be computed as follows. The peer i runs a poll by using P2PRep and inquires other peers for collecting their local reputation $r_{k,j}$ of j. Under the assumption of *unanimity* [3], the global reputation can be computed aggregating all $r_{k,j}$ values using the ordered weighted average (OWA) operator [20] which allows the decision maker to give different importance to the values of a criteria. Technically, an OWA operator is a weighted average that acts on an ordered list of arguments and applies a set of weights to tune their impact on the final result. Namely, in their setting, the authors get

$$\lambda_{OWA} = \frac{\sum_{k=1}^{n} w_k r_{t_k,j}}{\sum_{k=1}^{n} w_k}$$

where n is the number of reputations to be aggregated considered in decreasing order, that is, assuming $r_{t_1,j} \geq r_{t_2,j} \geq ... \geq r_{t_n,j}$ and $[w_1\ w_2 \ldots\ w_n]$ is a weighting vector.

The authors set the OWA weights *asymmetrically*, since the aggregation operator needs to be biased toward the lower end of the interval, increasing the impact of low local reputations on the overall result. The reason is that the authors assume that peers are usually trustworthiness and a malicious behavior is the exception.

Especially in [4], the authors proposed an extensive numerical evaluation of P2PRep. The underlying P2P network is such that each peer is reachable from all others and in which delays due to message routing are not take into account. Over this broadcast network, a set of queries are simulated, each asking for a randomly chosen resource. For each query, the peer querying the network is randomly chosen (with a uniform probability distribution) over all available peers. Then, a preferred offerer o is selected in two different ways: the first one, the *random policy*, selects o randomly choosing some peers among those having the resource required whilst the second one selects o using P2PRep.

The authors model the behavior of both well-behaved and malicious peers. They assume that (1) malicious peers provide only malicious resources; (2) malicious peers respond to the polling on a peer o by always providing a (malicious) 1 reputation if $o \in M$, and by providing a genuine opinion, otherwise. On the other side, all well-behaved peers i participate in a poll on offerer o by returning their local reputation $r_{i,o}$ if such a value is recorded; no response is returned otherwise.

The main settings of this simulation model are the following: the number of peers P in the network is uniformly distributed in $[300, 400]$; the number of malicious peers M, $M \subset P$ is the 40% of $|P|$; the number of different kinds of resources is 20; the max poll cardinality is uniformly distributed in $[5, 15]$. A simulation consists of 50 repeated experiments, each one evaluating a different and randomly generated scenario in which the number of queries for each experiment ranges from $1,000$ to $10,000$ with an increment of $1,000$.

1.4 Method Based on Fuzzy Logic Inference

The authors proposed a method based on fuzzy logic inferences, which can handle uncertainty, fuzziness and incomplete information in peer trust reports. Moreover, this method aggregates peer reputations with affordable message overhead. The authors start their development from an accurate analysis of eBay transaction. Thus, their method is well-suited for a centralized community, which is not the target of our simulation. On the other side, the relevance of this method depends on two facts. The first one is the definition of a set of rules to determine the weights of the fuzzy aggregator; from this point of view, this method can be viewed as a variant of P2PRep. The second one is to populate the simulation using data concerning real transaction obtained by the analysis of eBay transactions.

1.5 Remarks from the Literature

Looking at this review, we can define some minimal requirements to be considered when devising a simulation model for a TRS, i.e., a trust and reputation systems. Three major components can be identified: the network model, the content distribution model, the peer behavior model.

A good model representing the network connecting the P2P community become necessary when the dissemination of contents and local reputations is a crucial point to evaluate the performance of a TRS. To correctly model the dissemination is needed to identify the neighbors of a given peer. From our review, the most sophisticated network model is the one proposed for EigenTrust evaluation. The minimal requirement is that of the network model should allow the concept of neighborhood based on a given measure of distance between peers, e.g., hop-count distance.

The TRS evolves analyzing the positive or negative result of the transactions among peers. Each transaction is composed of a peer requiring a content to its neighborhood. It can be assumed that peers are interested in sharing and asking a subset of the total available contents in the network. From this point of view, the description given in [14] should be considered as the minimal requirement.

The peer behavior model is the crucial component of any TRS simulator. From this point of view, our review highlights several attempts for modeling the behavior of peers. The model should take into account several aspects of malicious peer behavior. Most of them (collusive or not, camouflage, malicious spies and so on) are highlighted in our review. We observe that there is no effort to model the behavior of well-behaving peers. The definition of a complete model of peer behavior is a challenging topic and it is approached in Sect. 2.

2 The Peer Behavior Model

In this section, we discuss the main assumptions modeling the behavior of peers. We assume that peers belong to two distinct classes: the class of *malicious peers* M and the class of well-behaved or *good peers* G.

2.1 Malicious Peers

A malicious peer $p \in M$ tries to distribute dangerous contents such as virus, worms, and so on. Its main objective is to distribute the maximum number of malicious contents. To reach it, the peer can follow the following phases:

distribution: p continues to distribute malicious contents until it reach a given number F_p of distributed content or its reputation is greater than a given threshold θ_p;

camouflage or disconnection: if the reputation of p is considered acceptable, the peer can hide temporarily its maliciousness acting for a certain time as a good peer in order to improve its reputation; otherwise p abandons the community.

Finally, after exiting, the peer p can connect again using the same identifier or to use a new one.

We observe that malicious peers can operate in a group. So we assume that small cliques $C_M \subset M$ of malicious peers can adopt a common strategy in order to cheat the reputation system. For instance, in a system using P2PRep protocol, they can give highest vote to peers belonging to C_M and neutral vote to the others. More formally, the clique contains both malicious peers and spies: a spy provides not malicious contents when selected as offerer but returns highest local reputations to all peers belonging to C_M.

2.2 Good Peers

A good peer $g \in G$ joins the community in order to find and to retrieve a list of required contents. For each contents, the peer g queries the community obtaining a list of offering peers O from which the offerer $o \in O$ having the best reputation is selected. Then, the peer g starts the file download from the offerer o.

After each download, the peer g checks the file to verify its integrity in terms of correctness – is it the required content? – and security – is it a malicious file? A fast and accurate check depends on the *fanatic* level associated to each peer: the probability of a peer $g \in G$ to identify malicious contents is proportional to its fanatic level. In other words, the fanatic level models the ability of a peer to recognize malicious contents after each download.

We observe that peer g becomes distributor of malicious contents every time the check fails. This situation can be interrupted when a periodic and more accurate check will be performed.

Finally, after obtaining all required contents, the good peer g temporarily exit from the community.

2.3 Peer Dynamics

The P2P community is not fixed: new peers arrive, existing peers can temporary disconnect or definitively abandon the community. Therefore, we need to model the community dynamics.

We introduce some parameters to model the rate of arrivals and the rate of abandons in such a way to allow different dynamics for peers belonging to M and G, respectively. For instance, we can have malicious peers more dynamic than the good ones, or vice versa.

3 The Simulation Model

Our model has been developed using the AnyLogic platform which allows to create models using several methodologies such as discrete event, agent-based and many others [15]. Moreover, it provides the optimization engine OptQuest [12] which can be used to optimize the model parameters.

The agent-based simulation seems well-suited to implement a decentralized community composed of autonomous individuals such as that populating a P2P network. Moreover, the *statechart* (see Fig. 1), which is the basic tool to define an agent, is the proper instruments to describe the behavior of a peer during the simulation.

The minimal requirements, described in Sect. 1.5, are implemented in our simulation model. The network model allows to define the neighborhoods of a given peer by using an hop-count distance over a grid network. The content distribution model assigns to each peer a small amount of content categories

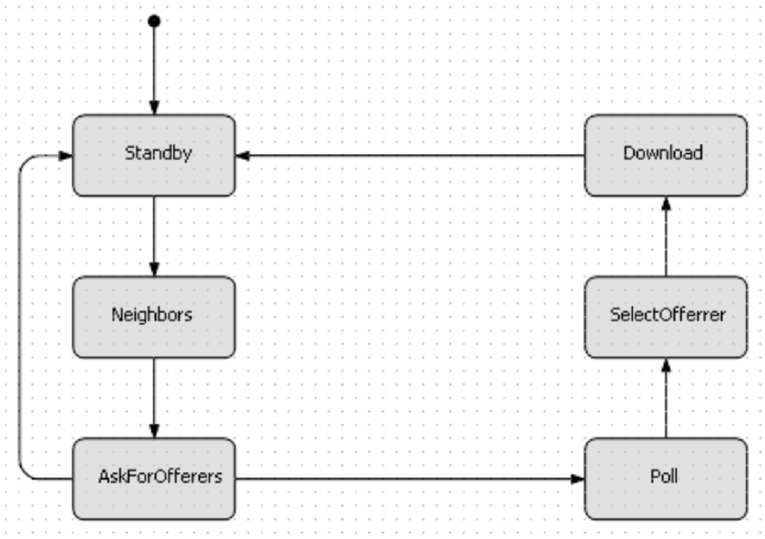

Fig. 1. The statechart modeling the good peer behavior

and the contents belonging to each category is distributed by using a Zipf function. For each content, the corresponding dimension is also randomly generated.

The main component of our agent-simulation model is the description of the agent implementing a generic peer: a peer is an autonomous agent described by a Java class which contains attributes and statecharts. The attributes model the parameters determining the peer characteristics whilst the statecharts model the peer behavior such as described in Sect. 2.

For instance, the statechart reported in Fig. 1 describes the behavior of the agent modeling a good peer: the agent leaves the `Standby` state when it decides to retrieve a content; then, it selects a list of offerers belonging to its neighborhood (`Neighbors` and `AskForOfferer` states); if this list is empty, the agent returns in the `Standby` state, otherwise it proceeds with the download; in the following two states, i.e., `Poll` and `SelectOfferer`, the agent tries to form an opinion about all the offerer peers taking into account local and global reputations; finally, the agent starts the download from the offerer having the best reputation; the download is an independent task thus the agent comes back to the `Standby` state immediately after the download starts.

The simulation experiment starts creating and arranging N_p peers over the network as depicted in Fig. 2: circles and squares represent, respectively, good and malicious peers while thin and large arrows represent, respectively, a good and a malicious content exchange among peers. The N_p peers are replicated with different attributes determining its basic characteristics. For instance, the fact of a peer is malicious or not is defined by the boolean attribute `isMalicious`.

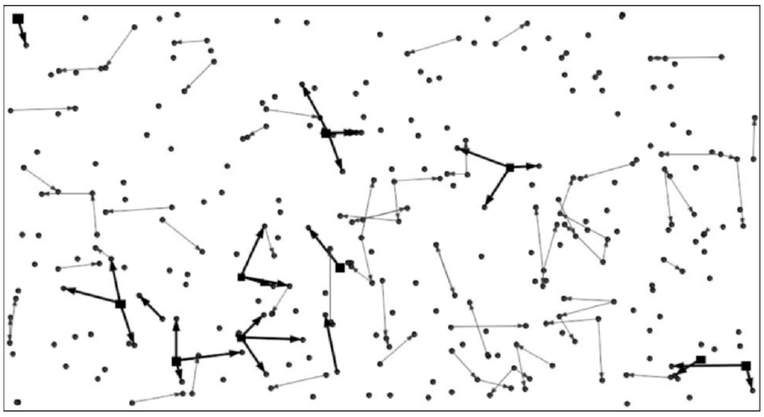

Fig. 2. The P2P network and some downloads among peers

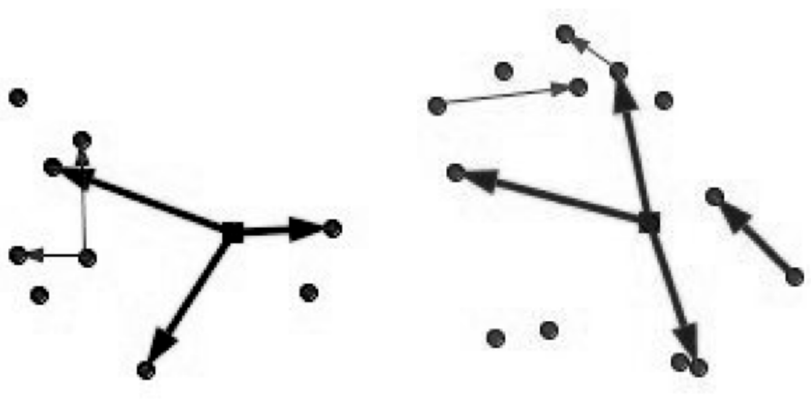

Fig. 3. Download details and distribution of malicious contents

To model the fact that each peer can have a different bandwidth, a maximum number of parallel downloads is allowed. Each download has a duration proportional to the content dimension. When a download finishes, as described in Sect. 2.2, the peer checks the content retrieved. If the content is malicious and the check fails, the peer can distribute this malicious content until the periodic and more accurate check discovers it.

Figure 3 details two possible download situation. On the left, a good peer distributes two contents to its neighbors whilst a malicious peer tries to distribute three malicious contents. If one of the receiving peer fails its check, it becomes a distributor of malicious content as depicted on the right part of the figure.

The TRS is implemented as external Java library in order to allow an independent development both to improve the existing TRSs or to add new ones. The interface between model and library is implemented in such a way to allow changes in the library without modifying anything in the model. In particular, the interface is used within `Poll` and `SelectOfferer` states (see Fig. 1). Currently, the library implements the `P2PRep` protocol using two different aggregator for computing the global reputation: the OWA operator described in Sect. 1.3 and its weighted version [19].

The main quality index is defined as the number of malicious transactions executed during the simulation by the whole community. The model also collects several statistics both a local level (statistic about a single peer) and a global level. Moreover, the model compares the TRS results with those obtained by a basic random policy in which the peer chooses randomly a peer belonging to the offerer list O.

A preliminary validation of our model has been performed comparing the simulation outcomes with those reported in [4] obtaining a positive results.

4 Conclusions

In this paper we have discussed a simulation model for the evaluation of a generic TRS within a decentralized P2P network. From the literature review, restricted to the papers in which a simulator is proposed, we have devised some minimal requirements that, in our opinion, every P2P simulators should have. Moreover, we have observed that no efforts have been made in literature to model the behavior of well-behaved peers while several considerations concerning the malicious peers behavior have been discussed. We have proposed a complete model of peers in which the behavior of both good and malicious peers is accurately defined. We have presented our agent-based simulation model having the minimal requirements discussed above and implementing the complete model of peer behavior. A Java library provides the TRS in a transparent way for the model user.

References

1. *eBay Feedback Forum.* http://pages.ebay.com/services/forum/feedback.html, 2005
2. K. Aberer and Z. Despotovic. Managing trust in a peer-2-peer information system. In *Proc. of the Tenth International Conference on Information and Knowledge Management (CIKM 2001)*, 2001
3. J. Aczél and C. Alsina. On synthesis of judgements. *Socio-Econom Planning Science*, 20(6):333–339, 1986
4. R. Aringhieri, E. Damiani, S. De Capitani di Vimercati, and P. Samarati. Assessing efficiency of trust management in peer-to-peer systems. In *Proc. of International Workshop on Collaborative P2P InfoSystems, WET ICE 2005*, June 2005
5. R. Aringhieri, E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, and P. Samarati. Fuzzy techniques for trust and reputation management in anonymous peer-to-peer systems. *Journal of the American Society of Information and Software Technology*, 57(4):528–537, 2006

6. E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, M. Pesenti, P. Samarati, and S. Zara. Fuzzy logic techniques for reputation management in anonymous peer-to-peer systems. In *Proc. of the Third International Conference in Fuzzy Logic and Technology*, Zittau, Germany, 2003 Zittau, Germany
7. E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, and P. Samarati. Managing and sharing servents' reputations in P2P systems. *IEEE Transactions on Data and Knowledge Engineering*, 15(4):840–854, 2003
8. E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, P. Samarati, and F. Violante. A reputation-based approach for choosing reliable resources in peer-to-peer networks. In *Proc. of the 9th ACM Conference on Computer and Communications Security*, Washington, DC, USA, 2002. Washington, DC, USA
9. Z. Despotovic and K. Aberer. Maximum likelihood estimation of peers' performance in P2P networks. In *Proc. of the Second Workshop on the Economics of P2P Systems*, 2004
10. Z. Despotovic and K. Aberer. Possibilities for managing trust in P2P networks. Technical Report IC/2004/84, EPFL, 2004
11. Z. Despotovic and K. Aberer. A probabilistic approach to predict peers' performance in P2P networks. In *Proc. of the Eighth International Workshop on Cooperative Information Agents (CIA 2004)*, 2004
12. OptTek Systems Inc. *OptQuest optimization software package*, 2005
13. V. Jacobson. Congestion avoidance and control. *ACM SIGCOMM Computer Communication Review*, 18(4):314–329, 1988
14. S. Kamvar, M. Schlosser, and H. Garcia-Molina. The EigenTrust algorithm for reputation management in P2P networks. In *Proc. of the Twelfth International World Wide Web Conference*, Budapest, Hungary, 2003. Budapest, Hungary
15. XJ Technologies Company Ltd. *Anylogic v5.4.1 software package*, 2005
16. P. Resnick and R. Zeckhauser. Trust among strangers in internet transactions: Empirical analysis of ebay's reputation system. In Elsevier Science, editor, *The Economics of the Internet and E-Commerce*, vol. 11 of *Advances in Applied Microeconomics*, 2002
17. M. Ripeanu and I. Foster. Mapping the Gnutella network – macroscopic properties of large-scale P2P network and Implications for System Design. *Internet Computing*, 6(1), 2002
18. S. Song, K. Hwang, R. Zhou, and Y. Kwok. Trusted P2P transactions with fuzzy reputation aggregation. *IEEE Internet Computing*, pp. 24–34, November – December 2005
19. V. Torra. The weighted OWA operator. *International Journal of Intelligent Systems*, 12(2):153–166, 1997
20. R. Yager. On ordered weighted averaging aggregation operators in multicriteria decision making. *IEEE Transactions on Systems, Man and Cybernetics*, 18(1):183–190, 1988