Decentralized Monitoring of Agent Communications with a Reputation Model

Guillaume Muller and Laurent Vercouter

École Nationale Supérieure des Mines de Saint-Étienne,
MAS Department – G2I Center, 158 cours Fauriel,
42023 Saint-Étienne CEDEX 2
{guillaume.muller, laurent.vercouter}@emse.fr

Abstract. Communication is essential in multi-agent systems, since it allows agents to share knowledge and to coordinate. However, in open multi-agent systems, autonomous and heterogeneous agents can dynamically enter or leave the system. It is then important to take into account that some agents may not respect – voluntarily or not – the rules that make the system function properly. In this paper, we propose a trust model for the reliability of agent communications. We define inconsistencies in the communications (represented as social commitments) in order to enable agents to detect lies and update their trust model of other agents. Agents can also use their trust model to decide whether to trust or not a new message sent by another agent.

1 Introduction

Peer-to-peer (P2P) systems are often presented as a promising approach to build scalable distributed applications. Peer-to-peer systems are closely related to multi-agent system because they both designate *open* and *decentralized* networks of *autonomous* entities. They are considered open since agents can dynamically enter or leave the system. The decentralization assumes that there is no central entity that can perform alone a task for the whole system. Agents need to cooperate in order to perform a collective activity that aims at achieving a common task. At last, the agents are considered autonomous since they may have been implemented by different designers or deployed by different users and their behaviour is then unpredictable.

In this context, peer-to-peer systems are vulnerable against the intrusion of malicious agents. As a matter of fact, the autonomy of each agent makes it possible for an agent to adopt a selfish or harmful behaviour. Moreover, such a bad behaviour would have dangerous effects on the whole system since it is decentralized and the malicious agent can be involved in a collective activity. Thus, it is a threat for open systems that malicious agents can dynamically enter the system.

Recent works [1, 2, 3, 4, 5, 6, 7] suggest to tackle this problem by the introduction of the notion of reputation. The reputation of an agent is computed from

R. Falcone et al. (Eds.): Trusting Agents, LNAI 3577, pp. 144–161, 2005.

its past behaviours such that an agent that had exhibited bad behaviours in the past has a low reputation value. Therefore, the very first step of this computation is the detection of a fraudulent behaviour.

The work presented in this paper focuses on the communicative behaviour of agents. The success of collective activities often depends on the good functioning of communication between agents and it may fail if some communications are, voluntarily or not, wrong. Some guarantees such as authentication, integrity, confidentiality, etc. can be obtained by the use of security techniques. However there are also some threats about the *veracity of the content* of the messages. We propose to use reputation values to evaluate the honesty of an agent in the messages it sends. This paper describes a framework in which agents can detect some agents that lie and how they use this detection to update decentralized reputation values.

The next section describes a scenario in which a peer-to-peer system is used to fetch some information distributed into the network. We use this scenario to illustrate how communications are represented. Section 3 aims at defining what is a "good behaviour" (with respect to communication) using a norm in deontic logic. Based on this norm, the concept of lie is defined and a process to detect lie occurrences is proposed. Section 4 explains how an agent combines the detection of a lie with a local trust model. This trust model distinguishes different kinds of reputation according to their reliability. Associated reputation values are used to protect an agent against undetected lies. The last two sections compare this work to related works and summarize its contributions.

2 Background

Pure peer-to-peer networks are truly decentralized networks, since **any** node can enter or quit at any time [8] . The overall system relies on the benevolence of each node to participate to the collective tasks: providing up-to-date information to new nodes, forwarding the queries and replies... However, the internal implementation of a node is unconstrained and unknown to the other nodes. Therefore when some nodes do not respect – voluntarily or not – the rules that define a good behavior in such systems, the overall functioning may be disturbed. There is a strong need for trust in such systems.

This section introduces a scenario of peer-to-peer sharing of movie theater show times, which is used in the next sections to describe some examples. The second part of this section presents a formalism to represent communication between agents.

2.1 Scenario

The scenario consists in a gnutella-like pure peer-to-peer network composed of several machines that can be be servers, desktop computers, laptops, PDAs, cell phones, ... Some of the nodes provide information about the show times of theaters. Other nodes may use this P2P network to solve requests in order to

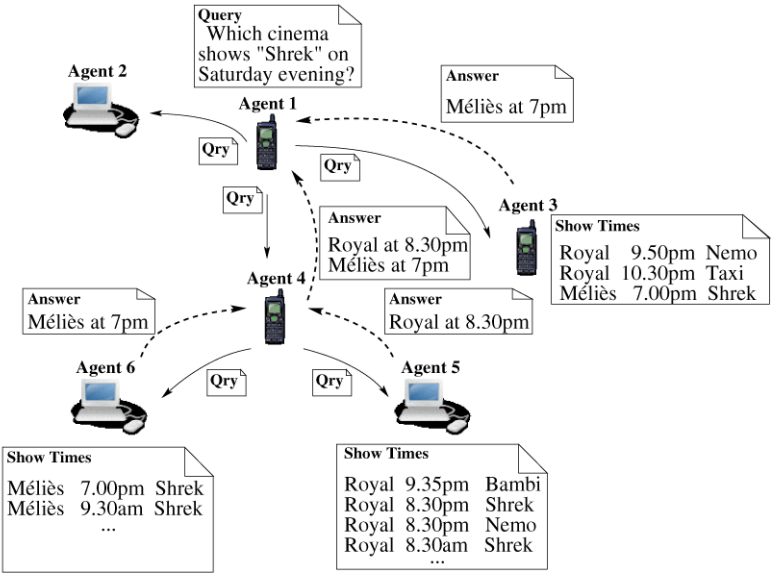

Fig. 1. P2P network to share theater show times

know when a given movie is shown and in which theater. Figure 1 illustrates this scenario.

This figure shows a network composed of six machines. Agents 1, 3 and 4 are running on mobile phones and agents 2, 5 and 6 are running on desktop computers. Agents 3, 5 and 6 have information about the show times of the theaters "Méliès" and "Royal". A user, assisted by Agent 1, wants to see the movie "Shrek" today. He asks his agent to send a query on the network to know where he can see this movie. The language used between agents and the algorithm used to solve a query are outside the scope of this paper and we do not detail them here. We consider a simple P2P discovery algorithm [9] such that each agent tries to solve locally a query and transmits it to its neighbours. Answers are transmitted along the same path as the query.

In this example, agent 4 receives answers from agent 5 and 6. Agent 1 directly receives an answer from agent 3. Agent 4 merges answers it received and forwards them to agent 1. The queries are correctly answered if we assume that every agent behaves as it should in the transmission of queries and answers. This assumption is not realistic in real P2P network that are widely open to several users. Figure 2 shows an example where the behaviour of Agent 4 disturbs the correct functioning of the system.

In this paper, this kind of wrong behaviour is called a lie. It is likely that this lie is done for the benefit of one theater at the expense of the other. Nevertheless, even if the lie is not done on purpose (it may be caused by a bug in the node implementation), the network needs mechanisms to detect the occurrence of lies to protect itself against liars.

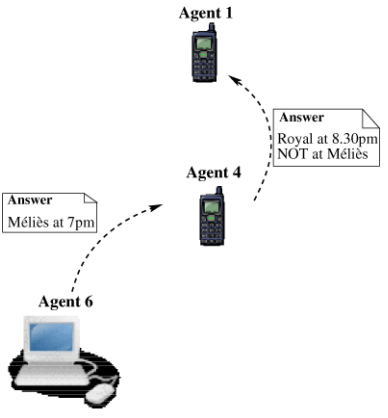

Fig. 2. Example of a lie

2.2 Agent Communication

In order to define what is a lie (and what is a correct communication behaviour), we use a formal representation of communication, which agents can reason on. This formalism is called "social semantics" since the meaning of a specific speech act is determined by the use of social commitments.

The social semantics approach of agent communication associates the utterance of speech acts to the setting up of social commitments between agents. [10, 11, 12] define operational semantics for this approach. This section introduces a slightly modified version of FORNARA et *al.* formalism.

A commitment is defined as follows:
$c(\text{uid}, \text{debt}, \text{cred}, \text{utt\_time}, \text{content}, \text{validity\_time}, \text{state})$

uid is an unique identifier for the commitment.
debt (`debtor`) refers to the agent that is committed.
cred (`creditor`) is the agent relative to which the debtor is committed.
utt_time is the time of utterance of the speech act,corresponding to the creation of the commitment.
content is a propositional formula representing what the `debtor` is committed to.
validity_time is the interval of validity of the commitment.
state is the current state of the commitment. There are five possible values for this state:
`pending`, `active`, `fulfilled`, `canceled` or `violated`.

A commitment can change under the action of time (e.g., if its interval of validity is over), or by the way of communicative acts (e.g., a creditor can refuse a commitment) or non-communicative acts (the debtor can perform an action it is committed to, in order to fulfill the commitment).

Generally, the utterance of a speech act implies the creation of a commitment which is in the `pending` state until its validity_time is reached. Actions (communicative or not) can either lead to the `fulfillment`, the `violation` or the `cancellation` of a commitment.

Hamblin [13] and Walton et *al.* [14] have introduced the concept of Commitment Stores. A Commitment Store is a set of commitments. We note $CS_x^y(t)$ the commitment store of an agent x to an agent y at time t. $CS_x^y(t)$ contains commitments taken before or at time t, where agent x is the debtor and agent y the creditor.

In the example of the figure 1, the message sent by Agent 6 to Agent 4 brings about the creation of a commitment with the values:

uid is 002509;
debt is agent 6;
cred is the agent 4;
utt_time Sat. 10-02-2004 8.00am;
content is plays(A&E, Shrek, Sat. 10-02-2004 7.00pm);
validity_time is [Sat. 10-02-2004 8.00am, Sat. 10-02-2004 7.00pm];
state is `active`.

3 Lie Detection

The scenario presented in the previous section emphasizes that a global result (such as fetching theater show times) is achieved by a collective activity of several agents. Thus, agents that do not behave as expected can prevent the success of the collective task. In this section, we introduce a norm to detect contradictory situations which might have been created by lies. Therefore, when lies are discovered, there is a co-occurrence of a violation and a deception. Since we consider that the deception, by itself, constitute a loss for the deceived party, the conditions for a fraud, according to [15], are met. We focus here on communicative actions and on fraud detection within agent communications.

Most of the works using trust in multi-agent systems [3, 5, 7] are applied in e-commerce applications where contracts are often established. Even if these contracts are implicit, fraud detection consists in the monitoring of the contract execution. In the case of communication between agents, there is no contract established. We, first, propose to define what are the accepted behaviours with respect to communication by the way of a norm. The detection process is presented in the second part of this section.

3.1 Norms About Communicative Behaviours

The good and bad communicative behaviours of agents can be defined according to states of their commitment stores. We first need to define what is inconsistency between commitments in order to define what are the authorized and prohibited states for the commitment stores.

Inconsistent Commitments. We define the incompatibility of commitments as follows:

$$(c \wedge c' \rightarrow \bot) \equiv$$
$$((c.\texttt{state} \in \{\texttt{pending}, \texttt{active}, \texttt{fulfilled}\}) \wedge$$
$$(c'.\texttt{state} \in \{\texttt{pending}, \texttt{active}, \texttt{fulfilled}\}) \wedge (c.\texttt{content} \wedge c'.\texttt{content} \rightarrow \bot))$$

Two commitments are inconsistent if they are in a "positive" state (`pending`, `active` or `fulfilled`) and if their contents are inconsistent.

A single commitment store is inconsistent if it contains two inconsistent commitments:

$$CS_x^y(t) \rightarrow \bot \equiv \exists c \in CS_x^y(t), \exists c' \in CS_x^y(t) \text{ s.a. } c \wedge c' \rightarrow \bot$$

Two commitment stores taken together are inconsistent if one of them is inconsistent or if one commitment from the first commitment store and one commitment from the second commitment store are inconsistent:

$$CS_x^y(t) \wedge CS_a^b(t) \rightarrow \bot$$
$$\equiv$$
$$(CS_x^y(t) \rightarrow \bot) \vee (CS_a^b(t) \rightarrow \bot) \vee$$
$$(\exists c \in CS_x^y(t), \exists c' \in CS_a^b(t) \text{ s.a. } c \wedge c' \rightarrow \bot)$$

A Norm to Control Communication. Tuomela et *al.* [16, 17] define several types of norms. The work presented here focus on a specific category, the r-norms (rule-norms), which are norms that must be respected by every agent of the system. A sanction is associated to the violation of such a norm in order to penalize the agents that do not respect it. However, in our case, the intrinsic decentralization of the system implies that there is no central institution that applies sanctions to the violators of the r-norm. Here, we consider that the sanction is executed by the other agents through a local increase or decrease of the reputation value of the violator.

The norm that defines the limits of an accepted communicative behaviour is written using deontic logic [18]. The modal operator O is used to represent an obligation such that $O(\alpha)$ express that α is an obligatory state. In the scenario considered in this paper, communication between agents should respect the following norm (Ω is the set of the agents in the system):

$$O(\neg(\bigcup_{x \in \Omega} CS_x^a(t) \cup \bigcup_{y \in \Omega} CS_a^y(t) \rightarrow \bot))$$

This formula sets that commitments taken towards and by an agent must be consistent. In order to detect the violation of this norm, agents should be able to detect one of the three situations below:

$$\bigcup_{x \in \Omega} CS_x^a(t) \rightarrow \perp \tag{3a}$$

$$\bigcup_{y \in \Omega} CS_a^y(t) \rightarrow \perp \tag{3b}$$

$$\bigcup_{x \in \Omega} CS_x^a(t) \not\rightarrow \perp \text{ and } \bigcup_{y \in \Omega} CS_a^y(t) \not\rightarrow \perp \text{ and } (\bigcup_{x \in \Omega} CS_x^a(t) \cup \bigcup_{y \in \Omega} CS_a^y(t) \rightarrow \perp) \tag{3c}$$

Situations Prohibited by the Norm. Figures 3 to 5 describe the three situations that are prohibited by the norm.

The situation of contradiction in sending refers to formula 3b. In the example shown by Fig. 3, agent 4 is the debtor of inconsistent commitments and is in a situation contradiction in sending. When this situation is observed, we consider that agent 4 has lied. We assume that the communication middleware has the non-repudiation [19] property to prevent an agent from claiming that it did not send an observed message.

The norm does not prevent an agent from changing its beliefs. It only constrains an agent to cancel its previous commitments, that are still active, about

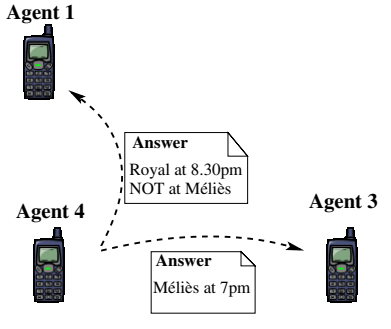

Fig. 3. Contradiction in sending

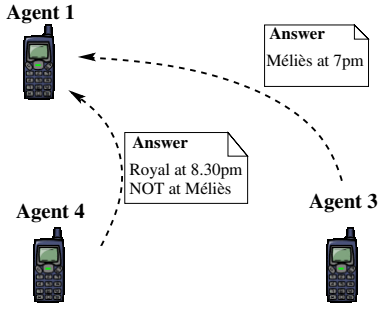

Fig. 4. Contradiction in receiving

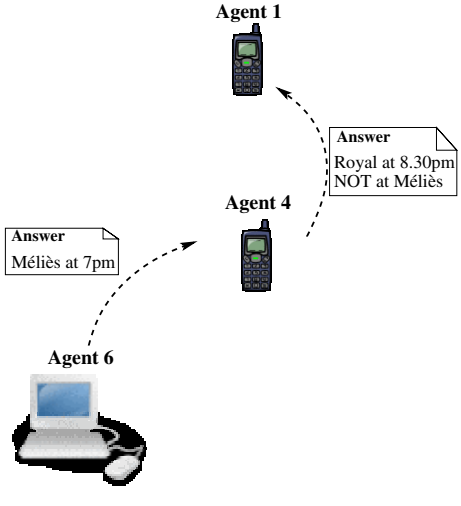

Fig. 5. Contradiction in transmitting

a given content α, if the agent wants to create a commitment about a content β inconsistent with α. Then, the only way for agent 4 to give evidence that it did not lie in the example of Fig. 3 is to provide a message proving that agent 4 had canceled one of the two inconsistent commitments before creating the other.

Figure 5 presents the situation of contradiction in transmitting (formula 3c). This contradiction only appears if agent 4 sent its message to agent 1 after that it received the message from agent 6. If agent 4 wants to send its message to agent 1, it has to cancel explicitly the commitments for which it is creditor and that are inconsistent with the message to send.

In the situation of contradiction in receiving (Fig. 4, formula 3a), agents 3 and 4 are committed towards agent 1 with inconsistent contents. The union of the commitments stores taken towards agent 1 is, therefore, inconsistent. This situation is also a violation of the norm but in this case the agent that is considered as a liar did not send any message. In this case, the norm forces an agent to be creditor of a set of consistent commitments. Therefore, the agent must refuse at least one of the commitment if it is inconsistent with other commitments (the agent is still free to choose which commitment it refuses).

However, a violation of the norm is not always a lie. The agent that detects the violation of the norm may have a local view of some commitment stores that needs to be updated. For instance, the agent may have missed a message that canceled one of the commitments involved in the inconsistency. The observation of one of the situation of violation of the norm starts a process that leads to such an update or to the evidence that a lie was performed.

3.2 Lie Detection Process

The simplest way to start a lie detection process is that an agent x observes some messages that violate the norm. In the situations of contradiction, an agent y is suspected of lying and the agent that observed the contradiction executes the following steps in order to confirm that a lie occurred:

1. agent x sends a message to agent y containing copies of the contradictory messages to state that x suspects y of lying;
2. if it can do so, the agent y sends to agent x a message that cancels the contradiction;
3. if the contradiction still holds, agent x considers agent y as a liar and can update its trust model of y with this information (see Sec. 4.2).

It is also possible that an agent x refuses a commitment for which it is creditor without knowing if it is a lie or not. In the example of Fig. 4, the agent 1 is in a situation of contradiction in receiving and has to cancel at least one of the two commitments in order to respect the norm. But it is possible that none of the two commitments is a lie. Agents 4 and 3 may have different beliefs. A process that tries to determine which message to reject starts: the messages involved in the inconsistency are sent to a set of agents $\{z_i\}$. This set should, at least, contain the senders of the inconsistent messages. The choice to add other agents

in this set is free but is likely to contain agents for which agent x have an high reputation value. Each agent z_i acts in one of the following ways:

1. it argues for or against the content of a commitment;
2. it gives information about another message that is at the origin of the message that created one of the inconsistent commitments;
3. It transmits the request to other agents;
4. It ignores the request and does not reply.

If agent z_i argues for or against the content of the commitment, an *argumentation* process between agents z_i and x begins. The goal of an argumentation process is to reach a consensus about a given fact. Nevertheless, it is not a problem of lying but rather one of divergent opinions. Some argumentation processes can be found in [12, 20].

The second possibility is that z_i provides information about a message that supports the message that created one of the inconsistent commitments. For instance, an agent y may have created a commitment about a content α because another agent w is committed towards y for the same content. Then, the real *source* of the commitment created by y is the agent w. Agent y will, therefore, involve agent w in the detection process to allow agents of the set $\{z_i\}$ to consider the message it sent. This also permits agent y to defer its responsibility on agent w, in case there is a lie.

The last two possibilities of actions for z_i are not important because they do not have any influence on the detection process.

A lie detection process is composed of several interlaced actions like the ones described above. For instance (see Fig. 6), an agent x is in contradiction in receiving with respect to messages from agents v and y. It begins an argumentation process with agent y (arrow labelled 1). During this argumentation, agent y informs agent x that its commitment is supported by a message sent by an agent w (arrow 2). Agent x asks agent w to justify. Agent w argues (arrow 3) and, finally, sends to agent x another message sent by agent v supporting its

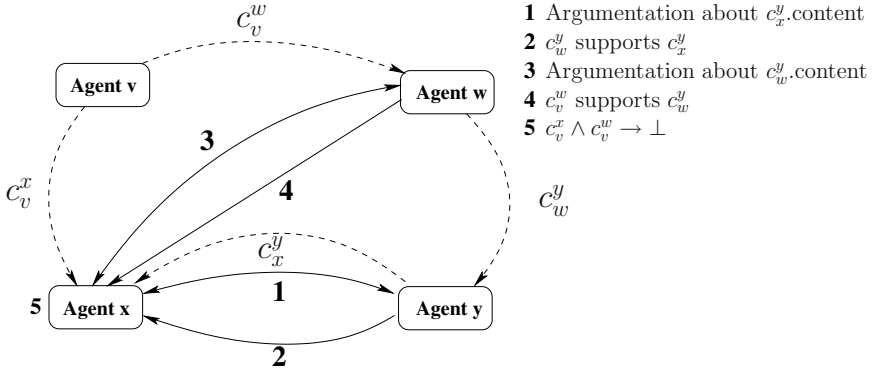

Fig. 6. Example of interlaced processes of argumentation and source detection

own commitment (arrow 4). At last, agent x discovers that v is a liar because the message described by w gives evidence of a situation of contradiction in sending.

4 Trust Modeling

The process of detecting lies benefits agents since it helps them not to believe some incorrect messages. It is also interesting for agents to build and to maintain a trust model about other agents. In fact, there are lies that will remain undetected either because the agent has never contradicted its lie or because the local observations of each agent are not sufficient to detect an inconsistency. Then, a trust model based on lie detection can be used to distrust the messages sent by an agent that has already lied.

Castelfranchi et *al.* [4] define trust in the following way: an agent **X** can only trust an agent **Y** for $\mathbf{g}/\alpha$, where **g** is a goal and α is the action **X** wants **Y** to do. We focus here on communication. Actions agents perform are communicative acts, i.e. sending messages. Moreover, there is no precise goal to reach but rather a persistent goal. This persistent goal is to prevent a norm, defining the limits of a correct communicative behaviour, from being violated. According to the definition of Castelfranchi et *al.*, we consider in the context of this article that an agent **X** trusts or distrusts an agent **Y** for respecting the communication norm while sending a specific message.

This section presents the trust model that is used: how trust is represented in agents, how it is initialized and how it is used.

4.1 Various Kinds of Reputation

An agent has a trust model about another agent by the way of reputation values. There exists different kinds of trust [21]. For instance, there are trusts related to the perceived environment, trust related to the presence of institution, etc. There are also trusts based on interpersonal relationships: an agent can compute a value based on its direct experiences, or based on external information. Conte et *al.* [2] distinguish different roles that agents can fulfill in a trust framework. In the case of a lie detection process, we identified a few roles:

A target is an agent that is judged.
A beneficiary is an agent that has the reputation value.
An observer is an agent that observes *raw data* used in an evaluation of the reputation of the target. In the example of this paper, these raw data are messages exchanged by agents.
An evaluator is an agent that transforms raw data (i.e. messages) into *interpreted data* such as a reputation value.
A gossiper is an agent that transmits either observations or evaluations about the target.

Depending on the agents that play these roles, a reputation value is more or less reliable. For instance, an agent may consider a reputation computed by

another agent less reliable than the reputation that it has itself computed from messages directly observed. It is then important to identify different kind of reputations that can have different values. From the notions of observation and detection introduced in the previous section, we define four kind of reputations:

Direct Experience based Reputation (DEbRp) is based on direct experiences between the beneficiary and the target. A direct experience is a message that has been sent by the target to the beneficiary and that has either been detected as a lie or as a sincere message.

Observation based Reputation (ObRp) is computed based on observations (raw data) gathered from gossipers. The beneficiary uses these observations to detect lies and to compute a reputation value.

Evaluation based Reputation (EbRp) is computed based on evaluations (interpreted data) gathered from gossipers. These evaluations are transmitted as estimated reputation values.

General Disposition to Trust (GDtT) is not attached to a specific target. This value is not interpersonal and it represents the inclination of the beneficiary to trust another agent if it does not have any information about its honesty.

There exists several ways to compute reputation values based on aggregation of several sources. For instance, [3, 5, 7, 1] proposes some functions to merge rating or reputation values. In this paper, we do not make any assumption on the method used to aggregate values.

4.2 Example

These various kinds of reputation are updated in situations such as lie detection (cf. previous section). Figure 7 shows an example of a situation of contradiction in transmitting that is detected and used to update some reputation values.

It is assumed that agent 2 has a way to perceive the messages exchanged between agent 4 and the agents 1 and 6. These messages may be obtained while arguing with other agents (see Sec. 3.2 for more details) or by another way. For instance, it is possible to use the work described in this paper to deploy specific agents that will be in charge of lie detection and that are able to spy communications.

Agent 2 is then the *observer* of two messages. Agent 2 uses these messages to detect a contradiction in transmitting and asks agent 4 if it is able to cancel this contradiction. If agent 4 can not provide any message to defend itself, agent 2 updates its Direct Experience-based Reputation about agent 4.

In the example of the figure, agent 2 forwards to agent 1 the message from agent 6 to agent 4. Based on this observation, agent 1 is also able to detect the lie. Agent 1 will, therefore, update its Observation-based Reputation about agent 4.

Figure 8 shows a similar example with the difference that agent 2 does not forward a raw observation, but either an interpreted one, for instance: “I lowered

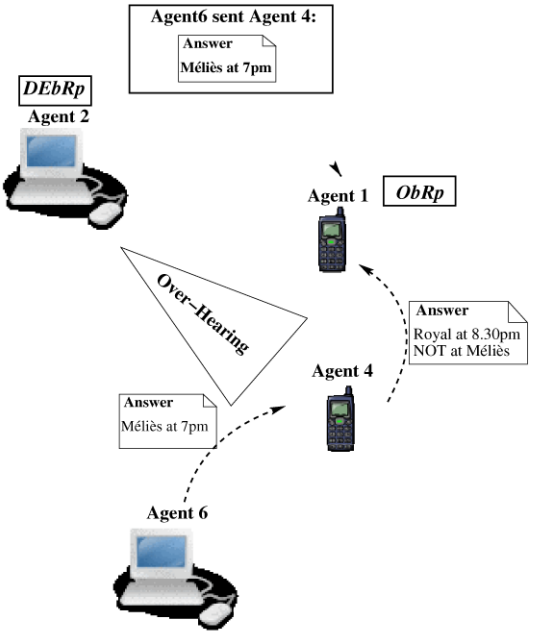

Fig. 7. Update of the Direct Experience-based Reputation and the Observation-based Reputation

agent 4's reputation for respecting the communication norm". Agent 1 then updates its Evaluation-based Reputation about agent 4's. Table 1 summarizes the roles the agents play and the type of reputation updated in these situations.

4.3 Representation and Initialization of Reputation Values

Reputation is a social concept that links an agent with its acquaintances. It is also a leveled relation [4] such that an agent a may trust another agent b more than another agent c. But the agent can also be unable to distinguish in a set of agent which one it trusts more. Therefore, reputation implies only a partial ordering of acquaintances from the point of view of an agent.

The computer representation of reputation must preserve these properties. We use real values to represent reputation in order to have leveled values that can be compared. This value belongs to the set $[-1, +1]$ where -1 represents a strong distrust and $+1$ stands for a strong trust. 0 represents a neutral opinion, which means the agent has gathered information about the other, but it can not form neither a positive nor a negative opinion.

However, this set of values is not sufficient. In an open system, agents dynamically enters and leaves the system. Then, there must be an initial reputation value attached to a new agent. If this value belongs to the set $[-1, +1]$, a new agent will be compared to other agents with a value that does not correspond to its previous behaviours. If this initial value is high, it is dangerous for the system

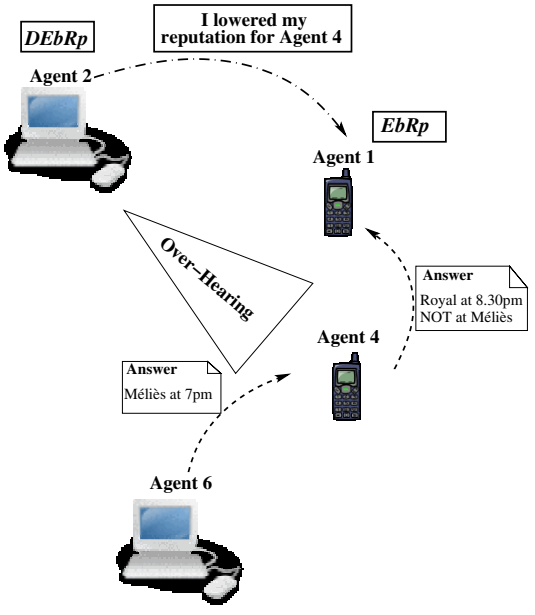

Fig. 8. Update of the Evaluation-based Reputation

Table 1. Links between roles and type of reputation

Roles					agent who updates its reputation	type of reputation updated
Observer	Evaluator	Gossiper	Beneficiary	Target		
Agent 2	Agent 2	$\emptyset$	Agent 2	Agent 4	Agent 2	DEbRp
Agent 2	Agent 1	Agent 2	Agent 1	Agent 4	Agent 1	ObRp
Agent 2	Agent 2	Agent 2	Agent 1	Agent 4	Agent 1	EbRp

that will be vulnerable against the intrusion of malicious agents before they are detected as liars. If this value is low, new honest agents are disadvantaged in comparison with other honest agents and that will lead the system to become less and less open.

This emphasizes the necessity to add a special value `unknown` to initialize trust values. The difficulty to add a value in addition to the set of numerical values is that the decision process that relies on reputation values must take into account this specific value [22]. Such a decision process is presented in the next section.

4.4 Decision Process to Trust or Distrust

When the beneficiary receives an information from another agent, reputation values are used to decide if this other agent should be trusted or not. We propose a decision mechanism that orders the different reputation values. In Figure 9, we as-

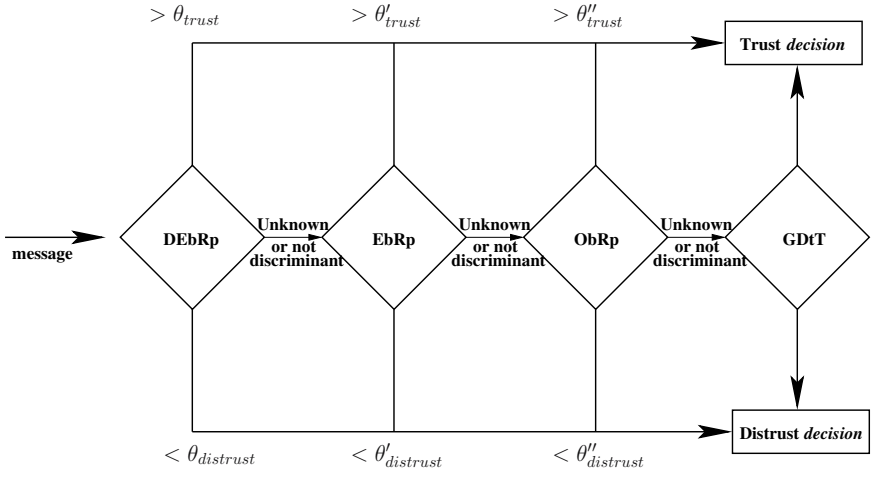

Fig. 9. Using reputation values to decide whether to trust or not

sume that DEbRp is more reliable than ObRp which is more reliable than EbRp. The GDtT is the less reliable sort of reputation value, since it is considered as a default value. The justification of this choice of order is due to the detection process in which (i) other agents can lie when providing observations and/or evaluations (ii) observations are done through the other agents' sensors, for which the beneficiary does not know the reliability and (iii) evaluations may be based on the personal interpretation of the evaluator that may diverge from the beneficiary's one.

The decision process works as follow: the agent first considers its most reliable reputation values, here, its DEbRp. The DEbRp value might be sufficient to decide to trust (respectively distrust) a target, if it has a high (respectively low) value. If the DEbRp is greater than θ_{trust}, then the agent decides to trust the target. At the opposite, if it is less than $\theta_{distrust}$, then the agent decides to distrust the target and rejects the message. In the other cases, DEbRp is in the `unknown` state or has a value between the two thresholds $\theta_{distrust}$ and θ_{trust}, the DEbRp value is not sufficient to take a decision. Therefore, a similar decision function is applied to the next most reliable trust value: ObRp. The ObRp is compared with two others thresholds θ'_{trust} and $\theta'_{distrust}$. If this value does not allow the agent to take a decision, the EbRp is used. As a last resort, the GDtT is used. At the end of this mechanism, the agent has taken a decision whether to trust or not the target for the message that the target has just sent.

5 Related Works

The work presented in this paper is related to two categories of existing works. The first category is the formalization and detection of frauds. The second category is the management of reputation values.

Most works [23, 24, 25] in the domain of fraud detection consider applying data mining algorithms to large databases of transaction histories. In our case, agents can not consider applying such algorithms. In fact, these algorithms are not adapted to run on the fly: (i) implemented agents would be heavy weighted and very slow, (ii) agents only possess small parts of overall exchanged communications, therefore the success rate of detection would be low. Lomuscio et *al.* [26] formalize violation in the bit transmission problem with deontic logic. Their axiomatization is based on a representation of the world as a global state, which is composed of the state of each agent of the system plus the state of the environment. In an open system, it is impossible to enumerate every state each agent may be in, since agents may come from different designers and their internal implementations are not available. Firozabadi et *al.* [27] suggest a formalization of fraud in two parts: a formalization of violation and of deception. They propose a logical formalism to represent how an agent should behave (obligations) and what an agent has done (actions) in order to define a situation of violation and deception is represented by considering the agents' local beliefs. We proposed to extend their work in the domain of agent communication [28], but representing agents in terms of mental states suffers various drawbacks [29]. Pasquier et *al.* [11] propose a flexible model of commitments that integrates sanctions (fixed or negotiated) to be applied in case of violation of the commitment either by the debtor or the creditor. However, their model do not address social sanctions (which include reputation).

Reputation modeling is the second category of related works. Several different proposals [3, 1, 7, 5, 6] exist to represent and to evaluate the reputation of some agents. However, most of these works focus on the evaluation of reputation but does not consider the very first step of this evaluation: the detection of a fraud.

[30] suggest to use a finite set of deceptive behaviors while gossiping. They apply a learning algorithm to recognize agents that follow one of this behavior. Our work follow a different approach since we do not need to identify every deceptive behavior that can occur, but we rather define what is the correct behavior that agents must have. Thus, we think that our approach is more adapted to open systems because we do not have to predict in advance every deceptive behavior.

In [3, 31] various reputation values are used, from different classes of trust [21]: categorial, interpersonal, etc. and are merged into a single value on which the decision is based, whereas in this paper, the various reputation values mostly are interpersonal and represents different points of view from an agent about another. We maintain separated values because we consider they are not all used in every situations. Also, in their model, the update of the reputation values occurs by comparing the execution of contracts to what have been previously negotiated between the parties, whereas our social commitments can be viewed as implicit contracts, that are not negociated.

Finally, our model allows the representation of reputation in continuous intervals (to model the partial ordering of an agent acquaintances) and includes a specific value for initialization as underlined in [22].

6 Conclusion

All the related works quoted in the previous section focus on one specific category without considering the others. One of the main interest of the work presented here is that it integrates the detection of lies, the representation of reputation and its use to prevent future deceptions. Moreover, this mechanism is decentralized and does not require the existence of a central entity.

This work can still be extended in a few directions. A first extension consists in the formalism that have been chosen to represent violations and lies. A violation has been represented as an action that contradicts a norm expressed in deontic logic. It would be interesting to integrate some existing works in the formalization of norm [32, 33, 34] and benefit from the definition of specific behaviors in case of detection of a violation. This would imply to use a more complete and flexible commitment model such as [12, 11] and to include social sanctions in terms of reputation.

Another extension of this work would be to use lie detection in the messages exchanged *during* lie detection. If an agent lies when it sends a message containing an observation or an evaluation of a target, it can lead some other agents to believe that a liar is sincere (or the opposite). This problem is the main reason why different levels of reliability are distinguished for reputation in Sec. 4.1. In fact, messages containing some observations or evaluations may be considered as classical messages and may also be detected as lies or be rejected if they are sent by an agent with a low reputation.

References

1. Schillo, M., Funk, P.: Who can you trust: Dealing with deception. In: In Proceedings of the workshop Deception, Fraud and Trust at the AA'99. (1999) 95–106
2. Conte, R., Paolucci, M.: Reputation in Artificial Societies. Social Beliefs for Social Order. Kluwer Academic Publishers (2002)
3. Sabater, J., Sierra, C.: Social regret, a reputation model based on social relations. SIGecom Exchanges. ACM **3.1** (2002) 44–56. http://www.acm.org/sigecom/exchanges/volume_3_(02)/3.1-Sabater.pdf.
4. Castelfranchi, C., Falcone, R.: Principles of trust in mas: Cognitive anatomy, social importance, and quantification. In: Proceedings of ICMAS'98. (1998) 72–79
5. Dellarocas, C.: Analyzing the economic efficiency of eBay-like online reputation reporting mechanisms. In: Proceedings of the 3rd ACM Conference on Electronic Commerce (EC-01), New York, ACM Press (2001) 171–179
6. Sen, S., Sajja, N.: Robustness of reputation-based trust: boolean case. In: AAMAS 2002. (2002) 288–293
7. Zacharia, G., Moukas, A., Maes, P.: Collaborative reputation mechanisms in electronic market places. In: Proceedings of the 32nd Hawaii International Conference on System Sciences. (1999)
8. Androutsellis-Theotokis, S.: A survey of peer-to-peer file sharing technologies. Technical Report WHP-2002-003, ELTRUN (2003). http://www.eltrun.aueb.gr/whitepapers/p2p_2002.pdf.

9. Gnutella: Gnutella 0.48 Specifications. (2000). http://capnbry.net/gnutella/protocol.php.
10. Fornara, N., Colombetti, M.: Defining interaction protocols using a commitment-based agent communication language. In: Proceedings of the AAMAS'03 Conference. (2003) 520–527
11. Pasquier, P., Flores, R.A., Chaib-draa, B.: Modelling flexible social commitments and their enforcement. In: Proceedings of ESAW'04. (2004)
12. Bentahar, J., Moulin, B., Chaib-draa, B.: Towards a formal framework for conversational agents. In Huget, M.P., Dignum, F., eds.: Proceedings of the Agent Communication Languages and Conversation Policies AAMAS 2003 Workshop. (2003) July 14th 2003, Melbourne, Australia.
13. Hamblin, C.: Fallacies. Methuen, London (1970)
14. Walton, D., Krabbe, E.: Commitment in Dialogue. SUNY Press (1995)
15. Simmons, M.R.: Recognizing the elements of fraud (1995). http://users.aol.com/marksimms/mrweb/fraudwww.htm.
16. Tuomela, R.: The Importance of Us: A Philosophical Study of Basic Social Norms. (1995)
17. Tuomela, R., Bonnevier-Tuomela, M.: Norms and agreement. European Journal of Law, Philosophy and Computer Science 5 **41-46** (1995)
18. von Wright, G.: Deontic logic. In: Mind. Volume 60. (1951) 1–15
19. PGP: DTfinition of non-repudiation (2004) http://en.wikipedia.org/wiki/Non-repudiation.
20. Morge, M.: A dialogue game for agents resolving conflicts by verbal means. In: Proc of the 2nd worshop on Logic and Communication in Multi-Agent Systems. (2004) to appear.
21. McKnight, D., Chervany, N.: Trust and Distrust Definitions: One Bite at a Time. In: Trust in Cyber-societies. Springler-Verlag Berlin Heidelberg (2001) 27–54
22. Grandison, T.: Trust specification and analysis for internet applications (2000)
23. Jensen, D.: Prospective assessment of ai technologies for fraud detection: a case study. In: AI Approaches to Fraud Detection and Risk Management edited by T. Fawcett, I. Haimowitz, F. Provost, and S. Stolfo, AAAI Press (1997) 34–38
24. Cahill, M.H., Lambert, D., Pinheiro, J.C., Sun, D.X.: Detecting fraud in the real world. Kluwer Academic Publishers (2002)
25. Abu-Hakima, S., Toloo, M., White, T.: A multi-agent systems approach for fraud detection in personal communication systems. In: Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI-97). (1997) 1–8
26. Lomuscio, A., Sergot, M.: A formalisation of violation, error recovery, and enforcement in the bit transmission problem. Journal of Applied Logic (selected articles from DEON02 - London) **1** (2003) A previous version of this paper appeared in the proceedings of DEON02.
27. Firozababdi, B.S., Tan, Y.H., Lee, R.M.: Formal definitions of fraud. In: Proceedings of the Fourth International Workshop on Deontic Logic, DEON'98. (1998)
28. Muller, G., Vercouter, L.: Liar detection within agent communication. In: Proceedings of LCMAS'04 – ESSLI'04. (2004)
29. Singh, M.P.: Agent communication languages: Rethinking the principles. In Huget, M.P., ed.: Communication in Multiagent Systems. Volume 2650 of Lecture Notes in Computer Science., Springer (2003) 37–50
30. Yu, B., Singh, M.P.: Detecting deception in reputation management. In: Proceedings of Second International Joint Conference on Autonomous Agents and Multi-Agent Systems. (2003) 73–80

31. Sabater, J., Sierra, C.: Reputation and social network analysis in multi-agent systems. In: Proceedings of AAMAS'02. (2002) 9
32. Dignum, F.: Autonomous agents with norms. In: Artificial Intelligence and Law. Volume 7. (1999) 69–79
33. Conte, R., Castelfranchi, C., Dignum, F.: Autonomous norm acceptance. In: Proceedings of ATAL 1998. (1998) 99–112
34. Vàzquez-Salceda, J., Dignum, F.: Modelling electronic organizations. In: Proceeding of CEEMAS'03. (2003) 584–593