A Reputation-based Routing Mechanism to Increase Personal Norm Compliance

Adrian Perreau de Pinninck, Stephen Cranefield, and Carles Sierra

IIIA – Artificial Intelligence Research Institute
CSIC – Spanish National Research Council
Bellaterra (Barcelona), Catalonia, Spain
adrianp,sierra@iiia.csic.es
.
Department of Information Sciende
University of Otago
Dunedin, New Zealand
scarenfield@infoscience.otago.ac.nz

Abstract. Current reputation based approaches to avoid unwanted interactions assume that agents decide by themselves whether an interaction with another should happen. Therefore, the main issue is how to assess an agent's reputation as quickly and accurately as possible. We propose a system in which requests for interaction are routed through a web of trust, and feedback on the interactions is sent by the interacting agents. Then we show that when agents use a reputation based approach to decide how to route the requests, the proportion of interactions that satisfies the agent's personal norms increases.

1 Introduction

An agent joins a multiagent system (MAS) in order to interact with other agents. From these interactions agents expect to get some satisfaction, where the definition of a satisfactory interaction varies from agent to agent depending on their personal norms. Agents have to rely on their prior experience when choosing with whom to interact, since they cannot know in advance whether the interaction will be satisfactory. A new agent in the system does not have such prior experience. Therefore, it will have to rely on the experience of others to make that decision.

We assume that all interactions in a MAS follow a basic protocol. Firstly, an agent willing to start an interaction (the *initiator* agent) will send a request to the agent with which it wants to interact (the *target* agent).[1] Secondly, the target agent would either accept or reject the request. If the target agent accepts the request an interaction would start between the two, otherwise no interaction would start. This basic protocol allows agents to interact with others as long as

[1] Although multi-partner interactions will not be discussed in this work, they could be included into this basic protocol by having requests sent to a group of agents.

they both accept and it allows agents to build up personal experience enabling them to model other agent's reputation in a fully distributed way. The main problem with such a simple protocol is that it takes a long time to build up enough experience to model agents' reputations. In the meantime agents will have to deal with many undesirable interactions.

A simple addition to the basic interaction protocol is a third stage in which information about past interactions can be gathered. This can happen either in parallel to the interaction process, or embedded in it. In some centralized approaches all interaction results are sent to the agent in charge of storing them and then queries can be made to it. In a distributed approach, gossip is sent to other agents in the MAS. This way, each agent is able to calculate the other's reputation without having extensive personal experience. Centralized approaches have a single point of failure and do not scale. Distributed approaches have to deal with false feedback, collusion by malicious agents, white-washing (changing identity to avoid detection) and other attacks.

The purpose of this work is to design a scalable multiagent system which is robust against malicious agents while increasing agent satisfaction according to their personal norms. To achieve this we propose to structure the multi-agent system through a web of trust, which guarantees authentication and privacy without the need for a centralized PKI (Public Key Infrastructure). In order to interact, agents in this system must follow an interaction protocol in which interaction requests are routed through the web of trust. After an interaction, agents may send feedback to other agents.

The proposed MAS barely decreases the agent autonomy. Thus, malicious activity is possible. We also propose a specific architecture template for agents. Agents developed following such an architecture will route requests through those agents that have signed their certificates in the web of trust (contacts). Furthermore, they will discard all requests that do not come through those agents, and they will only take into account feedback of interactions resulting from requests they routed. Finally, they will route requests based on their local knowledge of past interactions. This architecture template is called *reputation based routing* (from now on **RepRoute**). These behavior restrictions lower the impact of false feedback and white-washing. Furthermore, we claim that by using RepRoute, the rate of unsatisfactory interactions is lowered (this will be shown through experiments), even in the face of collusion. The downside of RepRoute is the small overhead from routing requests through the web of trust.

Experiments have been run to test whether the proposed RepRoute increases the rate of interactions that satisfy the agent's personal norms. These experiments tested two scenarios: i) Interacting agents select partners randomly and always accept requests. Thus, they delegate the task of avoiding unsatisfactory interactions entirely to routing agents. ii) Interacting agents use their experience from prior interactions to select partners, and interaction requests are accepted based on the target's past experience. Thus, they partially delegate the task of avoiding unwanted interactions to routing agents.

The rest of the paper is organized as follows: Section 2 defines the multiagent system and the interaction protocol. Section 3 describes the agent architecture in such a model. Section 4 defines the RepRoute mechanism. Section 5 describes the experimental scenarios and their results. Section 6 describes related work. Finally, Section 7 gives some conclusions and suggestions for future work.

2 Interaction Protocol

The network is formed by a set of agents. These agents are connected to the internet, through which they can potentially connect directly with any other agent in the network. In order to assess identity, a web of trust is being used which can be achieved through the OpenPGP standard, thus avoiding the need for a centralized PKI. In a web of trust [19], everyone creates their own certificate which is in turn signed by others to validate them, instead of having a certification authority that is assumed to be trustworthy. In this way, cryptographic privacy and authentication are provided. However, a web of trust is not enough to guarantee good behaviour from the agents.

In the proposed system there is an interaction protocol that should be followed by all agents (see Figure 1).When agent a_i wants to interact with agent a_t, it creates a request which is routed through the MAS. Upon receiving a request, an agent first verifies if it is the recipient. If the agent is not the recipient, it must decide whether to route the request and how. The decision is totally up to the agent, but there are only four possible outcomes: forward the request, reroute the request, block the request, or discard the request. Forwarding a request implies that the router agent supports the potential interaction and helps route it to its target. Rerouting the request implies that the router agent supports the potential interaction but it is not capable of helping it get routed to its target. Blocking implies that the agent does not support the interaction, but does not mind if it takes place. Finally, discarding implies that the agent does not want the interaction to take place. When the request reaches a_t, it can accept the interaction request by sending an acknowledgement to a_i and optionally to any other agents. Then a direct communication channel is established between a_i and a_t through which the interaction can happen. The system places no restrictions to interactions. Finally, if any of the agents involved in the interaction are unhappy with the results, they can send a complaint to any of the agents in the MAS.

In order to stop requests from traveling indefinitely through the network, routing information is added to them.

Definition 1. *Given a set of agents $\mathcal{A}$, a* request path *is a tuple $\langle\langle a_1, a_2, ..., a_n\rangle$, $A_r, A_b\rangle$ where: $\langle a_1, a_2, ..., a_n\rangle$ is a sequence of agents without duplicates, a_1 being the request initiator. The sequence represents an acyclic path. $A_r \subseteq \mathcal{A}$ is the set of agents that have received the request, i.e., the router agents. Finally, $A_b \subseteq A_r$ is the set of router agents which have blocked the request, i.e., the blocking agents. Let $\mathcal{P}$ be the set of all request paths. We define $p' \preceq p$ as a*

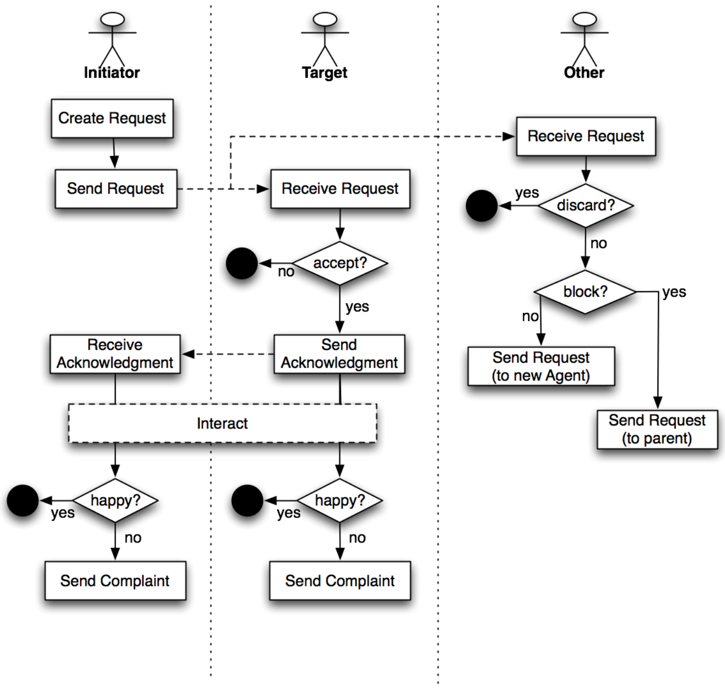

Fig. 1. Interaction protocol

relation indicating that p' is a predecessor to p, i.e., $\langle\langle a_1, a_2, ..., a_j\rangle, A_r, A_b\rangle \preceq \langle\langle a_1, a_2, ..., a_j, ..., a_n\rangle, A'_r, A'_b\rangle \leftrightarrow A_r \subseteq A'_r \wedge A_b \subseteq A'_b$.

Definition 2. *Given a set of agents $\mathcal{A}$, a set of request paths $\mathcal{P}$, and a set of identifiers $\mathcal{I}$, an* interaction request *is a tuple $\langle i, a, p\rangle$ where $i \in \mathcal{I}$ is an identifier, $a \in \mathcal{A}$ is the target agent, and $p \in \mathcal{P}$ is the route the request has travelled. Let $\mathcal{R} = \mathcal{I} \times \mathcal{A} \times \mathcal{P}$ be the set of all interaction requests formed through the request algebra defined below. A request where the target agent is part of the route is said to have* reached its target*.*

The initial request is created by the initiator agent, which will mint the request's unique identifier and select the target. The creation operation receives an identifier, an initiator, and a target agent returning a request: $\oplus : \mathcal{I} \times \mathcal{A} \times \mathcal{A} \to \mathcal{R}$, such that $\oplus(i, a_i, a_t) = \langle i, a_t, \langle\langle a_i\rangle, \{a_i\}, \emptyset\rangle\rangle$. Forwarding, rerouting, and blocking operations modify the request in the following way: The forwarding operation receives a request and an agent not in the request path and returns another request: $\otimes : \mathcal{R} \times \mathcal{A} \to \mathcal{R}$, such that $\otimes(\langle i, a_t, \langle\langle a_1, a_2, ..., a_n\rangle, A_r, A_b\rangle\rangle, a_r) = \langle i, a_t, \langle\langle a_1, a_2, ..., a_n, a_r\rangle, A_r \cup \{a_r\}, A_b\rangle\rangle$, when $a_r, a_t \notin A_r$, otherwise it is not defined. The rerouting operation receives a request and returns another: $\odot : \mathcal{R} \to \mathcal{R}$, such that $\odot(\langle i, a_t, \langle\langle a_1, a_2, ..., a_{n-1}, a_n\rangle, A_r, A_b\rangle\rangle) = \langle i, a_t, \langle\langle a_1, a_2, ..., a_{n-1}\rangle, A_r, A_b\rangle\rangle$. The blocking operation receives a request and returns another: $\ominus$:

$\mathcal{R} \to \mathcal{R}$, such that $\ominus(\langle i, a_t, \langle\langle a_1, a_2, ..., a_{n-1}, a_n\rangle, A_r, A_b\rangle\rangle) = \langle i, a_t, \langle\langle a_1, a_2, ..., a_{n-1}\rangle, A_r, A_b \cup \{a_n\}\rangle\rangle$. When discarding a request, no modification takes place since the request is not sent again. By using a multi-signature scheme we can ensure that requests cannot be falsified, *i.e.,* any agent can verify whether the identifier, route, and target have been modified.

An agent that receives a request for which it is the target decides whether it wants to accept the request and interact with the initiator agent or not. If it does not accept, it discards the message. Otherwise, it sends an acknowledgement to the initiator and optionally any other agents. The purpose of the acknowledgement is to let agents know the full route taken by the interaction request and the fact that the interaction will start.

Definition 3. *Given a set of agents $\mathcal{A}$, a set of request paths $\mathcal{P}$, a set of identifiers $\mathcal{I}$, and a request $\langle i, a, p\rangle$ that has reached its target, an* acknowledgement *is a tuple $\langle i, p\rangle$, where $i \in \mathcal{I}$ is the request identifier and $p \in \mathcal{P}$ is the request path the request took to reach the target. Let $\mathcal{K} \subset I \times P$ be the set of all acknowledgements.*

If the request target sends an acknowledgement to the initiator, a direct communication channel is established between them. Through this channel they can interact in whichever form they choose. Each interacting agent knows the interaction outcome when it finishes. The set O represents all the possible interaction outcomes. If either the initiator or the target agent is unhappy with the outcome of the interaction they may inform other agents through a complaint. Furthermore, the complaint gives the agent the request path taken by the request that started the interaction. This way, even if the request target did not send an acknowledgement to the router agents, when receiving a complaint they will have the full route information.

Definition 4. *Given a set of agents $\mathcal{A}$, a set of request paths $\mathcal{P}$, a set of identifiers $\mathcal{I}$, and a request $\langle i, a, \langle\langle a_1, a_2, ..., a\rangle, A', A''\rangle\rangle$ that reached its target, a* complaint *is a tuple $\langle i, a', \langle\langle a_1, a_2, ..., a\rangle, A', A''\rangle\rangle$ where $a' \in \mathcal{A}$ is the agent being complained about. This must be either the request initiator or target, i.e., $a' = a_1$ or $a' = a$. Let $\mathcal{C} \subset \mathcal{I} \times \mathcal{A} \times \mathcal{P}$ be the set of all complaints.*

3 Agent architecture

Agents executing the protocol in Section 2 are assumed to implement a specific architecture. This architecture consists of a local state and specific functionality. The state must contain all information received by the agent which is considered valid according to the protocol.

Definition 5. *Given the set of agents $\mathcal{A}$, the set of requests $\mathcal{R}$, the set of acknowledgements $\mathcal{K}$, and the set of complaints $\mathcal{C}$, an* agent state *associated to agent $a \in \mathcal{A}$ is a tuple $s = \langle A, R, K, C\rangle$ where $A \subset \mathcal{A}$ is the agent's set of contacts (the agents whose identity a has certified in the web of trust), $R \subset \mathcal{R}$ is a* *set of requests the agent has received or sent (whether it was the initiator, router, or target), $K \subset \mathcal{K}$ is a set of acknowledgments, and $C \subset \mathcal{C}$ a set of complaints. The following restrictions apply to agent states:*

- *The requests stored in the agent state are those the agent received or sent, i.e., $\forall \langle i, a', p \rangle \in R\ (a \in p)$.*
- *There is at most one acknowledgement per request id, i.e., $\forall \langle i, p \rangle, \langle i', p' \rangle \in R\ (\langle i, p \rangle \neq \langle i', p' \rangle \rightarrow i \neq i')$.*
- *The acknowledgments stored in the agent state are associated with a request in the local state, i.e., $\forall \langle i, p \rangle \in K\ (\exists \langle i, a', p' \rangle \in R\ (p' \preceq p \wedge a' \in p)))$.*
- *The complaints stored in the agent state are associated with a request in the agent state, i.e., $\forall \langle i, a, p \rangle \in C\ (\exists \langle i, a', p' \rangle \in R\ (p' \preceq p \wedge a' \in p)))$.*
- *Only the initiator and target agents may be complained about, i.e., $\forall \langle i, a', \langle \langle a_1, a_2, ..., a_n \rangle, A_r, A_b \rangle \rangle \in C\ (a' = a_1 \vee a' = a_n))$.*
- *There is at most one complaint per request id and interacting agent, i.e., $\forall \langle i, a, p \rangle, \langle i, a', p' \rangle \in C_j\ (\langle i, a, p \rangle \neq \langle i, a', p' \rangle \rightarrow a \neq a')$.*

Let S be the set of all agent states.

The reason why agents ignore all information about interactions whose requests they did not route is to stop false feedback, since the route taken by the interaction request can only be verified as truthful (through the multisignature scheme) by those agents which routed the request.

On the other hand, agents must implement the following functions:

- $\alpha : \mathcal{R} \times S \rightarrow 2^{\mathcal{A}}$ is the *acceptance* function. Given a request $r \in \mathcal{R}$ and a state $s \in S$, $\alpha(r, s)$ returns the set of agents (if any) to which an acknowledgement is to be sent.
- $\kappa : \mathcal{P} \times O \times S \rightarrow 2^{\mathcal{A}}$ is the *complaint* function. Given a request path $p \in \mathcal{P}$, an interaction outcome $o \in O$, and a state $s \in S$, $\kappa(p, o, s)$ returns the set of agents (if any) to which a complaint is to be sent.
- $\delta : \mathcal{R} \times S \rightarrow \{\texttt{forward}, \texttt{reroute}, \texttt{block}, \texttt{discard}\}$ is the *decision* function. Given a request $r \in \mathcal{R}$ and a state $s \in S$, $\delta(r, s)$ returns the action the agent will choose when routing r.
- $\rho : \mathcal{R} \times S \rightarrow \mathcal{A}$ is the *router selection* function. Given a request $r \in R$ to be forwarded and a state $s \in S$, $\rho(r, s)$ returns the agent to which the request will be forwarded.

4 Reputation Based Routing

In this section we introduce a class of agents implementing the proposed architecture, which we call reputation based routing agents. This technique uses the past experience in order to decide what to do with a request and, in the case that it is to be forwarded, to which agent to route it. The purpose of RepRoute is mainly to reduce the rate of unsatisfactory interactions in which the agent's personal norms are not fulfilled and also to achieve robustness against other malicious activity.

An implementation of the architecture functionality for a RepRoute type of agent must follow some restrictions. Firstly, all requests received by a RepRoute agent that where sent by an agent not in their contact list are ignored, *i.e.,* given a RepRoute agent a's state $\langle A, R, K, C\rangle$, for all requests $\langle i, a_t, \langle\langle a_1, a_2, ..., a_j, a\rangle, A_r, A_b\rangle\rangle \in R$, a_j must be an element of A. This restriction also implies that only requests received from a contact may be acknowledged, forwarded, rerouted, or blocked. The rest will be discarded. Given agent a's state $\langle A, R, K, C\rangle$, if $a_j \notin A$ then $\alpha(\langle i, a_t, \langle\langle a_1, a_2, ..., a_j, a\rangle, A_r, A_b\rangle\rangle, \langle A, R, K, C\rangle) = \emptyset$ and $\delta(\langle i, a_t, \langle\langle a_1, a_2, ..., a_j, a\rangle, A_r, A_b\rangle\rangle, \langle A, R, K, C\rangle) = \texttt{discard}$. In order to get their requests routed properly, a RepRoute agent a will route requests through its own contacts, *i.e.,* $\rho(\langle i, a_t, \langle\langle a_1, a_2, ..., a\rangle, A_r, A_b\rangle\rangle, \langle A, R, K, C\rangle) \in A$.

In all cases, RepRoute agents must have the functionality to, given a request and their agent state, calculate the probability with which it will result in a complaint. This *complaint probability* function is added to those in Section 2, and it is defined as $\pi : \mathcal{R} \times S \rightarrow [0, 1]$. Furthermore, two other functions are added which receive the complaint probability as input: The *acknowledgement probability* function, defined as $\phi : [0, 1] \rightarrow [0, 1]$, and the *blocking probability* function, defined as $\psi : [0, 1] \rightarrow [0, 1]$. These functions are executed from within the implementation of the initial four functions.

Consider a request $r = \langle i, a_t, \langle\langle a_1, a_2, ..., a_j, a\rangle, A_r, A_b\rangle\rangle$, $r \in \mathcal{R}$ sent through a path of contacts, and an agent state $s = \langle A, R, K, C\rangle$, $s \in S$. The RepRoute acceptance function will return all agents in the request path with probability $\phi(\pi(r, s))$, otherwise it returns the empty set. The RepRoute decision function will return `block` with probability $\psi(\pi(r, s))$. It will return `forward` with probability $1 - \psi(\pi(r, s))$, unless all the agent's contacts are already in the request path (*i.e.,* $A \subseteq A_r$), in which case it will return `reroute`. Finally, the RepRoute complaint function will either return an empty set or the set of all the request routers, *i.e.,* $\kappa(r, s) = \emptyset$ or $\kappa(r, s) = A_r$.

5 Experiments

In order to verify whether the presented approach increases the rate of interactions which satisfy the agent's personal norms, some experiments have been performed. The experiments consist of MAS simulations in which the society is made up of a certain number of agents that are organized as a network. Each simulation run lasts a number of rounds. In each round every agent is selected as an initiator once and a random agent is selected as its target. The initiator agent then starts the interaction protocol. To simplify the experiments, we have defined an interaction as a game in which agents can either cooperate or cheat. An interaction outcome $o \in O$ is a tuple $o = \langle x_i, x_t\rangle$ where x_i is the action taken by the initiator, and x_t the action taken by the target. Actions are either `cooperate` or `cheat`. Each agent has an attribute that defines their probability of cheating in an interaction. All agents have a personal norm indicating that they dislike to be cheated. When an agent is cheated it will complain to all agents in the request path, *e.g.,* if the current agent is the initiator,

$\kappa(\langle\langle a_1, a_2, ..., a_t\rangle, A_r, A_b\rangle, \langle x, \texttt{cheat}\rangle, s) = A_r$. Different distributions are used in the simulations for the cheating likelihood attribute (see Figure 2). The first is the uniform distribution over [0,1], the second one is an exponential distribution, where the uniform distribution is raised to a cubic power to get a higher probability of having lower cheating rates than higher.

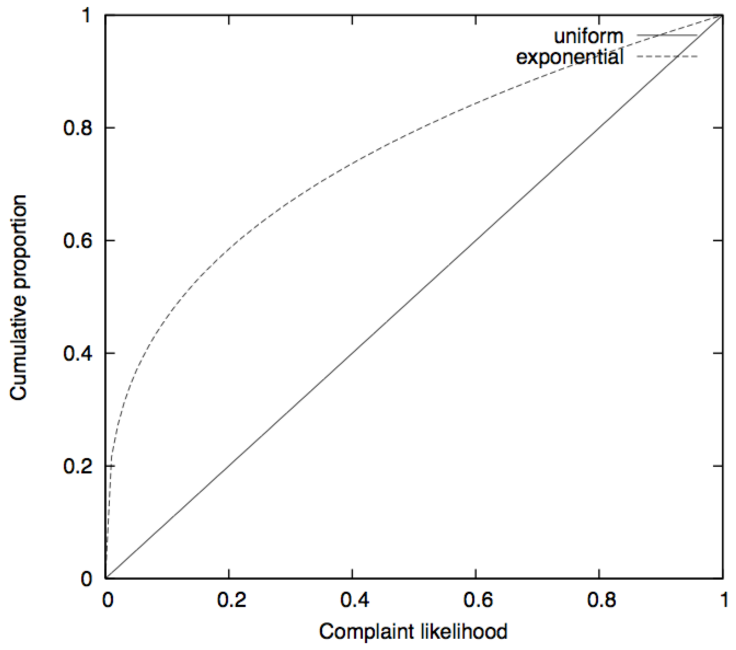

Fig. 2. Cumulative proportion of the population with likelihood of cheating

The simulated network topologies have characteristics that are found in real world contact networks, namely, the small-world property and a scale-free distribution. A graph with the small-world property has a similar diameter to a random graph with the same order and size, but a clustering coefficient orders of magnitude higher. These graphs have been generated following the Watts-Strogatz model [17] with a rewiring probability of 0.15. A scale free graph has a degree distribution following a power law. These graphs have been generated using the Albert-Babarasi model [3] of preferential attachment. Both types of networks are simulated to have either 100 or 1000 agents, and the mean number of contacts per agent is five. The two society sizes correspond to either small societies which will have more chances for interactions (simulations of 100 agents with 1000 rounds) and large societies where agents interact less frequently (simulations of 1000 agents with 50 rounds).

In those simulations where agents use RepRoute, two different functions have been tested for the blocking probability function ψ (see Figure 3): The first is

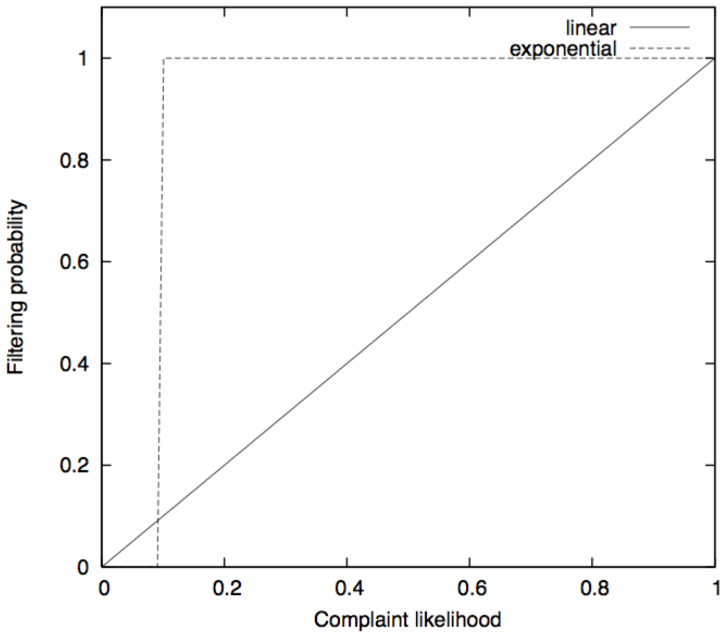

Fig. 3. Blocking probability functions

just a lineal function, the second is a threshold: routers filter all requests where the probability of complaint is higher than 0.1. RepRoute agents calculate the complaint probability π of a request $r = \langle i, a_t, \langle\langle a_1, a_2, ..., a_n\rangle, A_r, A_b\rangle\rangle$ as the probability that either the initiator or target will be complained about. The probability that an agent will be complained about is calculated as the ratio of all complaints about it over all interactions it has participated in. This is calculated using the information in the agent state. If the agent state contains no information about prior interactions of an agent, then if it is the initiator, the probability of complaint is calculated as the proportion of interactions, that started via requests $r' = \langle i', a'_t, \langle\langle a'_1, a'_2, ..., a'_n\rangle, A'_r, A'_b\rangle\rangle$ rerouted or forwarded by any of the non-blocking routers of r (*i.e.*, $\exists a_r \in A_r \setminus A_b$ $(a_r \in A'_r \setminus A'_b)$), that resulted in a complaint. Otherwise the complaint probability of the target agent is assumed to be 0 if there is no prior information. Finally, in order to lower simulation time, the router selection function is implemented through a pure shortest path algorithm. This does not affect results since all simulations where requests are routed use the same routing mechanism.

Our first conjecture was that the RepRoute approach would improve the peers' rate of satisfactory interactions. In the experiments to test this conjecture, a simulation was run where all agents use an implementation of RepRoute where the acknowledgment probability function ϕ always returns 1 (all requests are acknowledged). The baseline for comparing the results consists of a simulation with the same exact set of agents, where the initiator and target agents chosen in each round are exactly the same. The only difference being that requests are sent directly to the target by the initiator without routing them through the network. In the baseline simulations, agents accept requests that do not come from their contacts. Therefore, all possible interactions occur. Experiments in this scenario have been run for different values for the following attributes: number of agents, network topology, blocking probability function, cheating likelihood distribution, and number of rounds.

The results of the experiments are shown in Table 1. In all cases the agents had fewer unsatisfactory interactions when the RepRoute was used as opposed to no routing being used. Each row represents an experiment. The network column (Net) indicates the network structure (SF = Scale Free, SW = Small World), the size/length column (S/L) indicates the number of agents and rounds respectively, the blocking column (Blk) indicates the blocking probability function used by agents (P = linear probability, T = threshold), the cheating column (Cht) defines the cheating likelihood distribution (U = uniform, E = Exponential). The baseline (Base) and RepRoute (RepR) columns show the mean proportion of interactions with complaints. Finally, the significance column (Sign) indicates the statistical significance of the test verifying that the RepRoute algorithm has a lower proportion of complaints than the baseline. Statistical significance was tested by comparing the proportion of interactions each agent complained about in the RepRoute and the baseline scenarios. In all cases the significance was over 99%. Thus one can say that it is better to use the RepRoute technique than not doing anything at all, when no reputation technique is being used by the initiator to select the target and the target to accept request.

Net	**S/L**	**Blk**	**Cht**	**Base**	**RepR**	**Sign**
SF	100/1000	P	U	0.5158	0.4219	> 99%
SF	100/1000	P	E	0.2641	0.2071	> 99%
SF	100/1000	T	U	0.5158	0.4359	> 99%
SF	100/1000	T	E	0.2641	0.1621	> 99%
SW	100/1000	P	U	0.5158	0.4267	> 99%
SW	100/1000	P	E	0.2641	0.1937	> 99%
SW	100/1000	T	U	0.5158	0.4635	> 99%
SW	100/1000	T	E	0.2641	0.1831	> 99%
SF	1000/50	P	U	0.4880	0.3756	> 99%
SF	1000/50	P	E	0.2425	0.1747	> 99%
SF	1000/50	T	U	0.4880	0.4144	> 99%
SF	1000/50	T	E	0.2425	0.1674	> 99%
SW	1000/50	P	U	0.4880	0.4408	> 99%
SW	1000/50	P	E	0.2425	0.1644	> 99%
SW	1000/50	T	U	0.4880	0.4519	> 99%
SW	1000/50	T	E	0.2425	0.2020	> 99%

Table 1. Experiment Results for RepRoute against no routing

The second conjecture was that RepRoute is beneficial even when another reputation mechanism is used by peers in selecting their partners. In the experiments to test this conjecture, initiator agents selected a target randomly but, instead of using a uniform distribution, they selected the target agent with a probability inverse to the probability of complaining about the target, calculated using π. Therefore, agents with higher complaint rates were chosen less often. Furthermore, the acknowledgement probability function $\phi(x)$ was $1 - x$, *i.e.,* the probability of acknowledging a requests was the inverse of the probability of complaining about the initiator. The baseline for these experiments was to run the same network but changing the decision function to always forward, *i.e.,* for all request r coming from a contact and agent state s, $\delta(r, s) = \texttt{forward}$. The request order cannot be guaranteed to be the same in the baseline, since agents will have different information about each other and their choice of partner depends on this information.

Net	**S/L**	**Blk**	**Cht**	**Base**	**RepR**	**Sign**
SF	100/1000	P	U	0.3130	0.2698	N/A
SF	100/1000	P	E	0.1350	0.1190	N/A
SF	100/1000	T	U	0.3130	0.2601	N/A
SF	100/1000	T	E	0.1350	0.0817	N/A
SW	100/1000	P	U	0.3154	0.2759	N/A
SW	100/1000	P	E	0.1395	0.1159	N/A
SW	100/1000	T	U	0.3161	0.3039	N/A
SW	100/1000	T	E	0.1372	0.0894	N/A
SF	1000/50	P	U	0.4819	0.3694	$> 99\%$
SF	1000/50	P	E	0.2378	0.1702	$> 99\%$
SF	1000/50	T	U	0.4819	0.4079	$> 99\%$
SF	1000/50	T	E	0.2378	0.1595	$> 99\%$
SW	1000/50	P	U	0.4771	0.4314	$> 99\%$
SW	1000/50	P	E	0.2315	0.1596	$> 99\%$
SW	1000/50	T	U	0.4788	0.4461	$> 99\%$
SW	1000/50	T	E	0.2328	0.2004	$> 99\%$

Table 2. Experiment Results for RepRoute against shortest path routing when reputation is used to select interaction partners

The experiment results are shown in Table 2. The layout of the columns is the same as in Table 1. In all cases the average rate of non-compliant interactions was smaller for RepRoute than for the baseline. Nonetheless, the experiments with 1000 rounds showed a data distribution which was not normal, therefore t-tests for statistical significance could not be run. In the experiments with 50 rounds the data had a normal distribution and the tests showed significance over 99%. Figure 5 shows what happens in one of these experiments from an incremental point of view. The graph shows the mean complaint proportion in the simulations of the first row of Table 2. The graph has four lines: two for the baseline and two for the RepRoute approach. Each point in the graph defines the mean complaint proportion at a certain number of rounds. In the 'cumulative' lines the mean is calculated over all prior rounds, in the 'period' lines it is calculated over the last 100 rounds. It can be seen that the RepRoute simulation always has a lower mean complaint proportion. In both simulations it improves after a certain number of rounds.

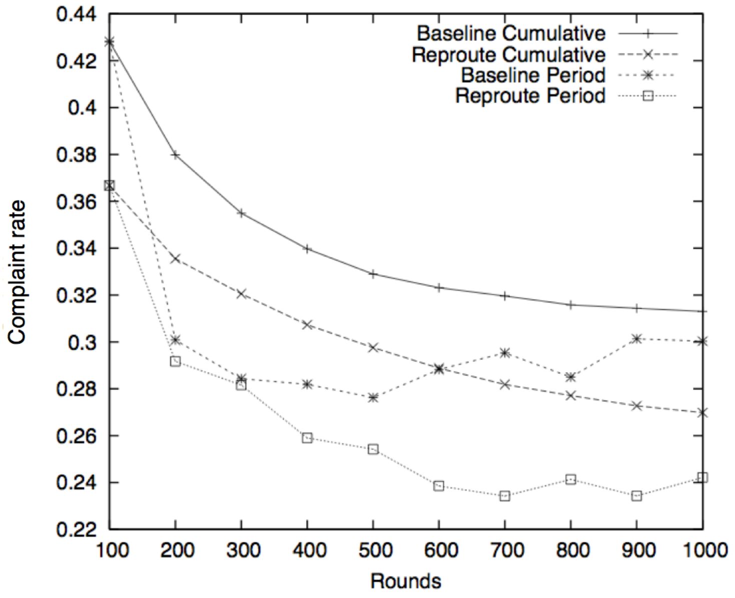

Fig. 4. Evolution of norm-violating interactions throughout an experiment using all reputation techniques

5.1 Collusion

When designing a mechanism to reduce the number of unsatisfactory interactions, threats must be taken into account. Threats may come from a single agent, or through collusion. Single agent threats include false feedback, identity change, and free-riding. Collusion threats include void interactions to artificially raise reputation and allowing identity change. The function that calculates the probability of complaint for a given agent must take these threats into account. Single agent false feedback and identity change are already handled by RepRoute, since agents changing their identity are no longer recognized by their contacts which will not route their requests. Free-riding, where agents route requests selfishly to avoid costs, is out of the scope of this work. Nonetheless, there is a lot of research in this area. Collusion in order to allow identity change is also handled in the RepRoute implementation used in our experiments by using the information of the current request routers to approximate the initiator's complaint rate when there is no prior information about it. Finally, collusion through void interactions must be dealt with. We propose another algorithm to calculate the complaint probability π that handles collusion threats. We also ran experiments to show whether the new probability function increases personal norm-compliance in a colluding scenario.

The new approach is based on calculating pairwise interaction complaint rates (*i.e.,* the probability that a complaint will be filed if two agents interact) as opposed to single agent reputation (*i.e.,* the probability that a complaint will be filed against a given agent), in order to avoid the collusion problem. The problem with this approach is that the router will always have less information about interactions between the interested agents than themselves. Therefore, it cannot make a better choice than them. In order to make up for this lack of information the router agent can estimate the probability of complaint between two agents a_i and a_j through the complaint rates in interactions with agents they have both interacted with. The mean is calculated out of all the approximations from each common interaction partner. The probability of complaint in an interaction between agents a_i and a_t when they have a common interaction partner a_c, is calculated as: $P[(cr_{i,c}\ XOR\ cr_{c,t}|int_{i,c} \wedge int_{c,t})]$, where $cr_{j,k}$ is a complaint in an interaction between agents a_j and a_k, and $int_{j,k}$ is an interaction between agents a_j and a_k. The rationales for using an XOR are that: Low complaint rates with a common partner tend to mean low complaint rates among each other. One low complaint rate and one high means that the two agents do not share the same personal norms or that the two with the low complaint rate are colluding. Therefore, the complaint rate among them would probably be high. Finally, high complaint rates with a common partner indicate similar personal norms and a low complaint rate among each other is likely. In the event that no information is available about partners in common, then the original single agent approach is used.

Our third conjecture is that the latter RepRoute mechanism to calculate the complaint probability is more robust to collusion than the former one. The experiments to test this conjecture were run with a certain percentage of the agents colluding by never complaining when interacting with each other in order to raise their reputation, but always cheating against the other agents. All non-colluding agents in the experiment had a cheating likelyhood generated randomly from an exponential distribution. Simulations were run with different percentages of the population being colluding cheaters: 10,30,50,70, and 90. For these simulations all agents selected a target randomly through a uniform distribution and they always acknowledged requests. This way the experiments focused on the impact of changes in the routing mechanism. Since colluding agents need to interact often with one another in order to raise their reputation, the society that suits them best is a small one with many interactions. Therefore, these experiments were run in societies of 100 agents and 1000 rounds. In order to reduce the number of experiments, each experiment with different percentage of colluding agents was run with different values for the blocking probability function and network topology parameters. In all cases the probability of complaint was calculated using the pairwise algorithm. The baseline was the same execution but using the single agent approach to calculate the probability of complaint. Table 3 shows the results of the experiments with colluding agents. The complaint per interaction rate is calculated by only taking into account non-colluders. In all cases the average is smaller for the pairwise calculation approach. Nonetheless in the experiment with 10% colluding agents the result was not significant due to the small amount of interactions with colluding agents. In the rest of the experiments the results were statistically significant to a high degree.

Net	**Coll%**	**Blk**	**Single**	**Pair**	**Sign**
SW	10	T	0.1896	0.1479	$\times$
SW	30	P	0.2747	0.2284	$> 99\%$
SF	50	P	0.5332	0.3635	$> 99\%$
SW	70	P	0.5802	0.3561	$> 99\%$
SF	90	T	0.8872	0.7662	$> 99\%$

Table 3. Collusion Experiment Results for RepRoute with pairwise over single agent complaint probability function

6 Related Work

How to avoid unsatisfactory interactions has been studied in computer science in the following fields: normative multiagent systems in which the goal is to reduce the number of norm violations [12], resource sharing systems [2, 10, 18] (*e.g.,* computing grids, P2P or MANET) where the goal is to avoid free riding, and electronic marketplaces whose goal is to maximize the value of interactions (*e.g.,* eBay and Amazon). In all these fields, incentives are used to promote the desired conduct of the participant. The incentive being used is mainly the quality of access to the system's resources and participants. Two approaches are used to calculate each participant's access level: the reputation-based approach uses an agent's reputation from previous interactions as an assessment of trust and the payment-based approach uses electronic currency. Both trust and currency can be accumulated by acting according to the desired conduct.

Both approaches can be implemented by centralizing the algorithms that calculate reputations (*e.g.,* eBay) or that generate the currency [16]. Centralization poses two problems: firstly, the system where the calculations are centralized becomes the bottleneck in large environments and secondly, the centralized system must be trustworthy otherwise it can take advantage of everyone else. These problems are solved by implementing a decentralized approach, which has problems of its own. Decentralized currency managements systems are based on a transferrable currency [6], thus they must deal with different types of fraud such as double spending or fake coins, whereas decentralized reputation calculation systems [1, 4, 9, 13, 18] must implement distributed information gathering systems which are robust against false feedback or collusion by a group of agents.

Our approach is a decentralized reputation-based system which uses a simple information distribution system without a gathering phase where false feedback is not an issue. Furthermore, the issue of collusion is dealt with in this paper by approximating the satisfaction probability through common partners. Other reputation mechanisms also deal with these attacks by detecting collusion through the use of Eigenvectors [9] and differences in average ratings [1, 18]. We believe that both these approaches can also be used in the model we have proposed.

This work has its origins in the enforcement of norms in normative multiagent systems, where the norms are well known to all agents. This is not the case in our approach, since each agent may have its own rules of conduct which may not be known to the rest. Since complaints for misbehavior are subjective, the system should allow agents with similar rules to interact, while inhibiting interaction among groups with opposing rules, allowing for the emergence of normative communities. Furthermore, interaction between agents in most multiagent systems is non-mediated, much in the same manner as for grid computing [2] and electronic marketplaces. In both cases there is an infrastructure used by the agents which does not make decisions on behalf of them. Therefore, all decisions to interact are done by the interacting agents themselves using either their local knowledge or knowledge from third-parties. In that respect our approach comes closer to both P2P and MANETs since we are building a network of agents which serves as a mediator for interaction. Nonetheless, in some normative multiagent systems interactions are mediated in order to control conformity to norms [5, 8]. In such cases norms cannot be broken and reputation is used as a measure of quality.

Reputation and currency techniques in resource sharing networks (*e.g.,* P2P and MANETs) are used to avoid some agents using the resources of others while not contributing themselves (*i.e.,* free-riding). In both P2P and MANETs interaction between nodes is mediated by other nodes. Nonetheless, the main difference with our approach is that P2P and MANETs are inherently dynamic networks. Neighbouring relations in both of them are short lived. Although some research exists on using social networking on P2P [10, 15], their aim is to make interaction more efficient. In current P2P implementations identity is easily changed, making sanctioning of bad behaviour difficult. The use of reputation in routing algorithms has been explored in MANETs as a means to sanction those agents that do not route messages properly in order to avoid power consumption [7, 11]. Research to avoid free riding in MANETs is centered on the network as the main resource. Our approach is different in that relations among agents are longer lasting and we also deal with the semantic level of communications. Relations are longer lasting since an agent that temporarily leaves the network will come back to the same spot when rejoining. Furthermore, in MANETs it is common practice to overhear communications of neighbouring nodes in order to gather the necessary information to calculate its reputation [7], whereas this is not possible in P2P or MAS.

7 Conclusions and future work

RepRoute is a mechanism through which agents delegate part of the decision on whether to interact with other agents by sending interaction requests through a contact network. When an agent receives an interaction request intended for another agent, it will route the request according to its local experience of past interactions. The routing agent may forward the request towards the destination agent if its local experience indicates that the interaction between the two will abide their personal norms, or it will block the request otherwise. When the request is blocked it can still be delivered via a different path. This mechanism to start interactions is coupled with a post-interaction stage in which the interacting agents are allowed to complain if they disliked the outcome of the interaction.

This approach has the following benefits: it is totally distributed, relatively easy to implement, self-policing, discourages identity change, and is robust against malicious activity. Furthermore, it lowers the rate of unsatisfactory interactions. To verify this, several experiments have been run. The data from these experiments supports our claims that RepRoute increases the rate of norm-comliant interactions in different scenarios.

A tradeoff is being made in order to improve agent satisfaction: the overhead inherent in the reputation based routing approach. The experiments have shown that there is an overhead in the number of messages being sent. The number of messages being sent when using the reputation based routing technique is roughly the number of messages sent without any routing or gossiping technique multiplied by the average path length of the contact network. Nonetheless, contact networks tend to have small world properties which makes the multiplication factor remain small. On the other hand there is an overhead in CPU and memory for calculating the probabilities of complaint. Furthermore, using a multi-signature scheme in order to avoid request falsification is also resource consuming. All of these issues raising the overhead have to be taken into account. When agent satisfaction is to be maximized, or when interactions are long and resource consuming, the overhead coming from the use of RepRoute pays off.

The scalability of the proposed system may be an issue due to the difficulty in path finding when the MAS becomes extremely large. Maintaining a routing table becomes cumbersome in large environments. In order to minimize the impact of routing tables, compact routing schemes [14] can be used. This is possible because social networks have scale-free and small-world properties. Current approaches to compact routing assume that a full view of the network is available to all. Although this can easily be done by setting up a DHT (Distributed Hash Table) where each agent's contacts are stored, it could be possible to rely on local knowledge from past routed interactions in order to improve routing efficiency. Nonetheless, such compact schemes introduce a small increase in path length. But this is not a problem, since the main concern in RepRoute is not the efficiency of routing but the improvement of the agent's personal norms satisfaction. In this case longer paths may be beneficial.

In future work we plan to test whether RepRoute can be made robust to free riding. We also want to couple our approach with more sophisticated algorithms for reputation calculation. We also plan to develop RepRoute middleware for MAS agents and P2P systems. Furthermore, we plan to test mechanisms for dynamically changing the network in which the most reputed router agents become network hubs, thus making the system more robust.

References

1. K. Aberer and Z. Despotovic. Managing trust in a peer-2-peer information system. In *CIKM '01: Proceedings of the tenth international conference on Information and knowledge management*, pages 310–317, New York, NY, USA, 2001. ACM Press.
2. B. K. Alunkal, I. Veljkovic, G. V. Laszewski, and K. Amin. Reputation-based grid resource selection. In *In Proceedings of AGridM*, 2003.
3. A. L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.
4. E. Damiani, D. C. di Vimercati, S. Paraboschi, P. Samarati, and F. Violante. A reputation-based approach for choosing reliable resources in peer-to-peer networks. In *CCS '02: Proceedings of the 9th ACM conference on Computer and communications security*, pages 207–216, New York, NY, USA, 2002. ACM.
5. M. Esteva, B. Rosell, J. A. Rodriguez-Aguilar, and J. L. Arcos. Ameli: an agent-based middleware for electronic institutions. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'04)*, 2004.
6. F. D. Garcia and J.-H. Hoepman. Off-line karma: a decentralized currency for peer-to-peer and grid applications. *Lecture Notes in Computer Science*, 3531:364–377, 2005.
7. Q. He, D. Wu, and P. Khosla. Sori: A secure and objective reputation-based incentive scheme for ad hoc networks. In *in Proc. of IEEE Wireless Communications and Networking Conference (WCNC2004*, pages 825–830, 2004.
8. J. F. Hübner, J. S. Sichman, and O. Boissier. S-MOISE+: a middleware for developing organized multi-agent systems. In *Proc. International Workshop on Organizations in Multi-Agent Systems: From Organizations to Organization-Oriented Programming (OOOP '05)*, Utrecht, The Netherlands, July 2005.
9. S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 640–651, New York, NY, USA, 2003. ACM Press.
10. S. Marti, P. Ganesan, and H. Garcia-Molina. Sprout: P2p routing with social networks. *Lecture Notes in Computer Science*, Current Trends in Database Technology - EDBT 2004 Workshops:425–435, 2004.
11. S. Marti, T. J. Giuli, K. Lai, and M. Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 255–265, New York, NY, USA, 2000. ACM.
12. A. Perreau de Pinninck, C. Sierra, and M. Schorlemmer. Friends no more: Norm enforcement in multi-agent systems. In *Proceedings of the sixth conference on Autonomous Agents and Multi-Agent Systems*, 2007.
13. J. Sabater and C. Sierra. Regret: reputation in gregarious societies. In *AGENTS '01: Proceedings of the fifth international conference on Autonomous agents*, pages 194–195, New York, NY, USA, 2001. ACM.

14. M. Thorup and U. Zwick. Compact routing schemes. In *SPAA '01: Proceedings of the thirteenth annual ACM symposium on Parallel algorithms and architectures*, pages 1–10, New York, NY, USA, 2001. ACM.
15. Y. Upadrashta. *Social Routing*. PhD thesis, University of Saskatchewan, 2005.
16. V. Vishnumurthy, S. Chandrakumar, and E. G. Sirer. Karma: A secure economic framework for peer-to-peer resource sharing. In *Workshop of the Economics of Peer-to-Peer Systems*, 2003.
17. D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393(6684):440–442, 1998.
18. L. Xiong, L. Liu, and I. C. Society. Peertrust: supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Transactions on Knowledge and Data Engineering*, 16:843–857, 2004.
19. P. Zimmermann. Pretty good privacy: public key encryption for the masses. In *Building in big brother: the cryptographic policy debate*, pages 93–107, New York, NY, USA, 1995. Springer-Verlag New York, Inc.