Safety in Multi-Agent Systems: Reputation Based on Dossier

André P. Borges, Vanderson Botêlho, Osmar B. Dordal,
Bráulio C. Ávila, Edson E. Scalabrin
Pontifíficia Universidade Católica do Paraná
Rua Imaculada Conceição, 1155
Curitiba, Paraná - Brazil 80215-901

Abstract

A multi-agent architecture possesses versatility when solving complex problems in various research fields. In open systems, in which agents are unknown to each other a priori, their trust relationship is an essential subject. Hence, we present a reputation model based on dossier, which grants authenticity and reliability of exchanged information to its participants. This model has been applied to a classical problem in the railway sector, e.g., to define economical driving plans for freight trains. At least three agents are placed for each station, to plan, operate and manage plans. Planning and management skills are conducted by stationary agents, and the operating skills are conducted by mobile agents. Each locomotive has in its on-board computer a container for accommodating agents and transport mobile agents, either to complete an assignment or return to the station of origin. Empirical results show that sharing experiences improves the efficiency of the generation of plans, and that the use of the reputation based on dossier guarantees the veracity of the testimonies, therefore decreasing the need to centralize the information.

Introduction

A multi-agent system is by essence composed of intelligent agents who are able to respond to environmental stimuli, present goal-oriented behaviours, and socially interact with other agents (Wooldridge and Jennings 1995). Moreover, this type of system does not possess a system control center, and the data are decentralized. Every open multi-agent system possesses an additional characteristic, i.e., the capacity to host agents that are *a priori* unknown, like electronic commerce (Delecroix, Morge, and Routier 2014), cloud computing (De la Prieta et al. 2013). However, if this system development approach is more attractive due to its potential heterogeneity, the lack of a centralized information control center decreases the trust relationship among agents. The main question for an open system (Huynh, Jennings, and Shadbolt 2006) is how an agent can trust another unknown agent.

This question has motivated a large number of researches regarding the concept of *trust* as a key element for the interaction of intelligent agents in an open system with a certain degree of trust. For the past two decades, various trust models have been proposed, which can be classified into two main classes according to their information source: direct and indirect models (Mui, Mohtashemi, and Halberstadt 2002). In the first class, each agent builds its trust model based on its own experiences/perception, whereas in the second class, trust is obtained from the experiences of other agents, which is also known as their *reputation* (Sabater and Sierra 2001). Although agents may use both sources of information concurrently, the reputation becomes more relevant in that virtual communities grow and direct experiences become more costly because the effort necessary for all agents to know each other directly grows exponentially. An agent's reputation, normally obtained through experience sharing, aims at solving the limitations of direct models; however, this creates new challenges: How must an agent be motivated to share its experiences? How can good witnesses be found? How can the veracity of testimonies be guaranteed? An alternative to the first two questions was proposed by (Botêlho et al. 2011) in the form of a *Certified Trust* model. Meanwhile, the presence of malicious agents, aiming at manipulating reputation systems for their own advantage, is a current problem for open systems. Hence, we propose an evolution of the previously mentions *Certified Trust Model* as a way to effectively deal with the third question. Such proposal includes a reputation system called Reputation based on Dossier, which assures the authenticity and integrity of the transmitted evaluations.

Related Studies

Various studies have been conducted with the aim of minimizing the inherent risks of interactions among agents in an open system. The first works following this path were by Griffiths (Griffiths 2005) and Marsh (Marsh 1994), who suggested the sole use of direct experiences for estimating the reliability of an evaluated agent. However, such a direct way to obtain information may be inefficient when applied to large communities of agents. As an alternative, we can obtain indirect information through observation. Trust by observation is applied when an agent evaluates the behaviours of its peers by observing their past interactions (Sabater and Sierra 2001). Under this type of situation, an agent's reputation is obtained through indirect sources of information. Other indirect approaches have created *reputation* systems that may be transmitted as reports or recommendations. Such reputation systems are the most popular owing to their usage in important *e-commerce* platforms, e.g., eBay and Amazon. We must also stress that there are more recent works dealing with the relationships among agents that take emotional aspects into account, e.g., a study by Dias e Paiva (Dias and Paiva 2013). In this example, the proposal described is for a computational model identified as *Interpersonal Emotion Regulation*. This model allows the relationship among agents to be represented by emphasizing their social relationships and friendship levels.

Reputation systems have been used in several problems where a certain degree of reliability must be achieved. In (Hartong, Goel, and Wijesekera 2008), the authors presented a wireless control system for ensuring safety in inter-company railway operations, by enforcing train separation, speed enforcement, roadway worker protections and other safety functions.

Our work focuses on the safety of sharing driving plans generated by a group of specialized agents with a good *reputation*. Beyond information, these *Executor* agents are able to move throughout a network of railway stations and share plans in a safe and reliable manner. Agents move from host to host in a DTN (disruption-tolerant networking) type of network (Voyiatzis 2012).

The reputation model used, along with the safety mechanisms, is presented and discussed in the following chapter.

Reputation based on Dossier Model

An indirect trust model, based on the Reputation by Dossier, may be described using the following scenario: service provider agent p provides a service to consumer agent c. Agent c evaluates the provided service and sends feedback f to agent p. Agent p stores f locally. The portfolio of evaluations received and stored by p is referred to as a *dossier*. The dossier of p is used as testimony to itself and becomes accessible every time another agent must verify the trustworthiness of p. This way, for the given interaction i, agent c evaluates agent p by giving value v for term t. All feedback may be related to other feedback f_l. This is represented by $f = (c, p, i, v, t, f_l)$.

Until now, we can say that such representation addresses two common problems for the current reputation models for open systems: the lack of interest of agents in sharing their experiences, and as the community grows, increase the number of messages needed to locate reliable witnesses.

Feedback is stored in the evaluated agent. Therefore, no other agents need to testify. The main advantage of this type of situation is that consumer agent c_j does not need to apply a sophisticated approach to find good witnesses because the feedback is already located in the dossier of each service provider agent p_k.

Trust T of agent c in agent p for the given term t is calculated by agent c through the means of a weighted average of all stored feedbacks in dossier D_p (Eq. 1).

$$T(p,t) = \frac{\sum_{f_i \in D_p(t)} \alpha(f_i) \times v_i}{\sum_{f_i \in D_p(t)} \alpha(f_i)} \tag{1}$$

Average weighting is necessary to decrease the relevance of the feedback as time passes. Such decrease is determined based on factor α. More recent feedback becomes more relevant than older feedback. Factor α is obtained through the following exponential function (Eq. 2):

$$\alpha(f_i) = e^{-\frac{\Delta t(f_i)}{\mu}} \tag{2}$$

where $\Delta t(f_i)$ represents the elapsed time between the moment the feedback was created and the moment in which the trust calculation was performed; and μ is the factor determining the rate of decrement of the exponential function.

Another important aspect of the Reputation by Dossier method is the guarantee of only legitimate feedback, because open communities are vulnerable to malicious agents that try to impersonate other agents or modify their dossier in order to benefit improperly. To avoid such an impersonation, asymmetric cryptography may be applied to the signature of any transmitted information (Foner 1999).

Each agent receives a pair of keys while registering in the system: a private key, *Kpri*, and a public key, *Kpub*. Private keys are dealt with secretly by their owner agents, and public keys are distributed freely. When agent a sends feedback f_0, it calculates the function hash of f_0, resulting in $h_0 = hash(f_0)$ and cryptography h_0 with its private key $s_0 = encrypt(h_0, Kpri(c))$, where s_0 is referred to as the signature of f_0. Hence, when a sends f_0, it also sends s_0. The recipient agent can verify whether f_0 was created by c by deciphering s_0 through a's public key $s'_0 = decipher(s_0, Kpub(c))$; if s'_0 is identical to h_0, it is fair to assume that f_0 was created by agent c.

Under certain circumstances, some agents may limit their communication to agents that belong to a single organization. Meanwhile, whenever necessary, it is possible to create system credited agents for monitoring purposes, which the remaining agents may trust as entities of the system. For this, it is necessary to create a Certificate of Authority (CA), which represents the system mechanism that is responsible for signing the pairs of keys of its accredited agents. This mechanism makes it possible to verify whether a message sent by these *Inspector* agents is also certified by the CA of the system.

After reassuring by means of a digital signature that the feedback is immutable and that the authorship is verifiable, it is crucial to guarantee that the evaluations that are a part of the file cannot be removed for the benefit of their owner agent. This problem is solved using a *linked feedback* structure (illustrated in Figure 1). When agent p is registered for the first time in the system, it receives neutral feedback by *Inspector* agent i. This agent has a private key issued by the CA of the system, and it is therefore possible to verify that it is a credited agent. The first feedback about agent p is $f_0 = (c, p, i_0, v, t, null)$. A $null$ reports that this feedback does not have a link with any other feedback. Each agent has a single feedback using link $null$, which is received when the agent is registered in the system. The initial feedback cannot be adulterated because it is signed by an Inspector agent; such feedback f_0 marks the initial state of the dossier. When c sends feedback to p, this feedback will

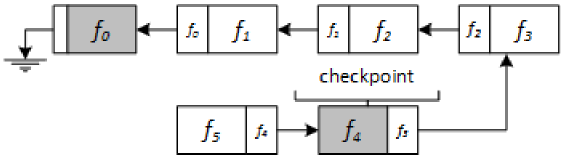


Figure 1: Linked feedback

be linked to the most recent feedback regarding p, for instance, $f_1 = (c, p, i_1, v, t, f_0)$. Consequently, agent c_k may give feedback $f_2 = (c, p, i_2, v, t, f_1)$ and so on. *Inspector* agents are also responsible for sending periodical checkpoint feedback to prevent an agent from removing its evaluations. Inspector agents publicly store the last checkpoint of each agent. Thus, any agent may verify whether the last feedback published by the Inspector agents is available in a given dossier. The time intervals for checkpoints to take place may be customized to meet the needs of each community.

In cases in which two or more consumer agents interact at the same time with a service provider agent, the order of feedback in the dossier will be determined based on the moment it is sent. The first dispatched feedback will be made effective, and the remaining feedback will be denied. In this type of case, the consumer agent will receive an updated dossier from the service provider for its feedback to be sent back to the providers, for whom the feedbacks were denied, with a link to the most recent feedback of the dossier. This process continues until all feedback has been sent.

The Reputation by Dossier achieves two conditions: 1) it prevents the evaluated agent from modifying its feedback; and 2) it prevents feedback from being omitted from the dossier. *Condition 1* is met using a digital signature for all feedback, and *condition 2* is met using a feedback link structure, which makes the dossier indivisible. From the feedback data structure, it is possible to verify whether the dossier contains all of the feedback f. The algorithm 1 allows the integrity of all f and all of *Dossier D* to be verified:

Algorithm 1 Verify integrity

```
Require: a dossier $D$
$f = lastfeedback(D)$
while $f \rightarrow link \neq null$ do                ▷ condition 1
    if $isNotSigned(f, f \rightarrow a)$ then
        return false
    end if
    $f = f \rightarrow link$
end while
if $isSigned(f, inspector)$ then                     ▷ condition 2
    return true
else return false
end if
```

For *condition 1*, the dossier will be examined for its integrity. All feedback is compared against the author's signature. The interlink of the feedback may also be tampered with because the hash value of the feedback can be modified and easily identified using a cryptography mechanism. For *condition 2*, when the initial feedback f_0 is reached, it is verified whether it has been issued by an inspector agent based on its signature. If both conditions are met, it is fair to assume that the dossier has not been altered and that no feedback has been removed.

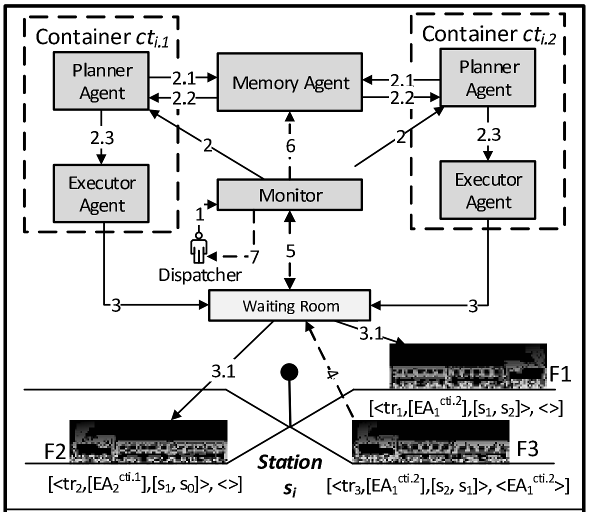
Case Study

To evaluate the Reputation by Dossier approach, its application has been studied for an intelligent system in the railway field with agents specialized in defining and executing the driving plans of the trains. The railway network is shown as a graph, where nodes represent physical or logical stations and edges have information about the railway (i.e., the track profile). Figure 2 shows the basic configuration of station s_i. Each station $s_i \in S$ hosts a set of containers $Ct_i = ct_{i.1}, ct_{i.2}, ..., ct_{i.n}$. Each container ct_i is a hosting environment for agents dedicated to a specific stretch of the railway. Each stretch is defined by the track between two stations, s_i and s_{i+1}. The pair $[s_i, s_{i+1}]$ defines the origin and destination of train tr_1. Any configuration may be expressed through the tuple $[tr_1, [EA_1^{ct_{i.2}}, P^1], [s_1, s_2], < L[s_1] >]$, where tr_1 represents the train to be conducted from station s_1 to station s_2 by the *Executor* agent $EA_1^{ct_{i.2}}$, using plan P^1 that has been defined by *Planner* agent $PA_1^{ct_{i.2}}$, and $L[s_1]$ represents the list of *Executor* agents that returned to their origin stations s_1.

Each container ct_i contains at least a *Planner* agent and an *Executor* agent, which are responsible for planning and executing the driving plans of the trains for a specific stretch of the railway network, respectively. The *Planner* agent must gradually become an expert along the stretch it has been allocated. Each planned action may assume one of the following behaviours: accelerate, maintain, or reduce the speed. Each behaviour is adjusted in accordance with the given power, generated from the application of an acceleration point $AP \in -1, , 8$. Each *Planner* and *Executor* agent may be seen as a practical agent of its given stretch, similarly to what happens to experts that maneuver large ships in a harbour. The *waiting room* is a container of agents, and is present in every station and temporarily hosts the *EA1*s. The *Monitor* agent is responsible for moving the agents from the waiting room to the internal containers of the station and vice versa. The *Informer* agent spreads the feedback that contains the experience between neighbour stations using a *Breadth-First* method.

We assume that communication between agents from different stations has a low connectivity because this is a similar situation to that of a real environment. The agents are limited to communicating with only other agents from the same container. For an agent to interact with a similar agent from another station, it must board a container of the train and disembark at its station of destination. The mobility of the agents between containers takes place when the train physically approaches the station.

Figure 2 also shows three basic flows: F1, F2, and F3. F1 lists the execution of a set of activities, which initiates when the *Monitor* receives a demand (1) from the *Dispatcher* to drive train tr_1 from station s_1 to station s_2. It terminates when the *Executor* agent $EA_1^{ct_{i.2}}$ boards on tr_1 to drive it (3.1), using driving plan P^1 that has been defined by *Plan-*



Flows:

F1 (departing from s_i to s_{i+1}): 1; 2; 2.1; 2.2; 3; 3.1
F2 (departing from s_i to s_{i-1}): 1; 2; 2.1; 2.2; 3; 3.1
F3 (arriving from s_{i+1} in s_i): 4; 5; 6; 7

Subtitle:

1. Dispatch order reception, from the Dispatcher.
2. Request for the generation of a plan P^i
2.1 Request of similar cases
2.2 Return of similar cases
2.3 Delivery of P^i for execution
3. Entry on the waiting room, for boarding
3.1 Boarding of the Executor on the train in order to begin to execute P^i
4. Disembarkation of the Executor in the Waiting Room
5. Perception of the Executor's arrival with the executed plan P^{i}'
6. Delivery of P^{i}' to the Memory agent
7. Return of feedback to the Dispatcher

Figure 2: Configuration and flows of station s_i.

ner agent $PA_1^{ct_{i.2}}$. Flow F3 represents the return of *Executor* agent $E_1^{ct_{i.2}}$ from destination s_2 to station s_1. This flow terminates when plan P^1', applied by the *Executor* agent, is passed on to the *Memory* agent, whose main goal is to update the base of the cases. Here, P^1' represents plan P^1 with the adjustments (when necessary) that the *Executor* agent has to make during the course of operating the train.

Hereafter, a scenario with container ct_i, an *Executor* agent (EA1), and a *Planner* agent (PA1), originating at station s_i and terminating at station s_{i+1}, will be considered.

Reputation Calculation The reputation management was made using the Reputation based on Dossier model. The reputation calculation takes into account the efficiency in retrieving similar cases and adapting them to driving plans. These plans are specific for a given stretch of the railway network. Each stretch has an associated reputation, R. The reputation R for any given stretch, St_i (cf. Eq. 3), is calculated based on the reputation of railway v ($rRail(v)$), the reputation of train T ($rTrain(T)$), and the services provided by the agents ($rService(ser)$).

$$R(St_i) = \frac{rRail(v) + rTrain(T)}{2} \times rService(ser) \quad (3)$$

The reputation of the railway, calculated as $rRail(v)$, may be measured as a function of the profile of the stretch of railway (vertical and horizontal), the strain on the rail system (Ekberg 1997), the number of journeys along that particular stretch, the cargo transported, the maximum allowed speed, and the average speed along the stretch on the rail alignment, among other factors (Sta 2009). The reputation of the train, calculated as $rTrain(T)$, takes into account such factors as the maintenance intervals and the vehicle condition. These factors influence the overall stress of the train. A set of factors may be analysed from the point of view of dynamic behaviour in accordance with the standard UIC Leaflet 518 for the acceptance of international traffic (Sta 2009). Finally, the reputation of a provided service, given by $rService(ser)$, which is the main focus of this study, refers to the efforts of an agent while elaborating on and implementing the plans. The reputation is individually calculated for each agent by receiving feedback on the provided services.

Table 1: Train configuration used in the experiments.

Train	Locomotives	Railway cars	Weight (tons)
1	3	58	6278
2	4	100	6342
3	4	58	6541
4	2	31	3426
5	3	47	5199
6	2	31	3441
7	4	59	6579
8	2	28	3118

The *MA1* provides services to the *PA1* by providing driving plans that meet its needs; hence, it is the *Planner*'s job to evaluate the reputation of the *MA1*. The reputation of the *Planner* is evaluated by the *Executor*, who analyses its efforts in adapting plan P during its execution. The final reputation of each stretch is calculated by the *Dispatcher* at the end of each journey. In this study, constant and homogeneous values were assumed for the reputations of the train and stretch to simplify their calculations.

Experiment Settings

The experiments were conducted in a simulated driving environment, and field equations (Luke et al. 2014) were implemented in Java and evaluated using several experiments. The profiles of the railways, trains, and initial base case were derived from real situations (see Table 1).

The safe sharing of dossiers has been guaranteed using a pair of keys: a private key, *Kpri*, and a public key, *Kpub*. Such mechanism guarantees that the reputation of each agent, resulting from its feedback, is kept safe and remains unaltered and complete.

To evaluate the learning curve of each agent and the performance of the collaboration in terms of the sharing and reuse of the plans, four scenarios were defined (see Table 2). An evaluation of previously implemented plans, whose propagation is managed by the *Informer* agent, is made as the plans return to their station of origin, regardless of the travelled route, and by reusing plans along a different stretch than the one for which the plans were first generated (e.g., scenario C).

The initial base case of the *MA1* in all tested scenarios contains actual journey plans and journeys executed in a simulator (Sato et al. 2012). The adaptation step of the *Plan-*

Table 2: Scenario configuration used in the experiments. St_1 and St_2, both with the same length (64 km), but with different profiles and maximum speed restrictions.

Scenario	Train (Table 1)	Reuse plans	Stretch
A	1	No	St_1
B	1	Yes	St_1
C	1	Yes	St_2
D	[1;8]	Yes	St_1

Table 3: Results obtained.

		Journey (%)											
Scenario		**B10**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Average**
A	Memory		40	41	44	42	39	43	39	42	46	42	42
	Planner (P)		48	50	51	49	47	50	46	49	53	49	49
	$P - M$		8	9	7	7	8	7	7	7	7	7	8
B	Memory (M)		47	58	69	70	67	71	67	76	72	77	67
	Planner (P)		54	64	74	77	74	78	73	81	78	82	74
	$P - M$		7	6	5	7	7	7	6	5	6	5	6
C	Memory (M)	77	77	79	79	78	81	80	83	80	80	83	80
	Planner (P)	82	86	85	85	83	85	86	89	86	85	88	86
	$P - M$	5	9	6	6	5	6	6	6	6	5	5	6
D	Memory (M)	77	73	61	71	52	59	68	65	70			65
	Planner (P)	82	80	70	80	61	66	76	73	79			73
	$P - M$	5	7	9	9	9	7	8	8	9			8

ner used a simple evolutionary approach, i.e., a genetic algorithm (Luke et al. 2014)(Baeck, Fogel, and Michalewicz 2000). The stopping criterion of the strategy was the number of generations, i.e., ten.

This approach was evaluated using different metrics: (i) the accuracy of the case recovery task, (ii) the efficiency of adaptation and application of such cases, and (iii) the fuel consumption in *Litres per Gross Ton Transported* (LGTT). The (i) and (ii) indicate the reputation of the *Memory* and *PA1*s. The (iii) indicates the evaluation of the *EA1*. Each journey generates two types of feedback, one for the *MA1* (after the planning), other for the *PA1* (after the execution), which are both locally stored in their respective dossiers. The reputations of the *MA1* and *PA1* are evaluated according to the efficiency (%) of the recovery and adaptation steps of the cases for each journey over time. Table 3 summarizes the results.

In scenario A, the *MA1* only uses the initial base case. At the end of each journey, the new experiences (new plans) were not incorporated in the base case of the *MA1*, i.e., the plans were not distributed by the *Informer* agent. The *Memory* reputation was observed to be lower than the *Planner* reputation. In scenario B, every journey made by the *EA1*, the applied plan was incorporated into the base case of *MA1*. This inclusion of feedback increased the reputations of the *MA1* and *PA1*s by $\approx 25\%$. Between journeys 1 and 3, there is an increasing linear trend of 20%. Moreover, from journey 3, there is a slight increase in stability. In scenario C, for each new journey made by the *EA1*, the applied plans are incorporated into the experience base of the *MA1*, and are thus reused by the *PA1*. The effectiveness of the distribution of the plan by the *Informer* agent for a different stretch from that for which the plan was generated. The base case of the *MA1* began with the experiences generated in scenario B. The *MA1* and *PA1* reputations were $\approx$ 80% and $\approx 86\%$, respectively. Despite the increased efforts owing to the unfamiliarity with the environment, these results are significant, encouraging the use of a secure and collaborative approach between agents, located at different stations, to exchange plans though the *Informer* agent. In scenario D, *MA1* initiates the experiences of scenario B. The reputation of the *Memory* and *PA1*s reflect the precision of the recovery and the adaptation tasks, with *PA1* proved to be higher. There was an expected drop in the rate of success because the train configurations differed for each journey. However, even under this scenario without a repetition of the train configuration, it is possible to note that, as new cases were included in the base, the overall efficiency improved. Hopefully, with a greater number of journeys with similar configurations, the efficiency rates will move rapidly toward scenario B.

Table 4: Consumptions obtained for scenario C.

Train	**Consumption (LGTT)**		**Reduction**		
	Actual (A)	**DCOP (B)**	**Our (C)**	**C-A** (%)	**C-B** (%)
1	6.19	4.16	3.36	50%	26%
2	5.68	4.18	4.22	30%	-5%
3	6.23	4.09	3.95	41%	10%
4	6.49	4.51	3.88	46%	23%
5	6.29	4.22	3.31	49%	24%
6	6.17	3.99	3.69	40%	8%
7	6.26	4.07	3.86	42%	11%
8	5.68	4.41	4.00	34%	6%

Table 5: Reputations obtained.

Scenario	**MA1**	**PA1**	$\overline{\textbf{RS(St)}}$
A	42.5%	50.0%	46.3%
B	74.9%	80.4%	77.7%
C	81.9%	86.9%	84.4%
D	68.2%	76.9%	72.6%

Under the scenarios in which we operated, it can be observed that, without reusing plans as past solutions, the average success ratios in the recovery and adaptation tasks remained low, i.e., 42% and 49% for the *MA1* and *PA1* reputations, respectively. However, when we started to reuse the plans as past solutions, the average reputation ratios of these agents increased to 65% and 73%, respectively.

Table 5 shows the reputation of the applicable services for a particular stretch (Eq. 3), which was obtained through averaging the reputation of the *MA1* and *PA1*s throughout their journey. The reputation for each stretch was not calculated. For the case in point, a value of 1 was considered.

Over time, the learning capacity, especially that of the *PA1*, must result in a reputation of above 80%. For lower percentages, the system should generate a low-quality service alert. In general, under all observed situations, the effort to adapt is present, efficient, and increases over time.

Table 4 contrasts the performance of human drivers (the Actual column) driving a simulator where the applied actions are determined through a constraint satisfaction system (DCOP column) (Sato et al. 2012) and by an *EA1* (Our column). The DCOP column represents the best values obtained by this approach. It should be emphasized that, for all consumption values (measured in LGTT), our approach is higher than that of the other competitors, with the exception of a single opportunity, where the DCOP is 5% higher. The feasibility of an automatic train driving system seems to be significant. For example, for a fuel consumption expenditure of approximately 250 million dollars per year, any cost savings above 6% can have a significant impact on the competitiveness of the freight transport sector.

Conclusions and Future Works

This paper addressed the problem of security risks in the interactions among participant agents of an open system. Inspired by studies that show trust models as effective tools in managing such risks, we propose a differentiated approach to reputation management using the concept of a dossier, in which the agent under evaluation provides its previous feedback to its evaluators. Such an approach handles the search for good witnesses and motivates them to share their experiences. Consequently, the risk of a given evaluated agent tampering with its dossier does arise. This was resolved, however, through the use of asymmetric cryptography mechanisms associated with the concept of linked feedback, which guarantees the integrity of the dossier in terms of alterations or omissions of information.

To validate the proposed model, it was tested using a railway system model, in which the agents are able to interact and move freely between stations and trains, aiming at generating, executing, and sharing the driving plans. Because this is an area that involves high costs and safety risks, the main contribution of the Reputation by Dossier method is allowing for transmitted information between agents to be verified in terms of its integrity, absence of alterations, and authenticity, assuring the veracity of its authorship, which is guaranteed through the use of security keys.

Our experiments show that the sharing of experiences between agents of the open community (railway network) may improve their efficiency in generating driving plans. In a low connectivity environment, this approach becomes more relevant when aggregated with information security mechanisms, as proposed by the Reputation based on Dossier. The set of experiences generated by the community made it possible to create plans for new stretches of railway with lower learning costs compared to a non-collaborative approach.

As future works, we intend to evaluate the reputation system including the reputations of the trains and railway tracks. Additionally, we intend to evaluate the Reputation by Dossier model in other open scenarios, confronting it with classic models of trust and reputation vis-a-vis scalability, assertiveness and execution time.

References

Baeck, T.; Fogel, D.; and Michalewicz, Z. 2000. *Evolutionary Computation 1: Basic Algorithms and Operators*. Basic algorithms and operators. Taylor & Francis.

Botêlho, V.; Enembreck, F.; Ávila, B.; de Azevedo, H.; and Scalabrin, E. 2011. Using asymmetric keys in a certified trust model for multiagent systems. *Expert Systems with Applications* 38(2):1233 – 1240.

De la Prieta, F.; Rodrguez, S.; Bajo, J.; and Corchado, J. 2013. A multiagent system for resource distribution into a cloud computing environment. In Demazeau, Y.; Ishida, T.; Corchado, J.; and Bajo, J., eds., *Advances on Practical Applications of Agents and Multi-Agent Systems*, volume 7879 of *Lecture Notes in Computer Science*. Springer. 37–48.

Delecroix, F.; Morge, M.; and Routier, J.-C. 2014. Bilateral negotiation of a meeting point in a maze: Demonstration. In Demazeau, Y.; Zambonelli, F.; Corchado, J.; and Bajo, J., eds., *Advances in Practical Applications of Heterogeneous Multi-Agent Systems. The PAAMS Collection*, volume 8473 of *Lecture Notes in Computer Science*. Springer. 327–330.

Dias, J. a., and Paiva, A. 2013. I want to be your friend: Establishing relations with emotionally intelligent agents. In *Proc. of the International Conference on Autonomous Agents and Multi-agent Systems*, AAMAS '13, 777–784. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems.

Ekberg, A. 1997. Rolling contact fatigue of railway wheelsa parametric study. *Wear* 211(2):280 – 288.

Foner, L. N. 1999. *Political Artifacts and Personal Privacy: The Yenta Multiagent Distributed Matchmaking System*. Ph.D. Dissertation, Massachusetts Institute of Technology. AAI0801075.

Griffiths, N. 2005. Task delegation using experience-based multi-dimensional trust. In *Proc. of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '05, 489–496. New York, NY, USA: ACM.

Hartong, M.; Goel, R.; and Wijesekera, D. 2008. Trust-based secure positive train control (ptc). *Journal of Transportation Security* 1(4):211–228.

Huynh, T. D.; Jennings, N. R.; and Shadbolt, N. R. 2006. Certified reputation: How an agent can trust a stranger. In *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '06, 1217–1224. New York, NY, USA: ACM.

Luke, S.; Panait, L.; Balan, G.; and Et. 2014. Ecj 21: A java-based evolutionary computation research system.

Marsh, S. P. 1994. *Formalising Trust as a Computational Concept*. Ph.D. Dissertation, University of Stirling.

Mui, L.; Mohtashemi, M.; and Halberstadt, A. 2002. A computational model of trust and reputation for e-businesses. In *Proceedings of the 35th Annual Hawaii International Conference on System Sciences*, volume 7 of *HICSS '02*, 188–. Washington, DC, USA: IEEE Computer Society.

Sabater, J., and Sierra, C. 2001. Regret: Reputation in gregarious societies. In *Proceedings of the Fifth International Conference on Autonomous Agents*, AGENTS '01, 194–195. New York, NY, USA: ACM.

Sato, D.; Borges, A.; Leite, A.; Dordal, O.; Avila, B.; Enembreck, F.; and Scalabrin, E. 2012. Lessons learned from a simulated environment for trains conduction. In *Industrial Technology, 2012 IEEE International Conference on*, 533–538.

2009. Testing and approval of railway vehicles from the point of view of their dynamic behaviour - safety - track fatigue - ride quality.

Voyiatzis, A. 2012. A survey of delay- and disruption-tolerant networking applications. *Journal of Internet Engineering* 5(1):331–344.

Wooldridge, M., and Jennings, N. R. 1995. Intelligent agents: Theory and practice. *Knowledge Engineering Review* 10(2):115–152.