Centrum voor Wiskunde en Informatica

REPORT*RAPPORT*

SEN

Software Engineering

Software ENgineering

Decentralized reputation-based trust for assessing agent reliability under aggregate feedback

T.B. Klos, J.A. La Poutré

REPORT SEN-E0422 DECEMBER 2004

CWI is the National Research Institute for Mathematics and Computer Science. It is sponsored by the Netherlands Organization for Scientific Research (NWO).
CWI is a founding member of ERCIM, the European Research Consortium for Informatics and Mathematics.

CWI's research has a theme-oriented structure and is grouped into four clusters. Listed below are the names of the clusters and in parentheses their acronyms.

Probability, Networks and Algorithms (PNA)

Software Engineering (SEN)

Modelling, Analysis and Simulation (MAS)

Information Systems (INS)

P.O. Box 94079, 1090 GB Amsterdam (NL)
Kruislaan 413, 1098 SJ Amsterdam (NL)
Telephone +31 20 592 9333
Telefax +31 20 592 4199

ISSN 1386-369X

Decentralized reputation-based trust for assessing agent reliability under aggregate feedback

ABSTRACT
Reputation mechanisms allow agents to establish trust in other agents' intentions and capabilities in the absence of direct interactions. In this paper, we are concerned with establishing trust on the basis of reputation information in open, decentralized systems of interdependent autonomous agents. We present a completely decentralized reputation mechanism to increase the accuracy of agents' assessments of other agents' capabilities and allow them to develop appropriate levels of trust in each other as providers of reliable information. Computer simulations show the reputation system's ability to track an agent's actual capabilities.

1998 ACM Computing Classification System: I.2.11 Distributed Artificial Intelligence---Intelligent agents; I.2.11 Distributed Artificial Intelligence---Multiagent systems
Keywords and Phrases: Trust; Reputation; Autonomous Agents; Computer Simulation
Note: This work was carried out under the CIM project on Cybernetic Incident Management, sponsored by Senter under projectnumber TSIT2021.

Decentralized Reputation-based Trust for Assessing Agent Reliability Under Aggregate Feedback*

Tomas B. Klos[1] and Han La Poutré[1,2]

[1] Dutch National Research Institute for Mathematics and Computer Science (CWI)
P.O. Box 94079, NL-1090 GB Amsterdam, The Netherlands
[2] Faculty of Technology Management, Technical University Eindhoven
{tomas,hlp}@cwi.nl
http://homepages.cwi.nl/~{tomas,hlp}

Abstract. Reputation mechanisms allow agents to establish trust in other agents' intentions and capabilities in the absence of direct interactions. In this paper, we are concerned with establishing trust on the basis of reputation information in open, decentralized systems of interdependent autonomous agents. We present a completely decentralized reputation mechanism to increase the accuracy of agents' assessments of other agents' capabilities and allow them to develop appropriate levels of trust in each other as providers of reliable information. Computer simulations show the reputation system's ability to track an agent's actual capabilities.

1 Introduction

Reputation mechanisms allow agents to establish trust in other agents' intentions and capabilities in the absence of direct interactions. In the context of e-commerce, for example, after a transaction is concluded, the parties involved in mutual interactions are allowed to publicly rate their trading partner in terms of his compliance to the terms of trade (e.g. on eBay or Yahoo! Auctions). This benefits other, new agents considering interacting with those partners, who would otherwise have no idea about their trustworthiness. Many different reputation mechanisms have been designed and analyzed in this context, not just for interactions on auction sites [1] and on consumer-to-consumer markets more generally [2, 3], but also for supporting businesses in finding and maintaining relations with partners [4].

The use of these ideas has also been proposed and is a popular area of research in systems of interacting autonomous agents, especially as more and more

* The work described in this paper was performed in the context of the CIM project on Cybernetic Incident Management, sponsored by the Dutch government (SENTER) as project number TSIT2021. See http://www.almende.com/cim/ for more information. We are grateful to Floortje Alkemade, Pınar Yolum and Pieter Jan 't Hoen for stimulating discussions, and to two anonymous referees for helpful comments.

trade will be automated with the possibilities offered by webservices on the semantic web. Such autonomous agents need to be able to establish reliability of webservice-providers [5–7] in order to be able to select among alternatives and for composition of webservices as a value added service [8]. In a more negative phrasing, trust and reputation are used as the basis for mechanisms for ostracizing unreliable and untrustworthy agents [9, 10].

The specific context that generated our concern for this subject is Crisis Management, where many different parties are involved in non-hierarchical network topologies, in a highly dynamic environment where accurate and up-to-date knowledge is vital but scarce, perception is limited and decisions have to be made on the basis of incomplete information and with only aggregate and often time-delayed feedback [11, 12]. Other agents involved may have better access to the information required by a particular agent, and agents are differentially suited for performing certain tasks and providing particular services. The relevant questions then are: who to trust, and how to update trust on the basis of different agents' proven reputations as reliable providers of information or services? How to set up the system so that it is able to cope with dynamics and to track changing reliability? Although our system is inspired by the subject of incident management, it caters for a much wider range of applications within and between organizations.

In order to answer such questions, we focus on using reputation-based trust in a decentralized system of autonomous but interdependent, cooperative agents. The system has a non-trivial communication network structure, and, as explained more fully in Section 3.1, its distributed nature does not allow for centralized reputation storage.

Our work is most closely related to the work on the Beta reputation system [2, 13], while we deal with similar situations as discussed in [14, 9, 15–18, 5, 6]. Specifically, we extend the Beta reputation system to a decentralized setting and also to situations of trust based on combined reputation feedback from multiple agents, rather than just one (centralized) source. Finally, we introduce aggregate feedback, which makes it harder for agents to distinguish between different agents' contributions to performance, and show that the system is able to handle that, as well as dynamically changing task environments.

In the following, we will first describe the situation the agents are involved in (Sec. 2). Then, Sec. 3 will outline our system for enabling the agents to trust certain others as providers of reliable information (both on the basis of their own direct experiences with others and on the basis of reputation) and to act accordingly; in addition, our system is compared with other approaches to reputation management. The performance of our system was tested in a series of computer simulation experiments, described in Sec. 4. Conclusions and discussion follow in Sec. 5.

2 Task Environment

In this section, we set up a very general task environment in which we can study the effect of trust and reputation mechanisms in a controlled manner. This environment should be imagined to exist separately from the multi-agent system that has to operate in it and in which we implemented and experiment with our reputation-based trust mechanism. In this manner, the sensitivity of our results to changes in the parameters controlling the task environment can be assessed, and different task environments can be plugged in to investigate which changes are necessary to the trust mechanism to make the multi-agent system effective in the changed environment.

The system's ultimate task is essentially one of environmental classification: the environment has 'features,' each of which has one of several possible values, in the current paper 0 or 1, and the multi-agent system has to determine the values of those features.[3] Agents and features are spatially distributed (see Figure 1), so features are located at a certain distance from the agents—only

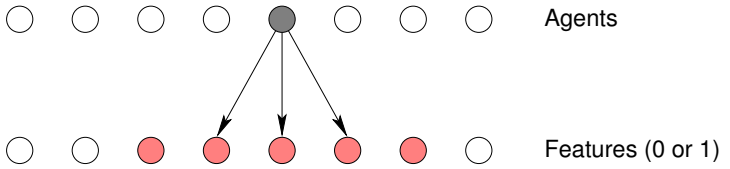

Fig. 1. Agents and features are spatially distributed.

horizontal distance is taken into account: imagine each cell on a 1-dimensional lattice to contain both an agent and a feature. It is each agent's task to determine and report the values of an individual subset of features, the agent's *task*, t, indicated by the shaded features in Fig. 1 for the shaded agent. An agent's task always consists of an odd number of features centered around the agent. In order to execute her task, the agent (1) makes observations of features and (2) communicates with other agents about the values of features.

Each agent observes the environmental features in a certain neighborhood around her, defined by her radius of vision g (in Figure 1, $g = 1$, indicating that the agent can perceive the feature in the cell she occupies herself, plus 1 cell on each side—vision is indicated by arrows), and records the values of those features. The vision of the agent need not equal the radius of her task: it may be larger or smaller, this is a parameter. In any case, the agent's observations are

[3] Typically, this classification task would precede agents' decision making based on the perceived environmental features, but here, we will take a shortcut and provide the agents with immediate feedback about the correctness of their classification. Although time-delayed feedback is also an important challenge in designing adaptive agents, we focus on aggregate feedback in this paper.

imperfect: the probability of making a mistake and observing the wrong value increases with the agent's distance to the feature as:

$$p_e = 1 - \exp(-\alpha(d+1)), \tag{1}$$

where p_e is the probability of an error, d is the horizontal distance between the agent and the observed feature (measured as the number of cells from the agent to the feature), and α is a characteristic of the cell the agent occupies (see Sec. 4.1). Notice that, since we're using $(d+1)$, there is always a positive error probability, no matter how close the agent is to a feature. Also notice that the agents are not aware of the value for α that affects their observation, so they are unable to calculate the probability that their observation is incorrect, and to discount their observation using this information.

The agent not only observes environmental features, but since different agents' vision neighborhoods may overlap, she can also communicate with other agents about the values *they* perceived for the features in her task. To this end, all the agents are part of a communication network (see Figure 2): they have connec-

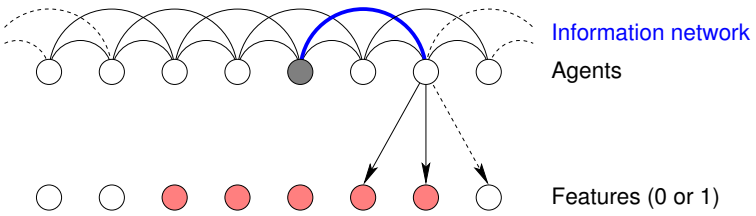

Fig. 2. Agents are connected through a communication network.

tions to certain other agents, with whom they can communicate about the values of features. In Figure 2, for example, the shaded agent, whose 5 task-features are also shaded, may benefit from communicating with the agent located 2 cells to the right, who has also observed (and may have done so more accurately) the shaded agent's 2 rightmost task-features—as well as the rightmost feature, but about that feature they will not have an incentive to communicate. When reporting feature values to others, the agents are truthful, but, as noted, they may unwittingly be communicating false feature values if they have misperceived them.

As detailed in Sec. 3.3, the agent decides on values to report for each of the features in her task, based on the values she herself observed, and on her communication with other agents. Then, each time the agent has reported the values of the features in her task, she receives—in terms of 'payoff'—the *fraction* of features in her task whose values she reported correctly. Notice that the agent is not told *which* features she reported correctly, only *how many* she reported correctly. This aggregate feedback (about the task in total, not the individual features), makes the learning problem harder. The question we are addressing in

this paper is what a reputation-based trust mechanism should look like to cope with this problem.

In this way, contributing to each other's payoff by supplying (correct or incorrect) feature values, an agent i may, for example, build up trust in another agent j as a provider of reliable information about feature 2, and in agent k as a provider of reliable information about feature 1. Furthermore, agent j may build up a more general reputation for being a provider of reliable information about feature 2 when agent i shares her opinion about and experiences with j with agents k or l.

Although the agents receive their aggregate feedback directly after reporting the values of the features in their task, note that it is *not* the agents' objective to learn the values of those features per se: they have to learn *to whom they should turn for supplying each feature value most reliably.*

3 Trust and Reputation in Multi-Agent Systems

3.1 Trust and Reputation

When deciding about interacting with another person (T for target), an agent i relies on her trust in T. Trust may be derived from i's own personal prior experiences with T, or from other people's experiences, at least to the extent that communication with those other people has given i access to those experiences. In the latter case, we speak of a reputation mechanism: reputation is one of a number of possible sources of trust, but the relevant information for i is her own private trust in T—irrespective of whether that trust is based on her own private experiences with T or on information obtained about experiences of others, and irrespective of whether or not her own trust assessment is in turn shared with others. Also, of course, a given person may have different reputations for different characteristics.

Trust As mentioned above, an agent i's trust in another agent j is—in the absence of other sources like reputation—just based on i's own past experiences in interactions with j. As j provides more evidence of being able to fulfill a certain task, i will come to expect—or *trust*—j to perform well on that task in the future also. For modeling trust, we will use the beta distribution, which captures the aforementioned idea nicely [13, 2]. The beta probability density function can be used to represent probability distributions of *binary* events, such as the probability that a particular agent will be correct in providing a value for a particular feature (based on whether she was correct or incorrect in the past). The beta distribution is a continuous probability distribution with the probability density function of p defined on the interval $[0, 1]$:

$$f(p|a, b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} p^{a-1}(1-p)^{b-1}, \tag{2}$$

where a and b are parameters and Γ is the gamma function.[4] The expected value of a beta random variable p is:

$$E(p) = \frac{a}{a+b}. \tag{3}$$

We will use this as follows. Consider a process with two possible outcomes t_j and f_j, which represent the events that a certain agent j is correct (t_j) or incorrect (f_j) in providing certain information to another agent i. Now let u_j^i and v_j^i be the number of times agent j was correct and incorrect, respectively, in reporting this information to agent i in the past. Then the density of the probability, for agent i, of observing event t_j in the future, $p^i(t_j)$, can be expressed as a function of past observations using the beta probability distribution, by setting $a = u_j^i + 1$ and $b = v_j^i + 1$ (where $u_j^i, v_j^i \geq 0$). Furthermore, the expected value of $p^i(t_j)$,

$$E[p^i(t_j)] = \frac{u_j^i + 1}{u_j^i + v_j^i + 2}, \tag{4}$$

can be interpreted as agent i's *trust* in agent j as a reliable provider of this information. As agent j reports the information correctly more and more often, agent i's trust in agent j's reliability will increase.

As an illustration, Fig. 3 plots $f(p^i(t_j))$ for different values of u_j^i in the two

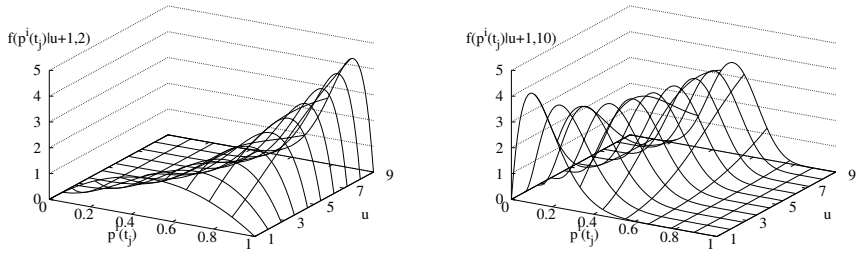

Fig. 3. Two plots for $f(p|a,b)$ with $a = u_j^i + 1$, and $b = v_j^i + 1$; v_j^i is either 1 (on the left) or 9 (on the right).

cases that the number of incorrect answers observed is $v_j^i = 1$ (on the left) and $v_j^i = 9$ (on the right). In the graph on the left, where $u_j^i = 1 = v_j^i$, the density is highest at $p = 0.5$—just like in the graph on the right where $u_j^i = 9 = v_j^i$. In those cases, there is an equal amount of evidence in favor of t_j and f_j, so $p = 0.5$ has the highest density and furthermore, both distributions are symmetric around

[4] The gamma function Γ extends the concept of factorial (defined for positive integers only) to complex and real numbers. It is related to the factorial by $\Gamma(n) = (n-1)!$.

$p = 0.5$. In the latter case, however, the total amount of evidence is much stronger, so the density is more strongly peaked at $p = 0.5$. In both graphs, the density shifts to higher probabilities of observing t_j (higher trust by agent i that agent j is correct) as the number of correct reports by agent j to agent i (u_j^i) increases.

According to [13], since the probability density $f(p)$ is vanishingly small for any given value of the continuous variable $p \in [0, 1]$, it is only meaningful to compute integrals of $f(p)$ or to use the expected value of p (Eq. 4). [13] therefore goes on to specify a reputation rating function based on this expected value, which is plotted in Fig. 4 for different combinations (u_j^i, v_j^i). For all combinations

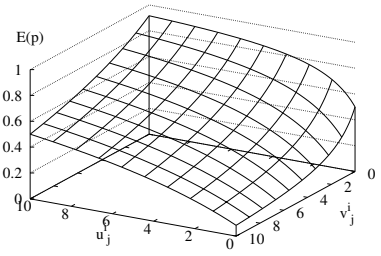

Fig. 4. The expected value of p for different combinations (u_j^i, v_j^i). Note that both the u- and the v-axes have been reversed to make the plot better visible.

where $u_j^i = v_j^i$ (so there are equal amounts of evidence for t_j and f_j), agent i's trust in agent j being correct, $E[p^i(t_j)] = 0.5$, which is liberally interpreted as a neutral disposition of agent i towards agent j's capabilities. Furthermore, $E(p)$ goes to 0 with increasing v_j^i if $u_j^i = 0$, and to 1 with increasing u_j^i when $v_j^i = 0$: if there is no evidence of agent j being correct, then agent i will come to more strongly trust agent j to be incorrect with increasing evidence of agent j being incorrect. The graph can be interpreted by thinking of agent i's trust in agent j's reliability (as a provider of correct information), as moving across the landscape in Fig. 4, as j provides more and more correct and incorrect reports. Each agent maintains such trust in every other agent, regarding those other agents' reliability as providers of information about every possible environmental feature.

Reputation In [13], the beta distribution is used in the 'Beta reputation system,' which combines trust based on the beta distribution as descibed above, with agents reporting each others' performance (u_j^i and v_j^i) centrally, so all agents have access to the same data. In addition, the system uses 'reputation discounting,' such that feedback about a target T received by an agent i from another agent j, is discounted, by i, using i's own trust in j. Discounting makes feedback from highly reputed agents more important than feedback from agents with a

low reputation. We *exclude* such discounting, because we distinguish between reputation for reliable task performance, and reputation for providing reliable information about others. More generally, agents should be allowed to develop different reputations for different capabilities, including providing information about others.

[2] describes the ability of this Beta reputation system, when used in a computer simulation of a market involving buyers and (strategically dishonest) sellers, to closely follow the sellers' honesty. The current paper concerns cooperative agents with good intentions, but not necessarily the right capabilities, so the reputation mechanism is supposed to follow agents' abilities rather than honesty. A more important distinction with [13] and [2] (than the domain of application) is that we only allow reputations to be build up locally, so just between the agents sharing information about a target T. Technically, this is preferred since the whole system becomes vulnerable to problems occuring to the central storage. Also, decentralized reputation is more realistic in many application areas, as centralized storage and access is often not feasible, requiring all participating agents to be hooked up to and to actually use the central system. Substantively (and more importantly), we want to allow agents to be able to develop different reputations in different communities, so that eventually, our system is able to self-organize into distinct communities, each consisting of well-coordinated agents.

Specifically, our implementation of reputation is as follows. If any two agents i and j share their experiences with a given agent k concerning a particular capability of k (such as reliably reporting information about a particular feature l) with each other, they add each other's counts of $u^x_{k,l}(t)$ and $v^x_{k,l}(t)$ (for the current rount t) to their own, for $x \in \{i, j\}$. After this, they share the same trust-value in k, which could be interpreted as k's reputation with (or: in the population of) i and j. In this sense, the recent surge of interest in reputation is understandable, considering the possibilities provided by the internet for sharing experiences about someone with others.

Feedback After reporting values for each of the features l in her task, an agent i receives as feedback a payoff, which is the *fraction* of features for which she reported a correct value. She then sets the current round t's positive and negative evidence with respect to reliably providing information about each feature l, $u^i_{x,l}(t)$ and $v^i_{x,l}(t)$, respectively (for all agents $x \in C_i$, the set of all agents to which agent i is connected in the organizational structure), as follows. Assume that, in timestep t, agent i ended up reporting a value of $f \in \{0, 1\}$ for feature l. Now if she received a payoff of $0 \leq \pi \leq 1$,[5] then she sets $u^i_{j,l}(t) = \pi$ and $v^i_{j,l}(t) = (1 - \pi)$ for all the agents j who had claimed the value was f. For all agents k who had claimed the value was $1 - f$, agent i sets $u^i_{k,l}(t) = (1 - \pi)$ and $v^i_{k,l}(t) = \pi$.

[5] Note that the agent is not told what the actual value of feature l was, but just that out of all features in her task, the proportion reported correctly was π.

To see why this is necessary, consider that the agents j who communicated feature value f—the value that agent i ended up reporting—contributed to the payoff π, as received by agent i. This payoff is higher (lower), *ceteris paribus*, if the value communicated by the agents j and subsequently reported by i was correct (incorrect), and this determines the extent to which agent i's trust in the agents j (as providers of reliable information about feature l) changes: if f was indeed the correct value for feature l, there should be an increase in agent i's trust in agents j, which there will be since π, now counting as positive evidence, is then higher, and if f was the incorrect value, there will be a decrease. Note that the feedback is *aggregate*, so this change in i's trust in j as a provider of information about feature l is obscured by the correctness of the other feature values reported by i. This reflects a common situation where rewards obtained from decisions are hard to attribute to the individual inputs of the decision making process, just like they may be delayed, another notorious problem in reinforcement learning [19]. In any case, it makes the agents' problem (of learning to whom they should turn for supplying each feature value most reliably, cf. Sec. 2) harder. The agents k reporting the other value for feature l (namely $1-f$) did not contribute to the payoff the agent received, so for them, the same as above with π is done with $1-\pi$: if they *were* in fact correct (and agent i was incorrect), agent i's payoff was lower, *ceteris paribus*, and since $1-\pi$ is added to $u^i_{k,l}(t)$, agent i's trust in agents k increases, as it should, since they were correct.

To summarize, if i reported f, then

- for agents j who communicated f, $u^i_{j,l}(t) = \pi$, and $v^i_{j,l}(t) = 1-\pi$, while
- for agents k who communicated $1-f$, $u^i_{k,l}(t) = 1-\pi$, and $v^i_{k,l}(t) = \pi$.

The result of this is that, if i's report was

correct (the value of feature l is indeed f), then π is high (c.p.), so (1) i's trust in agents j (who communicated correctly) increases, because positive evidence for j is equal to π (which is now high) and negative evidence for j is equal to $(1-\pi)$ (which is now low), and (2) i's trust in agents k (who communicated incorrectly) decreases, because positive evidence for k is equal to $(1-\pi)$ and negative evidence for k is equal to π;

incorrect (the value of feature l is in fact $1-f$), then π is low (c.p.), so (1) i's trust in agents j (who communicated incorrectly) decreases, because positive evidence for j is equal to π (which is now low) and negative evidence for j is equal to $(1-\pi)$, and (2) i's trust in agents k (who communicated correctly) increases, because positive evidence for k is equal to $(1-\pi)$ and negtive evidence for k is equal to π.

Forgetting Information about correct and incorrect claims by connected agents should eventually be forgotten, or at least discounted. Agents may become better at providing information and their reputation should be allowed to follow such developments. Following [13], we introduce 'forgetting' to make evidence from longer ago less important in assessing trust than more recent evidence:

older observations are discounted more strongly than recent observations, using a discounting parameter $0 \leq \lambda \leq 1$. Indexing $u^i_{j,l}$ by time t as $u^i_{j,l,t}$, at the end of any timestep t,

$$u^i_{j,l,t} = u^i_{j,l}(t) + \lambda \cdot u^i_{j,l,t-1}, \tag{5}$$

where $u^i_{j,l}(t)$ measures whether (or, the extent to which) agent j was correct about feature l in timestep t (see the discussion under "Feedback" above). The same of course holds, *mutatis mutandis*, for $v^i_{j,l,t}$. In our simulations, we follow [2], and use $\lambda = 0.99$.

3.2 Network Structure

A final part of the system that needs to be described concerns the structure of the network in which the agents are connected. An organization's structure is a very important factor in determining the system's efficiency and effectiveness [20, 21]. Many different topologies are possible, ranging from simple classes like hierarchy, ring, star, to the class of networks that contains the range from random through regular networks [22]. There has already been a wealth of studies investigating the impact of an organization's structure on its performance [20, 21] and also on the effect on the functioning of trust and reputation and referral mechanisms of different network structures.

In our case, the network influences the way in which information about environmental features and about reputation flows through the system, both of which are important inputs for the individual agents' information processing and ultimately, the system's performance [23–26]. An agent i passes information on to the agents she's connected to, both about environmental features and about their trust in other agents' reliability.

3.3 Trusting Agents

Here we discuss the way each agent decides about a value to report for each feature in her task on the basis of her own observations, information communicated by others, and the agent's trust in those other agents' reliability. After observing, each agent asks all the agents she is connected to in the system's communication structure, to report their observations of the values of the features in her task. If an agent receives information about the value of a particular feature from 2 (or more) different agents, she will need to decide upon a value to assume to be correct. Information provided by agents she trusts highly should be given more weight. For each feature l, this can be accomplished, for each possible feature value $f \in \{0, 1\}$, by determining $u^i_{X_f,l,t}$ and $v^i_{X_f,l,t}$ of the group X_f of all agents reporting the value f, as:

$$u^i_{X_f,l,t} = \sum_{j \in X_f} u^i_{j,l,t} \tag{6}$$

and

$$v^i_{X_f,l,t} = \sum_{j \in X_f} v^i_{j,l,t}. \tag{7}$$

This says that the agent simply adds the positive and negative evidence across all the agents claiming the value for feature l is f. The agent can then calculate the expected probability that this group is correct in claiming that the value of feature l is f, as:

$$E[p(f)] = \frac{u^i_{X_f,l,t} + 1}{u^i_{X_f,l,t} + v^i_{X_f,l,t} + 2}. \tag{8}$$

By doing this for both groups X_f, with $f \in \{0, 1\}$, the agent obtains the expected probabilities that each of these groups is correct, and she will decide on a value to report based on these two expected probabilities (see below).

4 Simulation Experiments

4.1 Simulation Model

We performed simulation experiments with the system described in Sec. 3, expanding upon a previous version of the paper [27]. In our simulation environment, the agents and the environmental features are distributed across a 1-dimensional grid. There are 50 consecutive cells, with the ends pasted together as a ring. Each cell contains 1 agent and 1 environmental feature, which has a randomly assigned value $\in \{0, 1\}$. The agents have a limited neighborhood that they can perceive, defined by a radius g (see Sec. 2), and a limited number of connections to other agents that they can communicate with. The accuracy of an agent's observations of a feature is influenced by her distance to that feature (d in Eq. 1), and by the value for α associated with the cell she occupies. Initially, each cell carries a randomly chosen value for α in the range $[0.05, 0.45]$. Note that this means that an agent at a distance *larger* than that of a second agent, may have a *better* perception of a feature. Each agent's task is to report the values of a number of environmental features: the feature at their own location, plus the values of the features in a number of cells to the left and to the right. Note that this means that the agent can not necessarily, by herself, perceive all the features she has to report on, so there will typically be features for the observation of which she has to rely on others.

Each simulation experiment can be described by the pseudo-code in Algorithm 1. Each simulation experiment is repeated a certain number of times (lines 1–2): in our experiments, we perform 50 runs of each experiment (using a different random seed in each experiment), and results are averaged over those 50 runs, using errorbars to indicate 1 standarddeviation around the average.

In a first series of experiments, the agents make observations and communicate their opinion about feature values just once (lines 3–6), before the start of the sequence of rounds (line 7), and after that, try to improve upon their performance solely by assessing others' performance better, which they do by iteratively adjusting their opinion (based on their trust in each other) and adjusting their trust in each other based on the feedback obtained from their opinion. These experiments are presented in Sec. 4.2. In a later series of experiments,

Algorithm 1 Pseudo-code for the simulation.

```
 1: for each in a sequence of runs do
 2:   initialize simulation     //using a run-specific random seed
 3:   for each agent do
 4:     opinion ← perceive feature values     //for now, outside the round-loop
 5:     communicate opinion to connected agents     //see Sec. 3.2
 6:   end for
 7:   for each in a sequence of rounds do
 8:     for each agent do
 9:       report values ← trust-based decision; obtain payoff     //see Sec. 3.3
10:       set this-round u- and v-evidence using payoff     //Sec. 3.1 ("Feedback")
11:     end for
12:     for each agent do
13:       share this-round evidence     //reputation, see Sec. 3.1 ("Reputation")
14:       update overall evidence     //discounting, see Sec. 3.1 ("Forgetting")
15:     end for
16:     for each agent do
17:       opinion ← reported values     //reported values replace the original opinion
18:       //communicate opinion     //new opinion is communicated, not used now
19:     end for
20:     //change some feature values     //this ensures dynamics, not used now
21:   end for
22: end for
```

renewal of observations occurs inside the round-loop, so that a changing environment can be monitored (see Sec. 4.3).

In any case, after the initial observation, each agent has an opinion about the value of each of the features in her field of vision, which she communicates to the agents she is connected to. In each of a series of rounds (line 7), then, each agent makes a trust-based decision about which values to report for each of the features in her task, reports them, and obtains a payoff, which she uses to update her (this-round) evidence in favor ($u^i_{j,l}$) and against ($v^i_{j,l}$) each of the agents she's connected to (lines 8–11).

Then, each agent shares her this-round evidence with respect to other agents, with each of the agents she's connected to in the communication network,[6] and updates her overall evidence with respect to each of her connected agents' reliability as a provider of information about each of the features in the agent's task (lines 12–15).

Finally, the agent replaces her previous opinion about the value of each feature (resulting from her trust-based decision, cf. line 9) with the value she reported earlier in the round (line 17). In addition, she may communicate her new opinion to the agents she's connected to (line 18), allowing her to change her

[6] For now, we are using just one network for exchanging both opinions about feature values and reputation-information. In future work, these will be 2 separate networks, which will allow us to study the effects of changing each of the networks' topologies separately.

reputation with those other agents, although this also introduces a lot of noise into those other agents' assessment of the agent's reliability.

4.2 Single Observation

In this first series of experiments, as explained in Sec. 4.1, the agents make just 1 observation of the feature values they can perceive, and communicate those to the agents they are connected to in the communication structure. Then they enter into a sequence of rounds, in each of which they first form an opinion about the value of each feature to report, (cf. Sec. 3.3), then report those values to obtain a payoff, and finally update their trust in others based on the payoff obtained.

Task Size = 1 (Unambiguous Feedback) Results from a benchmark experiment are shown in Figure 5, where each agent has a task consisting of just the

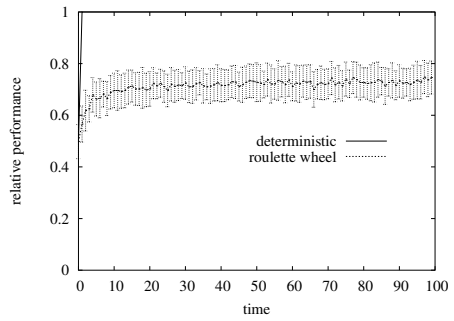

Fig. 5. Each agent has vision of 1 cell ($g = 0$) and a task comprising just the feature in her own cell. The graph shows the average agent-performance when the agents use deterministic vs. roulette wheel choice between possible feature values.

feature in her own cell, vision limited to that 1 cell, and no connections to other agents in the system (although in this setting more connections would make no difference, because different agents would not have relevant opinions to report to one another). The agents either make a deterministic choice between the alternative possible feature values—choosing the one supported by the agent or group of agents she trusts most highly overall—or they make a roulette wheel decision—choosing each feature value with a probability proportional to the relative expected probabilities mentioned in Sec. 3.3 (Equation 8).

Because the task consists of just 1 feature, feedback is unambiguous (π is either 0 or 1) and each agent is immediately able to focus on the correct feature value, which then replaces her initial observation as her opinion, in which she starts to trust herself more and more. The effectiveness of this process is compromised when the agent makes probabilistic (roulette wheel) choices, which

introduces a lot of noise, significantly slowing down the learning process: the graph is still rising at $t = 100$, but only very slowly. In some situations, however, non-determinism of the agents' choice mechanism has been shown to yield good performance when deterministic choice quickly locks the agents into developmental pathways caused by early random errors.

Task Size = 3 If the agent's tasksize is increased to 3, the impact of having different numbers of connections to other agents starts to become clear. Figure 6 shows these results when the agent has a vision of 1 cell ($g = 0$) in the graph

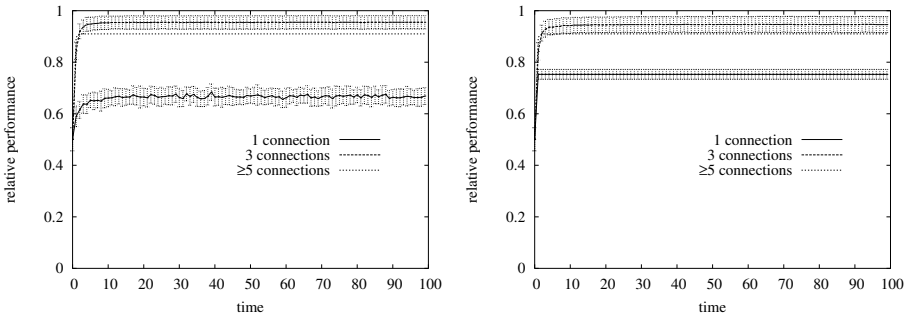

Fig. 6. Each agent has a task comprising the feature in her own cell plus the features in the cells on either side, and vision of 1 cell ($g = 0$) in the graph on the left and 3 cells ($g = 1$) in the graph on the right. The graphs shows the average agent-performance when the agents have different numbers of connections to other agents around them.

on the left and 3 cells ($g = 1$) in the graph on the right, and the agents use deterministic choice again. (Having 1 connection here means that the agent is just connected to herself.) The graph on the left shows that if the agent has a field of vision encompassing just 1 cell (her own cell), and no connections to other agents, then her performance appears to be quite bad because she has no way of reliably estimating the values of the 2 task-features outside her field of vision. In reality, the agent is doing very well given this limitation, because it would make us expect her to score about 67% (50% on the 2 features she can not observe, and 100% on the feature she can observe), which is exactly what she is doing. If she obtains connections to the agents next to her, who *can* observe the features she can not observe herself, then performance increases to approximately 95%, but then, with an increase of her field of vision (the graph on the right), comes a slight decrease in performance, caused by more opportunities for conflicts between the different agents' opinions. In both cases, adding more connections causes performance to degrade even further, while connections to more than 5 agents have no influence on the results, since those agents have no relevant contribution to make, given the current size of the agent's task.

Including Algorithm 1, line 18 If the agents inform each other of their newly formed opinion (see line 18 in Algorithm 1), then Figure 7 shows the disastrous

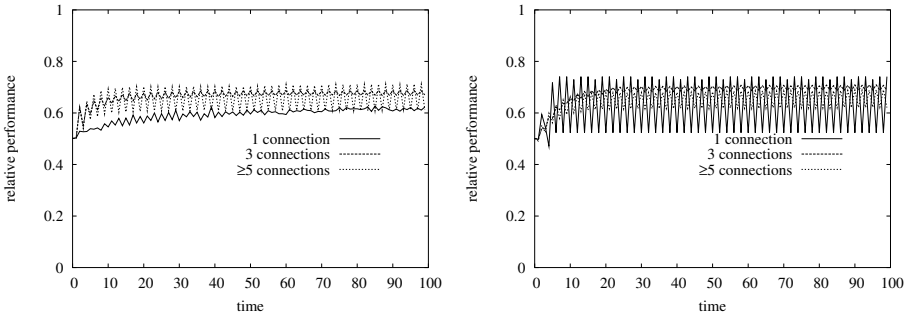

Fig. 7. Same results as in Figure 6, except the agents inform each other of their newly formed opinion.

effect on performance. Although one would imagine that the population of agents would benefit from each individual agent's better informed opinion being shared with others, the consequence is that the agents end up having to learn moving targets. Even the performance of each individual agent, with minimal field of vision (the graph on the left) and no connections to others (only 1 connection to herself), degrades, as she replaces her old observation—which earlier was staying the same all the time, allowing her to bootstrap her trust-mechanism off of it—with her newly formed opinion in each timestep. The original observation is a relatively easy target to focus on, and to learn the reliability of, but a changing observation is much harder to pin down.

Dynamically Changing Environment If the environment changes, performance can be expected to degrade more or less quickly, depending on the magnitude of the change. In Figure 8, the graphs are the same as in Figure 6, except that the value of 1 randomly chosen feature changes in each timestep, representing a modest dynamics—the important thing to focus on here, however, is the qualitative effect, rather than the quantitative one. Just like in Figure 6, the agents' performance initially increases rapidly, but it soon degrades because the ground-truth values of the environmental features are changing, while the agents are still basing themselves on their initial observation, taken before the simulation's round-loop even begins. Although one initial observation suffices when the environment is static, a dynamic environment slearly poses additional demands on the system architecture.

An interesting observation is that performance decreases more and more slowly over time, and stabilizes around 60%, except when each agent can only perceive the feature in her own cell, and has no connections to other agents. This is a clear example where having connections is beneficial, since multiple

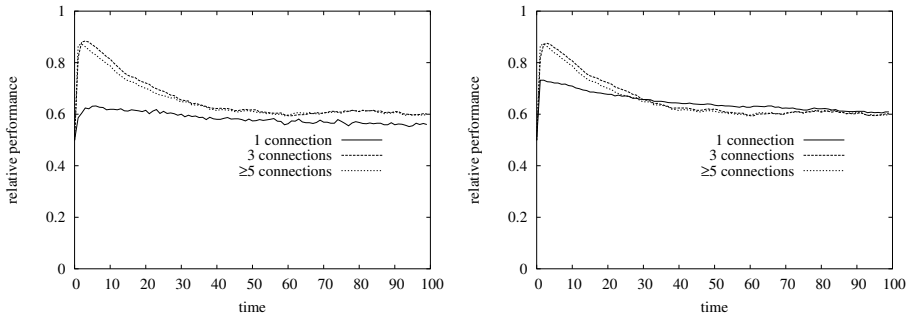

Fig. 8. Same results as in Figure 6, except the value of 1 randomly selected feature changes in each timestep.

agents may collaboratively come to a better classification of the environment. On the other hand, although performance in the case of 1 connection doesn't increase as quickly as in the other cases, it decreases even more slowly than in those other cases. An individual agent is not disturbed by the noise introduced by the reported observations of others, and can simply focus on her own task. This is especially the case in the graph on the right, where each individual agent can perceive all the features in her task, and starts to do better than the agents with more connections after $t = 30$.

4.3 Updating Observations

In Figure 9, finally, the observation activity (lines 3–6 in Algorithm 1) is put

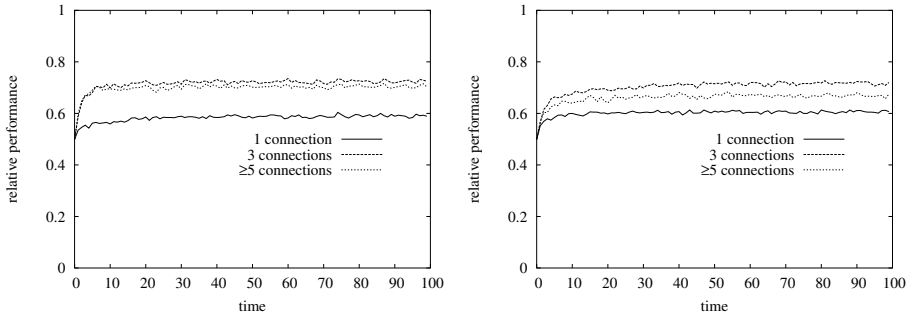

Fig. 9. Same results as in Figure 8, except observations are renewed in each timestep (cf. lines 3–6 in Algorithm 1).

inside the round-loop, so that the agents' observations are renewed in each timestep—in effect, lines 3–6 are placed after line 7 which starts the round-loop. Also, they inform each other of these new observations. (In this case, there

is no difference if the agents perform line 17 in the pseudocode, because these newly formed opinions are overridden by new observations at the start of the next timestep anyway.) In the graph on the left, the agent has more limited vision than on the right, and in that case, there is not much difference between having 3 vs. 5 or more connections: any additional information is helpful, although the agents without any connections to others are able to learn for themselves. Their performance is lower than in Figure 6 because the environment changes, but it is still in the vicinity of 67%. More generally, having new observations in each timestep, introduces new errors in each timestep as well, giving rise to the moving target problem described earlier. This problem also manifests itself in the degradation of performance when going from 3 to 5 connections in the graph on the right.

5 Conclusions and Future Work

We have designed a task environment for a multi-agent sytem to perform in, and have designed the interactions in that system to be based on trust. The agents provide each other with information necessary for task performance, and receivers of information assess providers' reliability, by feeding the payoff they receive back to the providers based on a the providers' contributions to the agent's performance. A complicating factor is that feedback is aggregate, so individual contributions to the agent's payoff are hard to separate from each other. We were still able to show promising performance of the system, as it depends on a variety of variables and design decisions in the construction of the system.

Although more systematic experimentation will be performed, some conclusions can already be drawn. It has become obvious that the agents can provide each other with too much information, introducing noise into each other's trust mechanisms, and rendering each other's learning objectives moving targets. These are harder to follow and pin down accurately, and this degrades performance in a static environment. However, providing each other with additional information is still necessary when the environment changes dynamically, but even then, performance has been shown to suffer from the agents having too many connections.

More generally, there is a trade-off between providing too much and too little information, and the parameters (and hence the 'solution') of this trade-off depend on the magnitude of the changes in the environment. Also, it was shown that the agents can have too many connections to others, with performance degrading as too many inputs about the same subject are received, making it harder for the agent to distinguish different agents' contributions from each other, and updating trust values accordingly.

On the whole, the results are promising in that we are able to increase and maintain performance in a variety of circumstances by varying the parameters of our proposed reputation-based trust mechanism. More work remains to be done, however. Further research will be focused on investigating further the influence

of the structure of the network, on applying different mechanisms for building trust and reputation, and, by way of sensitivity analysis, on the robustness of the system to variations of parameters, such as those governing agents' forgetfulness or (the dynamics of) the environment. Another point for future research is to disentangle the networks for communicating about environmental features on the one hand, and about other agents' performance on the other.

References

1. Malaga, R.A.: Web-based reputation management systems: Problems and suggested solutions. E-Commerce Research **1** (2001) 403–417
2. Jøsang, A., Hird, S., Faccer, E.: Simulating the effect of reputation systems on e-markets. In: Proc. 1^{st} Int. Conf. on Trust Management, Crete, Greece (2003)
3. Yamamoto, H., Ishida, K., Ohta, T.: Modeling reputation management systems on online C2C market. Comp. & Math. Organization Theory **10** (2004) 165–178
4. Ono, C., Nishiyama, S., Kim, K., Paulson, Jr., B.C., Cutkosky, M., Petrie, Jr., C.J.: Trust-based facilitator: Handling word-of-mouth trust for agent-based e-commerce. E-Commerce Research **3** (2003) 201–220
5. Yolum, P., Singh, M.P.: An agent-based approach for trustworthy service location. In: Proc. 1^{st} Int. Workshop on Agents and Peer-to-Peer Computing. Number 2530 in LNAI. Springer, Berlin (2002) 45–56
6. Yolum, P., Singh, M.P.: Self-organizing referral networks: A process view of trust and authority. In: Proc. ESOA, Melbourne, Australia (2003)
7. Yu, B., Singh, M.P.: An evidential model for distributed reputation management. In: Proc. AAMAS'02. (2002)
8. Singh, M.P.: Trustworthy service composition: Challenges and research questions. In: Proc. AAMAS'02. (2002)
9. Barber, K.S., Kim, J.: Soft security: Isolating unreliable agents from society. In Falcone, R., Barber, K.S., Korba, L., Singh, M.P., eds.: Trust, Reputation and Security: Theories and Practice. Volume 2631 of LNAI. Springer, Berlin (2003) 224–233
10. Schillo, M., Funk, P., Rovatsos, M.: Using trust for detecting deceitful agents in artificial societies. Applied AI **14** (2000) 825–848
11. Helbing, D., Kühnert, C.: Assessing interaction networks with applications to catastrophe dynamics and disaster management. Physica A **328** (2003) 584–606
12. Lin, Z.: The dynamics of inter-organizational ties during crises: Empirical evidence and computational analysis. Simulation Modeling Practice and Theory **10** (2002) 387–415
13. Jøsang, A., Ismail, R.: The Beta reputation system. In: Proc. 15^{th} Bled Electronic Commerce Conference, Bled, Slovenia (2002)
14. Barber, K.S., Kim, J.: Belief revision process based on trust. [28] 73–82
15. Barber, K.S., Park, J.: Finding partners to form information sharing networks in open multi-agent systems. Technical Report TR2003-UT-LIPS-020, U. of Texas, Austin (2003)
16. Barber, K.S., Park, J.: Agent belief autonomy in open multi-agent systems. In Nickles, M., Rovatsos, M., Weiss, G., eds.: AUTONOMY 2003. Volume 2969 of LNAI. Springer, Berlin (2004) 7–16
17. Damiani, E., De Capitani di Vimercati, S., Paraboschi, S., Samarati, P., Violante, F.: A reputation-based apprach for choosing reliable resources in peer-to-peer networks. In: Proc. CCS'02. (2000)

18. Wang, Y., Vassileva, J.: Trust and reputation model in peer-to-peer networks. In: Proc. P2P'03. (2003)
19. Mitchell, T.M.: Machine Learning. McGraw-Hill, Boston, MA (1997)
20. Loosemore, M.: The influence of communication structure upon crisis management efficiency. Construction Management and Economics **16** (1998) 661–671
21. Ouksel, A.M., Vyhmeister, R.: Performance of organizational design models and their impact on organization learning. Comp. & Math. Organization Theory **6** (2000) 395–410
22. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature **393** (1998) 440–442
23. Pujol, J.M., Sangüesa, R., Delgado, J.: Extracting reputation in multi-agent systems by means of social network topology. In: Proc. AAMAS'02. (2002)
24. Sabater, J., Sierra, C.: Reputation and social network analysis in multi-agent systems. In: Proc. AAMAS'02. (2002)
25. Sangüesa, R., Pujol, J.M.: Netexpert: Agent-based expertise location by means of social and knowledge networks. In Dieng-Kuntz, R., Matta, N., eds.: Knowledge Management and Organizational Memories. Kluwer Academic Publishers, Dordrecht (2002) 159–168
26. Venkatraman, M., Yu, B., Singh, M.P.: Trust and reputation management in a small-world network. In: Proc. Fourth International Conference on MultiAgent Systems. (2000) 449–450
27. Klos, T.B., Poutré, H.L.: Using reputation-based trust for assessing agent reliability. In Falcone, R., Barber, K.S., Sabater, J., Singh, M.P., eds.: Proc. of the 7^{th} Int'l. Workshop on Trust in Agent Societies at AAMAS-04, New York, NY (2004) 75–82
28. Falcone, R., Singh, M.P., Tan, Y.H., eds.: Trust in Cyber-Societies. In Falcone, R., Singh, M.P., Tan, Y.H., eds.: Trust in Cyber-Societies. Volume 2246 of LNAI., Berlin, Springer (2001)