UNIVERSITY OF SOUTHAMPTON

Trust and Reputation in Open Multi-Agent Systems

by

Trung Dong Huynh

A thesis submitted in partial fulfillment for the
degree of Doctor of Philosophy

in the
Faculty of Engineering and Applied Science
School of Electronics and Computer Science

June 2006

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND APPLIED SCIENCE
SCHOOL OF ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

by Trung Dong Huynh

Trust and reputation are central to effective interactions in open multi-agent systems (MAS) in which agents, that are owned by a variety of stakeholders, continuously enter and leave the system. This openness means existing trust and reputation models cannot readily be used since their performance suffers when there are various (unforseen) changes in the environment. To this end, this thesis develops and evaluates FIRE, a trust and reputation model that enables autonomous agents in open MAS to evaluate the trustworthiness of their peers and to select good partners for interactions. FIRE integrates four sources of trust information under the same framework in order to provide a comprehensive assessment of an agent's likely performance in open systems. Specifically, FIRE incorporates interaction trust, role-based trust, witness reputation, and certified reputation, that models trust resulting from direct experiences, role-based relationships, witness reports, and third-party references, respectively, to provide trust metrics in most circumstances. A novel model of reporter credibility has also been integrated to enable FIRE to effectively deal with inaccurate reports (from witnesses and referees). Finally, adaptive techniques have been introduced, which make use of the information gained from monitoring the environment, to dynamically adjust a number of FIRE's parameters according to the actual situation an agent finds itself in. In all cases, a systematic empirical analysis is undertaken to evaluate the effectiveness of FIRE in terms of the agent's performance.

Contents

List of Figures

List of Tables

Acknowledgements

The completion of this Ph.D thesis is the reflection of a long journey of learning and researching, during which much great help has been sought after. I would, therefore, like to take this opportunity to express my gratitude to those who have given me help and support to make this project possible.

First and for most of it is my supervisors, Professor Nick Jennings and Professor Nigel Shadbolt, who have been guiding and helping me for all the time the project was carried out. I am, therefore, greatly indebted to them for their knowledge and encouragement, which have been one of the main ingredients in making this work a success.

Second, I am grateful to the School of Electronics and Computer Science, University of Southampton for providing me the studentship, to the AKT project (Advanced Knowledge Technologies Interdisciplinary Research Collaboration, which is sponsored by the UK Engineering and Physical Sciences Research Council under grant number GR/N15764/01) and to Professor Jennings for funding my research. This financial support has allowed me to carry out my work and realise my goal of becoming a researcher in the field I love, Computer Science.

Third, I would like to thank Dr Duong Anh Duc, Professor Hoang Kiem, Mr Tran Duc Huyen, and Mr Nguyen Thanh Hung for having introduced and guided me into the field of Computer Science and for their continuous encouragement throughout my academic endeavours.

I have benefited a great deal from the discussions with and the collaborations from my colleagues in the Intelligence, Agents, Multimedia Group, to name a few, Dr. Sarvapali Ramchurn, Jigar Patel, Luke Teacy, Dr. Partha Dutta, and Dr. Raj Dash. Working with them always made me realise how fortunate I am to be with such good people.

This work has also been possible thanks to the many friends I have, in Southampton and elsewhere, who have always been a source of support and helped me enjoy the little free time this PhD allowed me, so I owe them a big thank you.

Last, but not least, I am most grateful to my mom, my wife, and my brother for their blessings, love, affection, and interest in my work. They have been the ultimate motivations behind this journey. Without them all successes of my endeavours would have been difficult to envisage.

To Mom, Giang, and Duy.

Chapter 1

Introduction

Trust is pervasive in human societies. Everyday, from the moment a person wakes up, trust plays an important role in each of his actions. For example, when his doorbell rings in the morning, he can open the door to an unknown postman. He trusts that the school bus driver will take his children safely to their school. Moreover, he trusts not only in people, but also in inanimate objects, systems, and institutions. For instance, when he turns the tap on, he expects the water to flow and it is of the required drinking standards. He trusts that his paper money is exchangeable for goods and services whenever he needs them. Without the trust he places routinely everyday, his life would be unbearable. For a company or an organisation, trust is of no less importance. Every company trusts in the legal systems when carrying out all of its (legal) business transactions. It also trusts that their employees and their partners will not betray them if they are offered a chance. From these examples, it can be seen that the existence of trust helps humans and organisations be confident about the behaviour of those they rely on. In short, trust is essential for any decision which makes an entity dependent on any other one.

Until recently, trust was viewed as a concept that is applicable for human beings only. In computer science, the word 'trusted' was used mostly in the area of security and was usually associated with the meaning of 'known to be safe' [Abrams and Joyce, 1995; Jøsang et al., 2006]. This is, however, a very limited view of the concept of 'trust' in the real world. In contrast, work in the area of agent-based computing (amongst others) has made trust a relevant research topic for computer scientists [Castelfranchi and Tan, 2001; Ramchurn et al., 2004]. The reason is that computer software agents with their emphasis on autonomous actions and flexible

interactions are now expected to exhibit behaviours that are more akin to those found in human societies than has hitherto been the case in computer systems. For example, a wide range of agents have been developed to conduct business in electronic environments such as the Internet (see [He et al., 2003] for a review). In such situations an agent can participate in online auctions (e.g. monitoring bids, and making bids), or they can negotiate on behalf of their owners (e.g. negotiating for the best price possible, or making business commitments such as payments and contracts). Having such levels of delegation, agents are also able to make certain decisions themselves, including decisions not to uphold their commitments. In such scenarios, the risks of traditional commerce come to the fore (e.g. fraud, unfulfilled commitments, and services/products of low quality). Moreover, the risks may even be intensified by the speed and the reach of the new technologies. Therefore, trust, which is essential in human social relations, needs to be re-created and maintained in new forms of computer supported collaboration and computer (agent) mediated communities.

In general, the term 'trust' can have various meanings depending on the context, as well as the trusting and trusted parties (see [Dasgupta, 2000] for a review). However, for the interacting entities in the context of agent communities, trust can be understood as the expectation or the belief that a party will act benignly and cooperatively with the trusting party [Dasgupta, 2000; Gambetta, 2000a]. Evaluating this expectation before making interactions is important because it can help an agent to estimate the trustworthiness of each potential partner and thus to decide whether the partner is reliable enough to interact with. Therefore, the existence of a trust measure in agent communities provides a mechanism to help agents identify reliable partners and avoid potential risks resulting from interactions with less reliable ones.

Since the first attempt of computationally modelling trust for agents by Marsh [1994], there has been a significant amount of research on trust for various computer environments and applications (see Sections 2.2 and 2.3 for more details). In recent years, however, open multi-agents systems (MAS), with their distributed nature, independent entities having rich reasoning capabilities, and their standardised communication infrastructure (see Section 1.1), have emerged as a natural model for computer communities. There are already a wide range of computer communities that are modelled as open MAS. Well-known examples are the Grid [Foster et al., 2001], the Semantic Web [Berners-Lee et al., 2001], Virtual Organisations (VO) [Norman et al., 2004], the CoABS Grid [Kahn and Cicalese, 2002], the Open Agent Architecture [Cheyer and Martin, 2001; Martin et al., 1999], var-

ious electronic commerce environments (see reviews in [Guttman et al., 1998] and [He et al., 2003]), and Peer-to-Peer sharing networks such as Gnutella, MFTP (eDonkey2000), and FastTrack (Kazaa). However, there has not been any trust model designed specifically for this type of computer community and models that have been developed for other contexts are not readily adaptable (see Chapter 2). Given the increasing ubiquity of open MAS, a trust model devised for them will benefit a wide range of agent applications that need a means to assess the trustworthiness of agents to operate effectively. Against this background, this thesis presents just such a model — FIRE[1] an integrated trust and reputation model for agents in open MAS.

1.1 Agents and Multi-Agent Systems

Before delving into a further discussion on trust, we first identify the main building blocks of open multi-agent systems. To this end, this section introduces the basic concepts of agency and multi-agent systems, which will be used throughout this thesis. First we consider the notion of agency.

> An *agent* is an encapsulated computer system situated in some environment that is capable of flexible, autonomous action in that environment in order to meet its design objectives [Jennings, 2001].

From this definition, there are a number of properties of agents that require elaboration. Agents are [Jennings, 2001]:

- clearly identifiable problem-solving entities with well-defined boundaries and interfaces,
- situated (embedded) in a particular environment over which they have partial control and observability — they receive inputs related to the state of their environment through sensors and they act on the environment through effectors,
- designed to fulfill a specific role — they have particular objectives to achieve,

[1] FIRE is from '**fi**des' (Latin for 'trust') and '**re**putation'. In the Ramayana legend of India, Sita proved the purity of her character by passing through the raging fire flames.

- autonomous — they have control both over their internal state and over their own behaviour, and
- capable of exhibiting flexible problem-solving behaviour in pursuit of their design objectives — being both reactive (able to respond in a timely fashion to changes that occur in their environment) and proactive (able to opportunistically adopt goals and take the initiative).

In this research, all agents are additionally assumed to be rational; meaning *for each possible percept sequence, they should do whatever action is expected to maximise their performance measure, on the basis of the evidence provided by the percept sequence and whatever built-in knowledge the agent has* [Russell and Norvig, 1995]. Irrational agents have unpredictable behaviours, and thus it is impossible to have any expectation about their actions. Hence, irrational agents are placed outside the remit of this research.

When adopting the agent approach to problem solving, it soon becomes apparent that most problems require or involve multiple agents: to represent the decentralised nature of the problem, multiple loci of control, multiple perspectives or competing interests [Jennings, 2001]. Hence, a *multi-agent system* is one that is composed of multiple interacting agents that work together to solve problems that are beyond the individual capabilities or knowledge of each agent (adapted from Jennings et al., 1998). It should be noted from this definition that the agents in a MAS are designed to work together towards some common goals of the system. In contrast, an *open* MAS allows agents from various sources to join and operate. Thus, the agents in an open MAS may work towards different, or even contrary, goals because of different ownerships. In other words, an open MAS is an open society of agents, where they can operate and obtain some benefits from the infrastructure and other agents in the society. In general, the two main features of an open MAS are:

1. Agents can freely join and leave at any time.
2. Agents are owned by various stakeholders with different aims and objectives.

From these two main features, other characteristics of an open MAS can be derived as follows (adapted from [Barber and Kim, 2002]):

- The environment in an open MAS is dynamic: Agents providing services might become unavailable and new agents offering new services might come

online. This means that the environment will change over time as the system operates.

- The number of agents is unbounded, given that agents can join an open MAS at any time.
- An open MAS is insecure: There may be incompetent, unreliable or even malicious agents in the system.
- No agent can know everything about its environment: It is not practical for agents to rely on access to complete information about the environment that they are in. Given the typical scale of an open MAS, the computational cost required to create such a world view will exceed any performance improvements (if it is even possible).
- Due to different ownerships in an open MAS, there is no central authority that can control all the agents. In addition, it is assumed that agents are self-interested.

Against this background, in order to show the importance of trust in open multi-agent systems, the next section presents a scenario for the domain of electronic commerce in which the various occurrences of trust are identified and their roles are analysed.

1.2 An Example Trust Scenario

The scenario in this section is an example of future online transactions that are mediated by agents. In particular, this scenario outlines the interactions that may be involved in a car purchasing transaction. In the scenario, James intends to buy a new BMW 760i to replace his old car. He assigns his personal (software) agent to find a suitable deal on the Internet. The parts where trust is involved are emphasised in italics.

1. James *instructs his personal agent to look for a good car retailer* in the region. He also tells the agent the desired model and his available funds.
2. James' agent *contacts an online directory service* to fetch a list of local car retailers.

3. James' agent then contacts James friends' personal agents to *ask them to rate* the service of those in the retailer list that they have had experiences with. It collects the ratings, aggregates them, giving high priorities to ratings from those with high experience in cars, then filters the original list into a list of potentially good retailers.

4. James' agent begins to contact these retailers, asking them about their offers on the specific model 760i of BMW, and at the same time asks them *to provide 'authorised dealer' certificates* from BMW UK.

5. A problem arises when no offer falls within range of the available funds. James' agent suggests some cheaper models to him, and also makes him aware of the fact that the retailer X offers finance options. James still insists on a BMW 760i and *instructs his agent to negotiate* with X on a particular finance option.

6. In order to consider the finance option request from James' agent, the agent of retailer X *contacts an established credit reference agency to obtain the credit history* of James. After having assessed James' financial status against its finance policy, the agent of retailer X agrees to sell the car with the requested finance option.

7. James' agent returns to him with a purchasing contract. He checks the terms and *signs it digitally.* The signed contract is then presented to the retailer's agent by James' agent along with *an electronic payment* as the deposit. The two agents negotiate to set an appointment when James can come to collect his new car. James' agent also records in its diary future payments according to the finance terms agreed.

The model of an open MAS (as described in Section 1.1) fits well this scenario since the participating agents here are owned by various stakeholders (e.g. personal agents, retailers' agents, and the credit agency's agents) and they all have their own aims and objectives (e.g. selling cars for profits, buying the required car at a good price with the limited available funds). To James' agent, the environment in the scenario is dynamic and uncertain because it cannot be sure whether the other agents are reliable. For example, a car retailer might sell illegally imported products which do not qualify for the manufacturer's guarantee; or a credit agency can be incompetent and might produce unreliable credit reports. Against all these uncertainties, thanks to the various trust relationships, a transaction can still be carried out. As the highlighted text shows, trust appears in every decision taken

in the transaction (those of James, his agent, and the retailer X). Its occurrences in the scenario can be generally classified into four main categories:

- *Trust in information sources*: This covers trust in information credibility, as well as the sufficiency of the sources. For example, in step 2, when asking for a list of local car retailers, James' agent trusts that the directory service it contacts has a list of car retailers with correct and sufficient information allowing it to proceed to later steps.

- *Trust between agents*: This covers the expectation that another agent will have desirable behaviours. That is the belief that when offered a chance, the other agent is not likely to behave in a way that is damaging to the trusting agent. This is the kind of trust that individuals in a society have in their family, friends, and close partners. In step 3, James' agent trusts agents belonging to James' friends and believes that they will provide their ratings honestly. Trust between agents is usually built on the relationship between the two agents and evolves along with the development of their relationship.

  It should be noted that when choosing which car retailer to contact, James' agent depends on the honesty of agents of James' friends. Consider the case that an agent W receives commission from introducing customers to the retailer X. Since agents are assumed to be rational and self-interested, agent W can report falsely about retailer X's performance despite the trust of James' agent. However, if James' agent knew about the relationship between the retailer X and agent W it would treat the ratings from W with respect to the retailer X with doubt and care. Therefore, when trust is built on information obtained from others, the possibility of lying or false information should always be taken into account.

- *Trust in the internal characteristics of an agent*: This covers understanding the capabilities of an agent and its interests in carrying out a delegated task. When James instructs his agents to find a potential car retailer (step 1) or to negotiate a finance option (step 5), he believes that his agent is sufficiently competent to deal with these tasks. More importantly, he believes his agent will do its best to serve his interests (rather than those of the retailer X). In other words, he knows his agent's characteristics and trusts it in those specific tasks providing these characteristics. The same applies to the retailer X when it delegates its agent to negotiate and settle deals.

- *Trust in the environment*: Here the environment consists of all the external

conditions that affect the agents' operations (e.g. the environment's infrastructure, rules, and so on). Knowing about the external conditions that can make an action successful or reduce the probability or the damage of its failure helps an agent be more confident when taking that action. This knowledge helps an agent to proceed when facing a potential risk of loss. In the above scenario there are various types of external conditions that affect the agents' decisions. These include:

- Technology: The digital certificate presented by X's agent (step 4) assures James' agent about the origin of its 'authorised dealer' certificate; or the digital signature presented by James' agent (step 7) assures the retailer's agent about the validity of the contract signed by James.
- Institutions of electronic commerce: Both agents and their owners (i.e. James and the retailer X) trust the digital contract and the electronic deposit payment (step 7). They believe that both parties will obey the agreed terms and, therefore, they trust each other. They are also confident that if a party breaks the contract it will be punished by an authority and the resulting loss will be compensated for.
- Norms in a society: When consulting a credit reference agency (step 6), X's agent expects that it will receive complete and impartial information because it believes the norm that the agency will do that to uphold the reputation of its service.[2]

It should be noted that the trust behaviours in this scenario are very close to those in human societies (as discussed at the beginning of this chapter). For example, in step 3, James' agent gives more attention to the ratings of those who know a lot about cars. It is also the case that trust research usually attempts to model trust in a manner that is as close as possible to some particular trust relationships in human societies. This is due to the belief that trust in human societies is essential and effective in enhancing relationships and promoting cooperation among individuals [Gambetta, 2000a], and, thus, should be replicated. Now, having analysed the trust scenario and identified the main trust categories, in the next section, we turn to determining the specific goals for our research in building a trust model for application in open MAS.

[2]The belief also involves trust in the motives (internal characteristics) of the credit reference agency that it wants to keep up its reputation to attract more customers. This will, in turn, increase its profits. This belief can be viewed as a norm in business.

1.3 Research Goals

As we can see in the scenario of Section 1.2, trust exists and plays an important role in many of the decisions of the agents and their owners. The existence of these various types of trust allows the transaction to happen despite the potential risks that exist in that scenario (e.g. being lied to or being deceived by the other party). Against this background, we believe it is important to have computational models of trust and to bring this into the arena of agent-based systems. This belief has also led to a significant body of work on trust in recent years (see [Falcone et al., 2001, 2003; Jensen et al., 2004] for some examples and [Ramchurn et al., 2004] for reviews). However, because trust is so ubiquitous and comes in many forms, it is essential that the scope of this research is clearly defined.

To this end, the main goal of this research is to create a trust model for open MAS. Therefore, it will study trust relationships between software agents only. This means that trust between humans, or trust of humans in agents/systems is outside the scope of this research (but see [Gambetta, 2000b] for more details of these areas). Since delegation of tasks to other agents is the main means to achieve bigger and more complicated goals in MAS [Zambonelli et al., 2001], task delegation in open MAS is here chosen to be the context for trust evaluation. However, trust relationships between agents may fall in any of the four main categories of trust identified in Section 1.2. Hence, it is necessary to consider which types of trust will be studied and which will not.

Trust in internal characteristics of an agent is made up of two components: trust in the capabilities of an agent and trust in that agent's goals or interests. The latter depends on the capability of agents to build up sufficient knowledge about the mental states (i.e. goals and interests) of another agent. So far, only the works of Castelfranchi and Falcone [1998, 2001] have analysed this type of trust (see more details in Section 2.1). However, they did not show how the mental states of an agent can be discovered and modelled. In general, this task is particularly difficult in open MAS due to the diversity of agents (recall the characteristics of an open MAS in Section 1.1). Since agents can have very different tasks, domains and designs, it is almost impossible to develop a general method to model the mental states of every agent in an open MAS. Therefore, in this research, trust made up from the mental states of an agent will not be considered. The former, trust in the capabilities of an agent, is a prerequisite for the decision to delegate a task to an agent. In MAS, there are several different mechanisms that allow an agent to advertise its capabilities so that others can discover the services that it is

providing (e.g. Middle-Agents or Matchmakers in [Decker et al., 1997; Klusch and Sycara, 2001], and Service Directory Service in FIPA Architecture [FIPA, 2002]). However, there is still no mechanism for verifying that an agent can actually do what it has advertised. Since an agent must know a partner's capabilities before delegating any task, it is desirable that an agent has a way to verify the partner's capabilities. This motivates one of the aims of this research: to study how trust in the capabilities of a particular agent can be established.

Trust in the environment is a broad category. Since the conditions of the environment that affect an interaction vary depending on the nature of that interaction, it is impossible to provide a generic model for this type of trust. Some of the conditions have been studied well such as security, platforms, and so on (see [Grandison and Sloman, 2000] for examples), including some specifically for agents (e.g. auction protocols in [Brandt, 2002] and [Hsu and Soo, 2002]; and security in [Wong and Sycara, 2000]). However, these conditions are chosen by the agents' designers, not by the agents themselves. Hence, agents trust the environment and its conditions because they are trusted by the agents' owners or designers. At the current time, agents have no capability to perceive the conditions imposed by the environment or to judge about its trustworthiness. These capabilities require the environment to have a standardised way of advertising the conditions in effect and their impacts. This is currently unavailable. Therefore, it is currently impossible to model trust in the environment for agents. This research therefore will ignore this type of trust and leave it open for future work.

Trust between agents: In the relationships between agents, knowing that a partner is able to carry out a task is usually not sufficient. Each agent has its own interests and will act according to those interests. It is also true that agents have degrees of freedom to disappoint others. Hence, the belief of an agent that a partner will do a delegated task in a desirable manner is a determinant factor when it considers task delegation to that partner. This belief is called *service provision trust*. In open MAS, it is possible that there are malicious agents trying to exploit naïve ones (those who believe blindly what others say). Thus, the risk of being deceived by an unknown agent is higher than in traditional MAS (where all the agents are designed to work together). Therefore, this type of trust is especially important in open MAS. Having a trust measure will help agents in open MAS to identify unreliable agents and to gain confidence when dealing with reliable ones. Hence, this research will study how *service provision trust* can be modelled in order to provide a trust measure for agents in open MAS.

Trust in information sources. Information sources in an open MAS can always be treated as agents providing an information service. Then, trust in information sources means trust in those information providing agents, and, thus falls into the category of trust between agents discussed above. However, there are a number of traditional information sources that operate as information databases without agent-like behaviours (e.g. contact/email directories, image databases, map/weather services). Although it is still possible to view them as very simple agents, there are criteria that are more relevant in evaluating the quality of these information sources than their general trustworthiness. Those criteria include information credibility, information provenance, or the correctness and sufficiency of information provided (see more in [Barber and Kim, 2002; Hertzum et al., 2002; McGuinness and da Silva, 2003]). Generally speaking, they are of a different area to that of trust in the target of study (i.e. information sources), and, thus, will not be covered in this research. Whenever information sources are treated as agents, service provision trust will be used.

In summary then, the goals of this research are:

1. To model trust relationships between agents in open MAS. This covers service provision trust and trust in the capabilities of an agent.
2. To provide a trust measure for agents in open MAS that helps them to identify reliable partners. This includes providing mechanisms to build and to maintain trust among agents.

In particular, we envisage that our trust model should satisfy the following requirements (here listed as R1, R2,... and subsequently refined to R1a, R1b, etc. in Chapter 2):

- **R1**: Be able to provide a trust measure in all situations that an agent may be in by making use of a variety of potential sources of information that can be used to derive the trustworthiness of a partner.
- **R2**: Be suitable for open MAS given their characteristics as discussed in Section 1.1. This includes the ability to cope with the distributed and 'no central authority' nature of an open MAS, the possible large number of agents that may be present, as well as the dynamic nature of the environment in an open MAS (e.g. agents come and leave, change their relationships with others, and/or change their behaviours).

- **R3**: Be adaptable to different domains of applications.
- **R4**: Be robust against possible cheating and defecting (i.e. lying or false information).

These desiderata for our trust model are general and at a high level. More specific requirements for the trust model will be derived from an analysis of the approaches to modelling trust in Chapter 2.

1.4 Research Contributions

By accomplishing the objectives set out in the previous section, this research advances the state of the art in the following ways:

- Although trust has been investigated in a significant amount of research, trust for agents in open MAS has not been explicitly addressed within the field of MAS to date. Most work in the area has tackled the problem of modelling trust in very specific or narrow contexts. This research, however, studies trust in the more general context of an open MAS (see Section 1.1), which has been used as a model for a wide variety of agent applications. FIRE's trust mechanisms are then built in a generic way such that they do not depend on any application-specific information to operate effectively (as those of most existing trust models do). Therefore, it can enjoy a much wider applicability than the current trust models in an open MAS.
- This research extends the current work on trust based on an agent's direct experiences, its relationships, and witness reports by adapting them to the context of open MAS and by integrating them into a coherent framework. Being more precise, this research devises new normalised reliability measures for these types of trust, allowing them to be aggregated into a single trust measure, but still taking into account their individual situations using configurable parameters. Adopting this framework means agent designers have the ability to evaluate trust from various perspectives (i.e. using various sources of trust information). It also allows them to adjust the trust model to suit their domain of application by changing various parameters of FIRE (e.g. the influence of each type of trust on the overall trust measure, and the sensitivity to the recency of ratings of each component of FIRE, and the

level of confidence based on the set of ratings taken into account). No such trust framework has been developed in the current literature.

- This research formalises the notion of third-party references into a type of trust information that can be use to derive the trustworthiness of an agent. This allows an agent to actively present the references about its past performance to potential interaction partners in order to establish trust relationships with them. This type of trust is here called *Certified Reputation*. By using references, an agent can prove its capabilities to other agents (to gain the trust in its capabilities), while the other agents do not have to look for relevant trust information themselves. Certified Reputation is highly available since agents can typically collect a large number of references themselves and they are incentivised to present these to establish new trust relationships.

  The idea of Certified Reputation has not been developed in other work on trust. However, there are a few cases where somewhat similar concepts are presented such as trust policy management engines (e.g. PolicyMaker [Grandison and Sloman, 2000], Trust-Serv [Skogsrud et al., 2003]) — which grant rights to an agent based on its self-presented certificates of its identity according to predefined policies or endorsements [Maximilien and Singh, 2002] — certificates endorsing that a service (provider) is trusted and preferred by their issuers. Nevertheless, they address a different problem in the case of trust policy management engines (i.e. granting rights in the security area), and the real benefits of using endorsements have not been fully demonstrated.

- Third-party information (used for deriving an agent's reputation) is typically prone to inaccuracy. To this end, this research develops a novel credibility model that allows FIRE to assess the reliability of information providers (i.e. reporters) and to weight, or to filter out, their information accordingly. More specifically, using our credibility model, an agent rates the credibility of a reporter based on the difference between the reports it receives and the *actual* interaction result it observes later. Hence, reporters' credibility is not objectively assessed based on how honest they are in revealing the interaction result they received, but rather it is subjectively judged based on their capability to give reports close to the actual results that a particular agent would receive. By so doing, an agent can detect not only inaccurate/false reports, but also honest, but useless, reports that result from the different views of the reporters. For example, one reporter may receive preferential treatment from a particular service provider and give out good (and honest)

ratings about this provider. Such ratings, though honest and accurate (in the view of that reporter), are not useful for other agents because they would receive only normal treatment from that provider. By taking an agent's individual situation (i.e. the actual performance it receives during interactions) into account, our credibility model can deal with cases similar to the one in this example appropriately. Hence, our credibility model is better suited for applications in open MAS than the existing solutions in that it takes the individual view of an agent into account and, more importantly, that it does not require additional application domain knowledge to work.

- This research shows how learning about an agent's environment can be incorporated into the agent's reasoning model to adapt FIRE (by adjusting its various parameters) to the current situation of a changing environment in a flexible manner. No such work has been done before for any trust model.

In terms of publications, the following contributions has been made:

1. Sarvapali D. Ramchurn, T. Dong Huynh, and Nicholas R. Jennings. Trust in multi-agent systems. *The Knowledge Engineering Review*, 2004, provides a survey of the current work on trust in agent systems.

2. T. Dong Huynh, Nicholas R. Jennings, and Nigel R. Shadbolt. FIRE: An integrated trust and reputation model for open multi-agent systems. In *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI)*, 2004, presents FIRE and the idea of certified reputation.

3. T. Dong Huynh, Nicholas R. Jennings, and Nigel R. Shadbolt. Developing an integrated trust and reputation model for open multi-agent systems. In *Proceedings of the 7th Int Workshop on Trust in Agent Societies*, 2004, extends the results in the previous paper further more by showing that FIRE also performs well in dynamic environments.

4. T. Dong Huynh, Nicholas R. Jennings, and Nigel R. Shadbolt. On Handling Inaccurate Witness Reports. In *Proceedings of the 8th Int Workshop on Trust in Agent Societies*, 2005, presents our model of credibility that can be used to detect and filtered out inaccurate witness reports.

5. T. Dong Huynh, Nicholas R. Jennings, and Nigel R. Shadbolt. An Integrated Trust and Reputation Model for Open Multi-Agent Systems. In *Journal of Autonomous Agents and Multi-Agent Systems*, 2006, presents in detail how

FIRE is constructed and why it is constructed that way with a comprehensive evaluation on its performance.

6. T. Dong Huynh, Nicholas R. Jennings, and Nigel R. Shadbolt. Certified Reputation: How an Agent Can Trust a Stranger. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multi-Agent Systems*, 2006, presents Certified Reputation as an independent trust model and shows how it deals with inaccurate references.

1.5 Thesis Structure

This thesis presents the FIRE model, designed to realise the objectives outlined in Section 1.3. The remainder of the thesis is structured as follows:

- **Chapter 2** gives a discussion on the current approaches to modelling trust in agent systems. It also reviews the current state of the art in the area and summarises the open issues.
- **Chapter 3** presents FIRE — the trust model devised in this research for agents in open MAS.
- **Chapter 4** describes the methodology and the test domain characterising an open MAS which will be used for evaluating FIRE.
- **Chapter 5** shows an empirical evaluation of FIRE's performance in the test domain. Particular attention is given to determining the contribution of each of FIRE's components to its overall performance and also to evaluating its performance in situations where various changes in an open MAS take place.
- **Chapter 6** extends FIRE to deal with situations in which witnesses/referees produce inaccurate reports about the behaviour of agents (either because they have a different perspective or because they seek to gain a strategic advantage by so doing). It also includes a detailed evaluation of the credibility model's performance in handling inaccurate witnesses and referees.
- **Chapter 7** implements a number of learning techniques in order for FIRE to adapt its parameters to suit the environment in which it is operating and shows how these techniques improve FIRE's adaptivity through empirical evaluations.
- **Chapter 8** concludes the thesis and outlines the directions for future work.

Chapter 2

Trust and Reputation in Agent Systems

This chapter introduces the concepts of trust, direct trust and reputation in agent systems. It starts with a discussion about the definitions and a classification of these concepts in Section 2.1, followed by a review on the main approaches to modelling direct trust (Section 2.2) and reputation (Section 2.3) in agents. The two sections break the task of modelling direct trust and that of reputation into subproblems and review the current approaches to solve each subproblem. Then Section 2.4 presents generic problems (which have not been discussed in the two previous sections) that may influence the effectiveness of a trust model in open MAS. Finally, Section 2.5 concludes this chapter by pointing out open issues that needs to be addressed and the specific requirements for the trust model in this research.

2.1 Trust Paradigms and Classification

To date there has been little consensus in the literature on exactly what trust is, although its pervasive importance has been recognised. Traditionally, there are two main views of trust. First, the cognitive view [Castelfranchi and Falcone, 2001] models trust as made up of underlying beliefs. That is, trust is a function of the values of these beliefs. Second, the probabilistic view ignores the role of underlying beliefs and uses a (scalar) metric to model a subjective probability with which an agent will perform a particular action [Yu and Singh, 2002]. Each of these views

will now be dealt with in turn.

The cognitive view of trust is mainly from the work of Castelfranchi and Falcone [1998, 2001]. The context they choose is that of task delegation where an agent a wishes to delegate a task to agent b. In so doing, agent a needs to evaluate the trust it can place in b by considering different beliefs it has about the motivations of b[1]. They claim the following beliefs of a are essential to determine the amount of trust to be put in b[2]:

- Competence belief: a should believe that b can actually do the task.
- Willingness belief: a believes that b has decided and intends to do what it has proposed to do.
- Persistence belief: a believes that b is stable enough about its intention to do what it has proposed to do.
- Motivation belief: a believes that b has some motive to help a, and that these motives will probably prevail over other motives negative to a in case of conflict.

In order to devise the level of trust that it can place in b, a would need to take all these beliefs into account. However, the evaluation of these beliefs requires modelling agent b's mental states. This task is generally complicated and imprecise in open MAS since there is no general way to model another agent's mental states given the great diversity of agents in both their origins and their domains. Therefore, we believe that the cognitive approach to modelling trust, although providing a natural view of trust from socio-psychological work, is not suitable for open MAS in general.

On the other hand, the probabilistic view ignores the beliefs about intentions of the other agent. In this approach, trust is quantified based mainly on agents' experiences (e.g. the outcomes of their interactions), which are observable to the involved agents. The main idea is that past experiences about an agent's behaviours can be used in predicting the future behaviours of that agent. In particular, they can be used for calculating the probability that the agent will show a particular behaviour. Obviously, an agent can always record past behaviours of

[1]From this point, we use a to denote the trust evaluating agent, and b the target agent being evaluated.

[2]The beliefs presented have been adapted and summarised from Castelfranchi and Falcone [1998, 2001].

those that it has interacted with. This ability allows agents to calculate trust values without the need of modelling mental states of other agents. Thus, in open MAS, this approach is more practical.

This research follows the probabilistic approach for the reasons noted above and uses the following definition (adapted from [Gambetta, 2000a]):

> Trust is a measurable level of the subjective probability with which an agent a assesses that another agent b will perform a particular action in a favourable way to a, both before a can monitor such action (or independently of its capacity ever to be able to monitor it) and in a context in which it affects its own action.

In this definition, the trust of a on b is a *subjective probability* because a can only have a limited view on b's behaviour; it cannot reason certainly what are b's next actions, but can only calculate the probability of b's possible actions based on its limited knowledge, which might not be true (hence subjective). *A particular action* is used in a deliberately broad sense to include any delegated tasks, including making payments, delivering goods, recommending other agents, and so on. *A favourable way to a* is also deliberately understood broadly to include honesty, security, safety, reliability, and timeliness. The *context* mentioned includes the external conditions of the environment such as the business context, the relevant agreements, the technology infrastructure, the legislative, and regulatory systems that may apply. This definition of trust is also agreed and used in various of the work on trust that adopt the probabilistic approach (e.g. [Abdul-Rahman and Hailes, 2000], [Dimitrakos and Bicarregui, 2001], [Mui et al., 2002], [Teacy et al., 2006], and [Yu and Singh, 2002]).

In a human society, the trust that an individual places on another can be built from two main sources:

1. Private information that it obtains from its direct relationships with the other.
2. Public reputation of the other, which can be obtained from other individuals in the society.

Similarly, trust in agent communities can also be built in the same manner. In order to build a trust measure an agent can take into account both of these sources

of information. A trust measure which is built on both direct relationships and reputation information is here called *composite trust*. The trust resulting from the direct relationships is now called *direct trust* in order to be distinguishable from the composite one. The direct trust that an agent a places in another agent b is derived from a's knowledge that ensues from evaluating its direct relationships with b. Therefore, direct trust reflects the subjective opinion of the judging agent (i.e. a). On the other hand, reputation is a collection of subjective opinions about an agent from other agents in the same society. It represents the view of the society about a member. Reputation of an agent is formally defined as follows (adapted from [Abdul-Rahman and Hailes, 2000]):

> The reputation of an agent is an expectation of its behaviours based on other agents' observations, or information, about the agent's past behaviours.

Since reputation information is from the subjective view of its provider, it is usually less reliable than information that an agent can observe and judge by itself (i.e. direct trust). Thus, the direct trust usually has a larger influence on the composite trust than reputation. However, in the case that an agent does not have enough information to calculate direct trust[3], it will have to depend on reputation information to evaluate trust. Since reputation information is obtained from other agents in the society and agents are free to lie, the reliability of reputation information should be taken into account.

In summary, a trust model should make use of both direct trust and reputation in order to be able to cover situations where one of them is unavailable or unreliable. In addition, making use of both brings more experiences into trust evaluations than using only one source of trust information. This should enhance a trust model's accuracy. Thus the trust model in this research will be built on both direct trust and reputation (recall the desideratum **R1**). These are then the two dimensions of trust that an agent can use to evaluate another agent in task delegation in this work. When combining the two dimensions of trust, the reliability of the trust measure in each dimension should be available in order to calculate the influence of each dimension on the final composite measure. The two next sections will discuss, in detail, about modelling these two dimensions of trust in agents.

[3]This is the case when two agents have not had any direct relationship or interaction. Thus, they do not know about each other directly. Or their relationship is too weak (e.g. too few interactions) to derive a reliable trust value.

2.2 Direct Trust

Direct trust, which only involves two agents, is calculated based on direct relationships between the two agents. This research focuses on two main types of such relationships:

1. Role-based relationships that stem from the social roles of the two agents (e.g. owned by the same organisation, relationships derived from links between the agents' owners in real life such as friendship or relatives, relationships between a service provider agent and its registered consumer agents), and

2. Relationships that result from direct interactions between two agents.

Relationships of the first type have not been studied much in modelling direct trust since there is no general way to computationally quantify trust based on them. The reason is that the number of social roles and the relationships between them may vary greatly depending on particular domains of application. Direct trust from role-based relationships, here called *role-based trust*, is thus left open to be defined by particular applications. The usual approach to this problem is using a rule-based (or policy) system to map relationships to trust values (see [Grandison and Sloman, 2000] for a comprehensive review). For example, an agent should have a high degree of trust in information provided by another agent that is from the same organisation, or a seller agent always has a tendency to increase the product price and to lower the product quality when possible[4]. These rules are specified for each agent and are used to map the role of an agent to a predefined trust value when evaluating role-based trust. Since the rule-based approach is fairly simple and adequate for modelling role-base trust, more effort is put on studying the latter. Thus, the remainder of this section focuses on reviewing the main approaches to modelling direct trust based on direct interactions, here called *interaction trust*.

Consider the context where an agent a wishes to delegate a task to another agent b and it is evaluating the trustworthiness of b to decide the task delegation. The most accessible source of information about b is past experiences of a from interactions with b, or a's interpretation of the results of past interactions with b. These observations reflect a's subjective view of b's behaviours and they can help a

[4]This rule reflects a norm in the relationship between a seller agent and an ordinary buyer agent.

predict the future behaviours of b. Therefore, this information is used to calculate the expected behaviours of b when it carries out the delegated task. However, there are two problems that need to be solved in modelling interaction trust:

1. How to represent interaction trust?
2. How to calculate the amount of trust from past interactions?

In most existing work, trust is represented as a single numerical value which shows the degree of expectation of agent a about a desirable action of agent b [Mui et al., 2002; Sabater and Sierra, 2001; Teacy et al., 2006; Yu and Singh, 2002; Zacharia and Maes, 2000]. The higher the trust value for agent b, the higher the expectation/probability that b will carry out that action in a's view. This also makes it easy to compare the trust values of two or more potential partners to select one from them. However, a number of models do use different representations. For example, the trust model of Abdul-Rahman and Hailes [2000] uses a set of ordered labels for trust degrees: 'Very Trustworthy', 'Trustworthy', 'Untrustworthy', 'Very Untrustworthy'. However, it is always possible to convert this type of representation back to an equivalent numerical representation for the ease of calculations and comparisons. Hence, numerical representation will be used for the trust model of this research.

With respect to the second problem, in order to calculate a trust value for b, trust models initially require agent a to gather its observations about b's behaviours. Without these observations, the interaction trust of a to b does not exist as a has no knowledge about b. For each interaction in a similar context to the task being considered, a gives ratings to the performance of b (i.e. a rates how good the result of each interaction is). This can be accomplished by comparing the outcome of a transaction (i.e. the quality of an action in the trust context being considered) against predefined criteria (e.g., b tells the truth or not [Schillo et al., 2000], whether b fulfills the contract with a [Teacy et al., 2006], or how good are the quality of the products purchased from b [Sabater and Sierra, 2001]). Then the ratings will be aggregated into a single value that shows the level of performance that a expects from b. This is the main idea of trust models in [Mui et al., 2002] and [Sabater and Sierra, 2001]. However, the contexts where trust is calculated in some work are very simple in that they consider the performance ratings of an agent to be simply a value of 'good' or 'bad', 'cooperate' or 'defect' (e.g. [Mui et al., 2002], [Schillo et al., 2000], [Teacy et al., 2006]), or a single performance measure (e.g. [Yu and Singh, 2002], [Zacharia and Maes, 2000]). These simple

rating systems limit the applicability of these models in real world situations since realistic interactions in an open MAS often involve various valuations (e.g. quality, timeliness, reliability). In contrast, Regret [Sabater and Sierra, 2001] gives a richer semantics to ratings, and, thus, it allows more complex trust contexts. For example, an agent can give a rating of -0.5 for late delivery of some good, and $+1$ for the quality of the same good. Moreover, Regret introduces the capability of rating on more abstract aspects by incorporating the ontological structures of these aspects into its model. For example, a rating of 'good seller' may be calculated from ratings about delivery date, price, and quality of the goods in the same interaction (see Figure 2.1). This favours Requirement **R3** of adaptivity in that this rating system can be reused for various types of agents from different domains (with different criteria in performance rating). This is particularly useful in an open MAS where there may be a great difference in the application domains and the designs of agents.

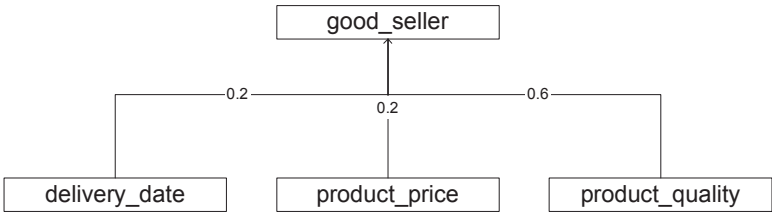

FIGURE 2.1: An example of ontological structure for ratings.

After having rated relevant observations, most models aggregate the ratings using some form of arithmetic mean function. In addition, to simulate the phenomenon that recent experiences affect trust more than older ones, Regret introduces recency as a weight in its mean function. It uses a time dependent function that gives higher values to the observations that are closer to the time that they are being considered. The function is then used as a weight value for the rating given for the corresponding observation. Thus, the recency of observations serves as the relevance factor of those observations[5] (see Appendix A.2 for more details). The aggregated value of ratings then becomes the value for the interaction trust. It should be noted that this trust value corresponds to the criterion that is used for rating. In binary rating systems (i.e. ratings have two values, such as 'cooperate/defect'), the trust value is the probability that an agent will perform the action that was rated (e.g. to cooperate or to defect). In single-criterion rating systems, the trust value is the expected performance of an agent with respect to the criterion being rated. In multiple-criterion rating systems (such as Regret),

[5]There are many psychological studies that support recency as a determinant factor [Karlins and Abelson, 1970].

each criterion will have a corresponding trust value which shows the expected performance in terms of that criterion in a future similar interaction. For example, in the context of buying a product, we will have the expected performance of an agent in terms of product price, delivery, and product quality in future selling actions of that agent. A notable class of trust models based on binary rating systems is those that calculate trust values using *probability density functions* (PDFs) [DeGroot and Schervish, 2002]. TRAVOS [Teacy et al., 2006] is such an example. Using binary ratings allows TRAVOS to make use of the beta family of PDFs to model the probability of having a successful interaction with a particular given agent. This probability is then used as that agent's trust value.

As discussed in Section 2.1, each trust measure should have a reliability measure in order to model its influence on the composite trust measure. For interaction trust, only Regret and TRAVOS introduce a reliability measure. Specifically, Regret uses the number of observations available and their deviation to determine the reliability of trust values calculated from those observations. The principles are: the more observations are available, the more reliable the resulting trust is until the number of observations exceeds a predefined threshold; and the greater the deviation in the observations, the less reliable the resulting trust. As for TRAVOS, given an acceptable margin of error and using PDFs, it calculates the probability the actual rate of successful interactions with b lies within the margin of error about b's trust value and this is then used as the reliability of b's trust values. Although PDFs provides a sound theoretical foundation for calculating trust values and their reliability, its dependence on binary rating systems significantly hampers its application in open MAS (because binary ratings are far more limited in terms of their expressiveness compared to the other types of ratings).

Comparing the models reviewed above, Regret provides the most complete framework for modelling interaction trust. It allows future adaptation to different domains (with a flexible rating system), as well as providing facilities for combining with other dimensions of trust (by providing a reliability measure). Hence, this research will not devise a new model for interaction trust, but will reuse the interaction trust part of Regret (the subjective reputation, as termed by Sabater and Sierra) with relatively minor adaptations (see Section 3.3).

2.3 Reputation

Reputation, as defined in Section 2.1, is built on observations about an agent's past behaviours. In order to build up a reputation measure, an agent needs to consult other agents in the society to collect their observations about the target agent (i.e. the agent whose reputation is currently being evaluated). Here, the observations are usually in the form of ratings, in which other agents in the society show how they value an agent after an interaction. These valuations, after being collected and properly aggregated, can be used to represent the reputation of other individuals [Sabater and Sierra, 2001]. Hence, the main tasks of a reputation model are to define how to collect observations about a specific agent from other agents in a society, and how to combine them to represent the reputation of that agent such that it is as close as possible to its actual trustworthiness.[6] It should provide a mechanism by which individual agents within their society can obtain information about other agents without, or prior to, direct interaction.

Reputation and interaction trust have a close relationship. The interaction trust that an agent places in another after an interaction is reflected by the corresponding ratings of that agent. Since reputation of an agent is built based on the observations (i.e. ratings) of other agents, it can be said that the reputation of that agent is built from the interaction trust it receives from other individuals in the society (provided it is being reported accurately). On the other hand, interaction trust between agents can be seen as reputation at the individual level (as is the case with Regret). In the case that two agents have no previous experiences with each other, thus they do not trust each other, reputation is a source of information that can be used to establish the initial trust between them.

Similar to modelling interaction trust, modelling reputation has the following basic problems that need to be addressed:

1. How to represent reputation?
2. How to gather observations about a specific agent?
3. How to aggregate the collected observations to represent the reputation of that agent?

[6] Because each member in a society has a particular point of view, each member may record a different rating from the same interaction. This means that each agent has a different perception of the reputation of a given entity and, therefore, that reputation is linked to subjectivity [Sabater and Sierra, 2001]. One of the goals of reputation models is to provide a reputation measure that models the real trustworthiness of an agent as closely as possible.

The last two questions are the main research foci when modelling reputation. Their issues will be presented in subsequent sections. Regarding the first question, most researchers tend to reuse the same representation as they used for trust (see discussion in Section 2.2) for reputation. This makes it easier to combine direct trust and reputation values later into a composite trust measure.

Since reputation is a social concept, besides the basic problems listed above, modelling reputation often needs to obtain information about the relationships between the involved agents. As there is no guarantee about the honesty of other agents in providing observations, information about relationships often helps an agent to evaluate the reliability of the observations that have been collected. Hence, the use of social relationships in modelling reputation will also be outlined and reviewed in Section 2.3.3.

2.3.1 Collecting observations

Observations about an agent's behaviours can only be obtained from those who have had direct interactions with it (i.e. witnesses). Depending on a particular model, an observation may be in one of the following forms:

1. Raw result of an interaction. This is the most basic and most useful form of observation since the receiving agent will be able to make its own ratings about the performance of the participating agents without being affected by the subjective view of the providing witness. However, in practice, it is not widely used because it is not suitable when the nature of the interaction is too complicated[7] or when the witness — one of the participants — has privacy concerns about disclosing raw results of its interactions.

2. Interaction trust value of the witness toward the target agent. This form of observation provides the view of the witness on how the other agent will perform. This is clearly subjective because other agents have no clue how the interaction trust value of the witness is derived. However, it is still useful in some cases when only simple opinions are needed. For example, in step 3 of the scenario in Section 1.2, James' agent only asks for the general opinions about some car retailers' service, not details of the interactions with them.

[7] That is the case when the raw results contain a large set of data that is not relevant for performance assessment; or other agents cannot make judgments from the provided results without knowing the context of the interaction.

3. Rating of an agent's performance in a particular interaction. This is the middle approach of the two forms of observations above. Though it is still from the subjective view of the witnesses, it provides more information about how the other agent performs in an interaction. For example, using the rating system of Regret, an agent can learn about various aspects of an interaction. In the example of Regret provided in Section 2.2, an agent can learn about how the other agent performs in terms of the price, the quality of product, and the delivery date of the same transaction. The witness can provide to the evaluating agent several ratings of different interactions with the target agent, rather than only one trust value as in the second form of observation.

Each form of observation suits a specific type of application, depending on the level of information required. However, in our opinion, the third form can suit a wider range of applications as it provides richer information for reputation calculation. Another reason in its favour is that it provides a level of abstraction over raw interaction results, overcoming the possible limitations of raw results regarding privacy and overly complicated data (as discussed in the first point above). Thus, given the diversity of agents in open MAS, we believe the third form is the most suitable for a general trust model (Requirement **R3**). In addition, the rating system of Regret, which is used for modelling interaction trust, can be reused for exchanging observations in this research.

The next question is how to find the right witnesses and collect their observations about the target agent. There are a number of approaches to this problem:

- *Centralised approach*: Observations are reported and then stored in a central database. The reputation system — usually the database itself — will use information in the observation database to calculate the reputation of an agent when asked. This approach is used in the reputation systems of online auction sites, such as eBay[8] and Amazon[9], and SPORAS [Zacharia and Maes, 2000]. These reputation systems offer a mechanism that allows their users to rate each other's general trustworthiness after a transaction and to report the ratings to the systems. The reputation of a user is then updated by the systems according to the new ratings. For example, eBay reputation is the sum of all ratings a user has received in the last 6 months. SPORAS extends the reputation model of eBay by introducing a new formula to calculate

[8]eBay site: `www.ebay.com`.
[9]Amazon site: `www.amazon.com`.

the amount of change of the reputation value according to the new rating value. It also devises a reliability measure for reputation values based on rating values' deviation (see Appendix A.1 for more details). However, the centralised approach is not suitable for open MAS as the agents are typically widely distributed. The cost of reporting ratings to a central database and asking reputation information from it might therefore become a problem as the number of agents become larger. Moreover, this approach assumes that the rating (reputation) system is accepted and trusted by all the individuals that join the system. This will not be the case in open MAS as there is no ultimate authority for all agents. Thus, agents from various sources may well question the trustworthiness of a reputation service and may not use it. Hence, this approach cannot fulfill Requirement **R2**.

- *Distributed approach*: Observations are stored locally at the agent who makes the observations. When an agent a wants to find out about the reputation of an agent b, it will look for agents that interacted with b (i.e. witnesses) then ask them for their observations about b. The searching process used is a distributed search through a's neighbours, forming chains or a graph of agents from a to b's witnesses. The distributed approach overcomes the main limitations of the centralised approach as they occur in distributed environments. Specifically, the task of calculating reputation is now carried out by each individual. This provides a level of freedom to the agents in choosing the method of calculating reputation which they believe will produce a reliable reputation measure. Besides collecting observations and calculating reputation, each agent also chooses the witnesses by itself. This provides more confidence for each agent on the resulting reputation value compared to the centralised approach. This approach is thus compatible with the open MAS's distributed and no central authority nature (Requirement **R2**). Given the increasing popularity of environments modelled as open MAS, most recent research on reputation adopts this approach for their reputation models (e.g. [Mui et al., 2002], [Sabater and Sierra, 2002], [Yu and Singh, 2002]). However, a basic degree of cooperation in locating witnesses and providing observations is still needed between agent a and other agents in the process of a distributed search.

  There is a variation of the distributed approach in which agent a only asks its neighbours or friends about their general trust evaluations about b (e.g. [Abdul-Rahman and Hailes, 2000]). Hence, there is no distributed search involved. However, observations are still stored locally at each agent. In

this approach, the agents provide their general views about b, which are not necessarily from direct interactions with it. In this case, the agents providing observations are sometimes called recommenders [Abdul-Rahman and Hailes, 2000]. The problem with this variation is that an agent's neighbours may not have the information about the target agent b. Given the possibly large number of agents in an open MAS, there is a high possibility that a may not be able to find any information about b if it does not perform a distributed search. Thus, this variation is not suitable for an open MAS.

- *Hybrid approach*: As its name suggests, this approach is both centralised and distributed in nature. More specifically, broker agents are used to provide reputation services. They centralise reputation information but are distributed in a system. For example, the trust model by Jurca and Faltings [2003] is one that follows this approach. It defines a set of broker agents (called R-agents) to buy and sell reputation information. There are no synchronisation requirements among different R-agents. Hence, some R-agents may possess more accurate information than others. Other agents have to contact an R-agent to buy any reputation information they need. Therefore, it is also necessary that they are equipped with an ability to learn and value the service of R-agents to avoid bad ones (which appears similar to interaction trust with R-agents). Although this approach is compatible with distributed environments, there is still no guarantee on the quality of service provided by R-agents. Other agents might suspect the objectiveness of R-agents and refuse their reputation service. Moreover, the number of agents in an open MAS may again cause a problem in finding R-agents that have reputation information about the target agent b. The two shortcomings mentioned make this approach unsuitable for open MAS.

The analysis of the approaches in this section has led to the following refined list of requirements which are derived from the requirement **R2**:

- **R2a**: Because of the no central authority nature of open MAS, in order to have the necessary degree of confidence in reputation values, each agent should collect observations and calculate reputation values for itself.
- **R2b**: The trust model should be scalable to the large number of agents that might be present in open MAS. The number of agents that the trust model can handle is preferably unlimited. This means that agents should

be equipped with a distributed search method that can locate witnesses effectively in possibly "large" agent societies.

For the above reasons, we will follow the distributed approach for collecting observations in open MAS. Among the models that adopt this approach, Regret and the model of Mui et al. assume that the network (graph) of agents from the judging agents to the witnesses is already available and just use this information in their models. Thus, they do not show how the information can be obtained. Yu and Singh [2003b] also follow this approach but they make an attempt to build a graph of agents (called the *referral network*) to locate witnesses based on the expertise of each agent in the graph. They develop a mechanism to locate information sources (i.e. witnesses) based on individual agents' knowledge and help (through each agent's contacts) without relying on a centralised service (see Sections 2.3.3 and 3.5 for more details). Hence, this approach is well suited for applications in an open MAS which is distributed by nature. Due to the diversity and the distributed nature of agents in open MAS, we believe that the task of locating witnesses should be treated with more attention as it is an essential part in modelling reputation. Therefore, we will survey the referral network introduced by Yu and Singh and its applicability in locating witnesses and collect observations in open MAS.

2.3.2 Aggregating observations

After collecting observations about an agent, the next step is to calculate its reputation from these observations. Because of the various forms of observation and the various ways of collecting them, there are a similarly large set of methods for aggregating the observations that have been collected. However, since the distributed approach (in the previous section) has been chosen for the reputation model of this research, we will limit our discussion to those methods that are used for the distributed approach.

The simplest way is averaging all the observations (e.g. [Schillo et al., 2000]), or better, averaging the observations weighted by the relevance/reliability of the observations' sources (e.g. [Abdul-Rahman and Hailes, 2000], [Mui et al., 2002], [Sabater and Sierra, 2002]). In [Abdul-Rahman and Hailes, 2000], for example, an agent assigns weights according to the trust it places upon the sources (recommenders). In [Mui et al., 2002], as each observation belongs to a chain of agents (see Figure 2.2) from agent a to a witness of agent b, Mui calculates the weight of an observation by taking the product of all the weights of each link in the chain.

The weight of a link between an agent x and y (denoted by w_{xy}) is defined as the

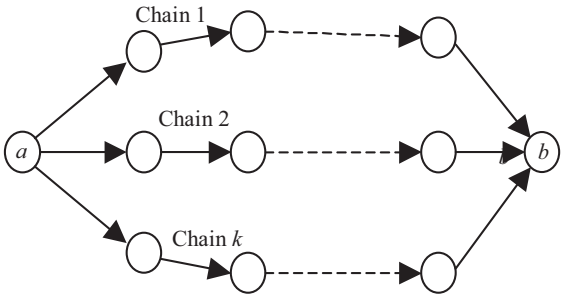

FIGURE 2.2: The chains of agents to witnesses in Mui's model.

reliability measure of the link where w_{xy} is equal to 1 if the number of encounters between x and y exceeds a defined threshold m. Otherwise w_{xy} is the proportion of that number to m. The Regret model takes a different approach using fuzzy rules to determine the weight based on the social relationship between b and the witness (w). For example, a possible rule would be: 'If the level of cooperation between w and b is high then the trustworthiness of the information coming from w related to b is very bad' because b may be able to affect w's ratings. This approach requires modelling the social relationships (see Section 2.3.3) of the agents involved in reputation calculations.

In the work presented above, weights are chosen to reflect the influence of each witness on the final reputation of the target agent. Obviously, the reliability (i.e. honesty, completeness) of observations is an important factor that affects the reliability of the aggregated reputation value. However, none of the methods of selecting weights above has been proved to be better than the others. Therefore, further empirical study is needed to evaluate the performance of each method. Intuitively, the method of Mui et al., which is based solely on the number of encounters, is too simple. In our opinion, the reliability of a witness should be based on additional factors, such as the witness's trustworthiness in providing observations, or the social relationships between the witness and the target agent (as suggested by Abdul-Rahman and Hailes and Sabater and Sierra respectively). Given its influence on the effectiveness of the reputation model being devised, we believe that the issue of selecting weights for the observations collected needs more careful study.

In contrast to other models, the model of Yu and Singh [2002] uses the Dempster-Shafer theory [Kyburg, 1987] to model trust. The main benefit of this theory is that it can model the case of uncertainty. Following Marsh [1994], their model defines, for each agent, an upper and a lower threshold for trust. When calcu-

lating the trustworthiness of b, performance ratings from direct interactions with b are categorised into trust, distrust, and uncertainty based on the two thresholds; and then are used to calculate the probabilities of the trust, distrust, and uncertain belief about b. These three probabilities, whose sum is 1, will be used as an observation about b (observations in the form of interaction trust — see Section 2.3.1). Such observations are collected from witnesses and combined with the current beliefs of a about b into a new set of beliefs of a using Dempster's rule of combination. Then the difference in trust and distrust beliefs about b is used as its reputation value. While Yu and Singh claim that this approach, which handles the case of uncertainty explicitly, is better than using a scalar value for an agent's belief, there is no apparent improvement in terms of the reliability of the reputation value calculated. Thus, in their paper [Yu and Singh, 2002], there is no evidence presented about the advantages of their method compared to others. In another aspect, their method is much more complex than those presented above; it also requires that observations be in the form of interacting trust, which provides less information than the two other forms of observations. Thus, we believe that there is no reason to use the Dempster-Shafer theory for aggregating observations.

In summary, we choose weighted mean of observations as our method of aggregating observations. However, focus should be placed on how the weights can be chosen to reflect the reliability of each observation.

2.3.3 Social relationships in modelling reputation

Since reputation is a social concept, social relationships are also one of the factors that affect an agent's reputation. For example, the fact that an agent belongs to a government office may imply high trustworthiness of the information that it provides. As seen above in Regret, the nature of the social relationship between agents can be used to determine the reliability of reported observations. In addition, being equipped with an effective method to model the social relationships between individuals, an agent can find witnesses quicker (as in the case of Yu and Singh's referral network described below). Hence, studying social relationships of agents is needed to make a reputation model more effective.

In recent work (e.g. [Mui et al., 2002], [Sabater and Sierra, 2002], [Yu and Singh, 2003b]), social relationships are usually captured in a graph where its nodes denote the agents and the edges denote the social relationships between them. Attributes can be labelled to the edges of the graph to represent the nature and/or properties

of the corresponding relationships (e.g. cooperation, competition). Such a graph is called a *social network* or *sociogram* [Sabater and Sierra, 2002]. As discussed in Section 2.2, there are two main types of relationships in open MAS: role-base relationships and relationships emerging from direct interactions between two agents. Some of the former are assigned to an agent by its owner or designer. They may reflect the initial roles, position, or membership of an agent in an organisation (e.g. company employee, or friend relationships). This type of relationship usually gives an agent an initial image of the society that it is about to join which, in turn, lets it know who it can trust initially. The latter type of relationship appears when an agent interacts with other agents in its society. Regular interactions between two agents can reflect a close relationship (e.g. a regular buyer). Relationships of this type can be selected to extend the initial social structure of role-based relationships. The more information an agent's social structure stores, the more the agent has learned about its environment, and the more useful it is when the agent comes to the questions such as who to trust (e.g. organisational relationships) and who to ask for recommendations (e.g. agents that have interacted with many others).

There are several researchers who have studied social relationships and how to use them in modelling trust. Sabater and Sierra use a social structure (called a sociogram) to identify witnesses in calculating reputation. They also use the information about the nature of the relationship between a witness and the agent being considered in the social structure to determine the possibility of lying, and thus, the reliability of the information provided from that witness. However, they assume that each agent already has a social network about other agents and they did not show how such social network can be obtained. Yu and Singh propose a method of representing a social network (based on their referral network) and provide techniques to gather information through the network. In more detail, in this system, agents cooperate by giving, pursuing, and evaluating referrals (a recommendation to contact another agent). Each agent in the system maintains a list of acquaintances (other agents that it knows) and their expertise. Thus, when looking for a certain piece of information, an agent can send the query to a number of its acquaintances who will try to answer the query if possible or, if they cannot, they will send back referrals pointing to other agents that they believe are likely to have the desired information (based on those agents' expertise). Yu and Singh's referral system uses a vector space model [Salton and McGill, 1983] to model agents' expertise. An agent's expertise is then used to determine how likely it is to have interaction with or to know witnesses of the target agent. The requesting agent also uses the referrals received to gradually build up a social

network and update its knowledge about the expertise of unknown agents.

2.4 Generic Issues of a Trust Model

Besides the basic issues of building an interaction trust measure and a reputation measure presented in the previous sections (Sections 2.2 and 2.3), there are a number of generic issues that can affect the performance of a trust model. These include: bootstrapping, dynamism in open MAS, inaccurate reports, and correlated evidence. They are going to be discussed in turn in subsequent subsections (Sections 2.4.1 to 2.4.4).

2.4.1 Bootstrapping

Newly joined agents who have no acquaintances will face various difficulties in joining the community. Typically, a new agent should have an initial set of contacts to establish its first interactions, as well as to collect reputation information about some initial potential partners. In addition, new agents may find themselves not accepted by some service providers because of their low initial reputation. However, this problem is ignored in most of the research in this area. We believe that solving the bootstrapping issue is necessary so that agents will be able to make use of a trust model in any situations (desideratum **R1**). Therefore, we add a new requirement for our trust model:

R1a: The trust model should be able to deal with the bootstrapping issue of newly joined agents.

2.4.2 Dynamism in open MAS

As discussed Section 1.1, due to its openness, the environment in an open MAS will change continually. Possible types of changes include:

- *The agent population.* Existing agents leave the environment and new ones join on a continual basis. This means that an agent might have to repeatedly learn about new agents since its previous interaction partners may no longer be available.

- *Agent behaviour.* Since agents are owned by different stakeholders, their goals and motivations may change over time. In addition, an agent's situation may also change. Now all of these will result in agents' changing their behaviours. For example, an honest agent may become a liar, or an agent may reduce its service quality due to less resources available to it.

- *Relationships between agents.* Agents may break old relationships and make new ones depending on their situations and needs. For example, virtual organisations can be automatically formed or disbanded according to the participants' capabilities and goals [Norman et al., 2004]. Hence, the relationships, and thus the trust, between them are also changed over time.

Given such a wide range of changes that can happen in an open MAS, a trust model for such an environment should reasonably maintain its normal effective operations under these types of changes. However, none of the existing trust models explicitly take such dynamism into account and none of them has been demonstrated to cope well with it. Therefore, in order for a trust model to be suitable for open MAS (Requirement R2), it should reasonably maintain its normal effective operations in situations where various changes in an open MAS take place (here called Requirement **R2c**).

2.4.3 Inaccurate reports

As agents in open MAS are self-interested, they may lie when being asked for their observations if they can gain some benefits from so doing (see [Schillo et al., 2000] for an example). In an attempt to solve this problem, the model of Schillo et al. shows how witness information can be reliably used to reason effectively against lying. However, the model greatly simplifies direct interactions (e.g. cooperate/defect in the disclosed Prisoner's Dilemma), thus, it is not useful in realistic settings.

To help overcome this problem, Jurca and Faltings [2003] presented a model in which agents pay for reputation information. When an agent needs to find reputation information, it contacts an R-agent to buy the information. Agents also receive money when reporting their observations to R-agents, but only after the verification of their reports. In this context, a mechanism is devised to determine the specific amount for each payment so that the agents that report truthfully will not lose money and agents that report falsely will lose money. Thus, lying agents

will gradually lose their money until they do not have enough to buy reputation information. This mechanism makes it rational for an agent to report its observations honestly. However, the idea of side payment may not be feasible in an open MAS. For example, in order to have the rational property mentioned above, the model of Jurca and Faltings requires that the currency for side payment is unexchangeable with the currency used in ordinary transactions. In open MAS, devising a new currency system that is different from the traditional ones, to be accepted by the agents from various origins is not practical.

Regret uses fuzzy rules to classify the reliability of the witnesses based on their relationships with the target agent (see Section 2.3.2 and Section 2.3.3). In this way, they also take into account the possibility of a witness lying based on fuzzy rules. However, this approach is a preventive measure and is based on social information, which is not always available in every situation. In our opinion, the trustworthiness of a witness in reporting its observations should also be taken into account. The reason for this is that the experiences with the witness (i.e. interaction trust) or the relationships between the witness and the collecting agent (i.e. role-based trust) are more reliable than social information.

In order to determine the accuracy of third-party ratings, Whitby et al. [2004] assume that the "true" rating of an agent is defined by the majority's opinions. In particular, they model the performance of an agent as a beta PDF which is aggregated from all witness ratings received. Then a witness is considered unreliable and filtered out when the reputation derived from its ratings is judged to be too different from the majority's (by comparing the reputation value with the PDF). Since this method bases it decisions entirely on PDFs of witness reports, if these reports are scarce and/or too diverse it will not be able to recognise lying witnesses. Moreover, it is possible that a witness can lie in a small proportion of their reports without being filtered out. To rectify this, TRAVOS provides a probabilistic method for filtering out the opinions of inaccurate reputation sources. Reputation is shared in the form of frequencies of successful and unsuccessful contracts that the reputation source has had with the trustee, which after interacting with the trustee itself, the truster compares with its own observations. By this means, the truster calculates the probability that the reputation source's information supports the true behaviour of the trustee within a reasonable margin of error, and uses this probability to weight the impact of the reputation source's opinions on future decisions made be the truster. However, TRAVOS's dependence on its binary rating system again is its weakness.

Yu and Singh propose a similar approach to that of Whitby et al. Specifically, they use a weighted majority algorithm to adjust the weight for each witness over time. Although the weights of the deceitful agents are reduced, these agents are never disregarded completely. Several successful applications of this approach have been demonstrated, but only for agent populations where deceitful agents are in the minority and are balanced between agents that falsely exaggerate their friends' performance and those that defame other agents.

In summary, all the proposed approaches above are limited in that they require additional domain knowledge or make unrealistic assumptions about the environment. In order to fulfill the Requirement **R4** of robustness, an effective mechanism is needed to deal with inaccuracy reports (here called Requirement **R4a**).

2.4.4 Correlated evidence

This problem happens when the opinions of different witnesses are based on the same event or when there is a considerable amount of information shared among a group of agents that make their opinions similar to each other. In both cases, the reliability of the information should not be as high as the number of similar opinions suggests [Sabater and Sierra, 2002]. Sabater and Sierra use graph analysis techniques to address this issue. The process starts with identifying graph components of a domain dependent sociogram. Then an agent in each component will be selected to be the representative agent for all agents in the component. Witnesses will be selected from those representative agents only. However, a node that deems to be representative for a component in a sociogram is not necessarily able to give a full witness' account for all the agents in the component and, therefore, choosing only one agents from those in a component may discard possible unique witness reports of the rest. Moreover, the approach is based on heuristics and there is no empirical result presented to show its capabilities. The problem of correlated evidence affects the efficiency and robustness of a reputation model and should be dealt with (here called the requirement **R4b**).

Requirements	
R1	The trust model should be able to provide an effective trust measure that can **R1a** deal with the bootstrapping issue of newly joined agents. **R1b** make use of role-based trust, interaction trust, and witness reputation when the required information for these dimensions of trust is available.
R2	The trust model should be suitable for open MAS. In particular, **R2a** each agent should be able to collect observations and calculate the reputation values by itself. **R2b** the trust model should be scalable to a large number of agents that might be present in open MAS. **R2c** the trust model should reasonably maintain its normal effective operation in situations where there are various changes in its environment.
R3	The trust model should be adaptable to different domains of applications that an open MAS may have.
R4	The trust model should be robust against **R4a** possible lying from agents. **R4b** the correlated evidence problem.

TABLE 2.1: The requirements for a trust model in open MAS.

2.5 Requirements for Trust and Reputation Systems in Open MAS

In the previous sections, it has been shown that a composite trust measure can be built from a number of trust measures: role-based trust, interaction trust, and reputation from witnesses (here called *witness reputation* from now on). However, the information required for each type of trust might not all be available at the same time. Therefore, we believe all three types of trust should be modelled so that an agent can have a trust measure in all situations (Requirement **R1**). This is specified in a new requirement:

R1b: The trust model should be able to make use of role-based trust, interaction trust, and reputation when the required information for these dimensions of trust is available.

Through reviewing the current trust/reputation model, we have identified a number of core issues in building a trust model for open MAS. In order to address these issues, our original requirements (Section 1.3) have been refined into more specific ones. These are summarised in Table 2.1. In order to provide an overview

of the trust models reviewed in this chapter, a comparison of them against our refined requirements is presented in Table 2.2.

The meanings of the symbols used in the Table 2.2 are as follows:

- Empty box: the model does not satisfy the corresponding requirement.
- −: the model attempts to solve the related problem(s) and has partly satisfied the corresponding requirements.
- +: the model satisfied the corresponding requirements.
- N/A: the corresponding requirement is not applicable.

Other notes for Table 2.2:

- The requirement **R1b** is split in the three dimensions of trust (i.e. role-based, interaction trust, and witness reputation). In order to satisfy this requirement in each dimension, a trust model has to implement the corresponding trust component and provide a reliability measure for the corresponding trust measure. The reliability measure is needed for combining that trust measure with the others.
- An empty box in the column of **R2b** means that the corresponding model provides no proof nor evidence about its scalability; and/or the number of agents that was run in the empirical study was small (e.g. 100 agents as in Yu and Singh [2002]).

As we can see from Table 2.2, Regret is the only model that provides an Interaction Trust component and satisfies the requirement about adaptivity in open MAS (**R3**). Therefore, the interaction trust component of Regret will be reused in FIRE with minor adaptations so that it can be fit with the other trust components. Hence, FIRE will also inherit the rating system with the rich semantics of Regret. It then allows adaptivity in the other trust dimensions, as well by offering the same rich semantics for sharing ratings and modelling complex trust contexts based on ontologies.

With respect to the witness reputation dimension, none of the models reviewed fully meets the requirements for a trust model in an open MAS. This analysis suggests the following issues that need to be addressed in this research:

- **R1a**: The bootstrapping issue.
- **R2a**: Most of the current witness reputation models show how agents can calculate a reputation measure from the observations collected from witnesses. However, it is not clear how agents can locate the right witnesses in a distributed and open environment (such as an open MAS) except for the model of Yu and Singh. Therefore, their the referral network (see Section 2.3.3) will be used in this work as the method to find the needed witnesses.
- **R2b**: The scalability of the trust model in an open MAS.
- **R2c**: The ability of the trust model to cope with the dynamism of an open MAS.
- **R4a** and **R4b**: The problems of lying and correlated evidence.

Against this background, we developed FIRE to address the remaining issues and to satisfy the requirements for a trust model in open MAS. The next chapter presents FIRE and discusses its various design decisions. In particular, it deals with Requirements **R1a**, **R2a**, **R2b**, **R2c**, and **R4b**. The problem of lying (Requirement **R4a**) is dealt with in Chapter 6.

Model	Role-based Trust (**R1b**)	Interaction Trust **R1b**	Witness Reputation			Generic Issues				
			R1b	**R4a**	**R4b**	**R1a**	**R2a**	**R2b**	**R2c**	**R3**
Model by Abdul-Rahman and Hailes [2000]		$-^a$	$-^b$							
eBay			+		N/A			+		$-^c$
SPORAS			+		N/A			+		$-^c$
Model by Jurca and Faltings [2003]			$-^b$	−			$-^d$	$-^e$		
Model by Mui et al. [2002]		+	+				$-^g$			
Regret	+	+	+	−	−		$-^g$			+
Model by Schillo et al. [2000]			+	$-^f$	$-^f$		$-^g$			
TRAVOS[i]		+	+	+			$-^g$			
Model by Yu and Singh [2002]		$-^a$	$+^h$				+			−

[a] There is no reliability measure.
[b] The model uses a general trust value for reputation instead of witness information.
[c] The model uses the centralised approach.
[d] The model uses the hybrid approach to collecting and aggregating observations. Agents contact an R-agents to sell reputation information by itself.
[e] Although there is no proof about the model's scalability, the model has been tested with 10000 agents.
[f] The model uses an overly simplified context that is not realistic.
[g] There is no method presented for locating witnesses.
[h] The model uses the Dempster-Shafer theory to combine the interaction trust and the witness reputation.
[i] TRAVOS is based on (and depends on) a binary rating system that is too limited for application in open MAS.

Table 2.2: The comparison of the trust models reviewed.

Chapter 3

The FIRE Model

This chapter formalises the basic FIRE model of trust and reputation developed in this research. This basic model is then subsequently extended in Chapter 6 to deal with situations in which witnesses produce inaccurate reports about the behaviour of agents (either because they have a different perspective or because they seek to gain a strategic advantage by so doing). Chapter 7 further extends FIRE by making a number of its parameters adaptive to various changes in the environment.

In more detail, this chapter is organised as follows. First, it discusses various sources of trust information and how they can complement one another in producing a comprehensive trust measure, especially when some of them can be missing or anomalous (Section 3.1). Then Section 3.2 shows how a trust value is calculated from a set of evidence (i.e. *ratings*). Sections 3.3 to 3.6 present, in turn, the four components of FIRE — Interaction Trust, Role-based Trust, Witness Reputation, and Certified Reputation. Section 3.7 shows how trust values produced by these components can be combined into a single overall measure. Finally, a summary of the chapter is provided in Section 3.8.

3.1 Sources of Trust Information

As can be seen in the previous chapter, trust can come from a number of information sources: direct experience, witness information, rules or policies. However, due to the openness of a MAS, the level of knowledge of an agent about its environment and its peers is likely to vary greatly during its life cycle. Therefore, at

a given time, some information sources may not be available, or adequate, for deducing trust. For example, the following situations may (independently) happen:

- An agent may never have interacted with a given target agent and, hence, its experience cannot be used to deduce how trustworthy/reliable the target agent is.
- An agent may not be able to locate any witness of the target agent (because of a lack of knowledge about the target agent's society) and, therefore, it cannot obtain witness information about that agent's behaviours.
- None of the current set of rules to determine the level of trust matches the role of this particular target agent.

In such scenarios, trust models that use only one source of information will fail to provide a trust value of the target agent. For that reason, FIRE adopts a broader base of information than has hitherto been used for providing trust-related information. Although the number of sources that provide trust-related information can vary greatly from application to application, we consider that most of them can be categorised into the four main sources as follows:

- *Direct experience*: The evaluator uses its previous experiences in interacting with the target agent to determine its trustworthiness. This type of trust is called *Interaction Trust*.
- *Witness information*: Assuming that agents are willing to share their direct experiences, the evaluator can collect the experiences of other agents that interacted with the target agent. Such information will be used to derive the trustworthiness of the target agent based on the views of its witnesses. Hence this type of trust is called *Witness Reputation*.
- *Role-based rules*: Besides an agent's past behaviours (which are used in the two previous types of trust), there are certain types of information that can be used to deduce trust. These can be the various relationships between the evaluator and the target agent or its knowledge about its domain (e.g. norms, or the legal system in effect). For example, an agent may be preset to trust any other agent that is owned, or certified, by its owner; it may trust that any authorised dealer will sell products complying to their company's standards; or it may trust another agent if it is a member of a trustworthy

group[1]. Such settings or beliefs (which are mostly domain-specific) can be captured by rules based on the roles of the evaluator and the target agent to assign a predetermined trustworthiness to the target agent. Hence this type of trust is called *Role-based Trust*.

- *Third-party references provided by the target agents*: In the previous cases, the evaluator needs to collect the required information itself. However, the target agent can also actively seek the trust of the evaluator by presenting arguments about its trustworthiness. In our model, such arguments are references produced by the agents that have interacted with the target agents certifying its behaviours[2]. However, in contrast to witness information which needs to be collected by the evaluator, the target agent stores and provides such certified references on request to gain the trust of the evaluator. Those references can be obtained by the target agent (assuming the cooperation of its partners) from only a few interactions, thus, they are usually readily available. This type of trust is called *Certified Reputation*.

Now FIRE integrates all four sources of information and is able to provide trust metrics in a wide variety of situations. Certified Reputation, in particular, greatly enhances FIRE in this respect since the evaluator does not have to obtain this type of information itself (as is the case with other types of trust). Hence, the addition of Certified Reputation decreases the possibility that the evaluator fails to evaluate the trustworthiness of the target agent due to a lack of information. Our working hypothesis here is that integrating these various sources will also enhance the usefulness of the trust model. This will be verified subsequently in our empirical evaluation (see Chapter 5). Specifically, each type of trust information is processed by a particular component of FIRE: Interaction Trust (IT), Witness Reputation (WR), Role-based Trust (RT), and Certified Reputation (CR) components; and the resulting trust values are combined into an overall trust value (see Section 3.7) to benefit from all the available information.

It should be noted that the WR and CR components depend on third-party information (witness experiences and references) and, therefore, they are susceptible

[1]This belief is similar to the neighbourhood reputation in Regret, which calculates the reputation of an agent from the reputation of the agents that it is connected to.

[2]The arguments can also be the target agent's identity, its certifications (e.g. 'authorised dealer', performance awards), its sources of products (to guarantee their quality), and so on. However, deducing trust (or the expected performance) of the target agent from such information requires knowledge about the application domain. This is dealt with in Role-based Trust based on rules encoding an agent's beliefs. Therefore, we only consider third-party references here because they can be quantified and computationally aggregated in a standardised way as we show later in this chapter.

to unreliable information. Since agents in an open MAS are self-interested, they may provide false ratings to gain unwarranted trust for their partners. However, in this chapter, for the sake of simplicity, the basic FIRE model is presented with the temporary assumption that all agents are honest in exchanging information. Although this is unrealistic for an open MAS, our aim now is to ascertain that our philosophy and trust components are actually effective before extending them to more complex scenarios later in this thesis. Specifically, the problem of various sorts of disinformation in reporting ratings are considered and dealt with in Chapter 6, where FIRE is extended to detect and filter out unreliable reports. Hence, in addition to the characteristics of an open MAS, we have made a number of assumptions about the agents and their environment. Before going on to discuss FIRE, we state these assumptions:

Assumption 1 Agents are willing to share their experiences with others (as witnesses or as referees).

Assumption 2 Agents are honest in exchanging information with one another.

In FIRE, except for the RT component which deduces trust based on rules, the other components deduce trust from information about the target agent's behaviour. Here we use *ratings* to capture this type of information. Specifically, then, a rating is the evaluation about an agent's performance given by its partner in an interaction between them. For instance, consider an example where agent a subscribes to a news service provided by agent b. Each time a receives a piece of news from b, it can evaluate the news provided in terms of topicality, quality, and honesty. From its evaluation, agent a may give ratings about agent b's service in those terms for that particular interaction. Ratings are thus tuples in the following form: $r = (a, b, c, i, v)$, where a and b are the agents that participated in the interaction i, and v is the rating value a gave b for the term c (e.g. topicality, quality, honesty). The range of v is $[-1, +1]$, where -1 means absolutely negative, $+1$ means absolutely positive, and 0 means neutral.

Each time agent a gives a rating, it will be stored in the agent's local rating database. Ratings in this database will be retrieved when needed for trust evaluation or for sharing with other agents. However, an agent does not need to store all ratings it makes. As the environment of an open MAS is dynamic, old ratings usually become out-of-date due to changes in the environment. In addition, since each agent has limited resource (i.e. memory), storing all ratings about various agents is not necessarily an option. Therefore, each agent will only store at maximum the

H latest ratings given to another agent. Here H is called the *rating history size*. This parameter is adjustable according to a particular agent's situation.

3.2 Trust Formula

In order to calculate the trust value of a target agent, the components of FIRE will have to collect relevant ratings about that agent's past behaviour. The subsequent sections will define how and which ratings are collected by each component. This section describes how the set of ratings each component collects is used to estimate the target agent's future behaviour, or more specifically, the expected rating value that agent is likely to receive in a future interaction. It is also viewed as the target agent's trust value. Now, a common way to estimate that value is to calculate it as the arithmetic mean of all the rating values in the set. However, these ratings are usually not equally relevant when estimating the expected rating value. For example, some ratings may be older than others and, thus, are deemed to be out-of-date; some may come from a more reliable source that suggests a higher level of credibility compared to others. Therefore, we devise a *rating weight function* ω_{K} for each component of FIRE[3] which calculates the relevance of each given rating. K is thus one of I, R, W, and C standing for interaction trust, role-based trust, witness reputation, and certified reputation respectively. Then instead of considering all ratings equally, the trust value is calculated as the weighted mean of all the ratings available[4], whose weights are given by the corresponding weight function:

$$\mathcal{T}_{\mathsf{K}}(a,b,c) = \frac{\sum_{r_i \in \mathcal{R}_{\mathsf{K}}(a,b,c)} \omega_{\mathsf{K}}(r_i) \cdot v_i}{\sum_{r_i \in \mathcal{R}_{\mathsf{K}}(a,b,c)} \omega_{\mathsf{K}}(r_i)} \tag{3.1}$$

where $\mathcal{T}_{\mathsf{K}}(a,b,c)$ is the trust value that agent a has in agent b with respect to term c, which is calculated by the component K; $\mathcal{R}_{\mathsf{K}}(a,b,c)$ is the set of ratings collected by component K for calculating $\mathcal{T}_{\mathsf{K}}(a,b,c)$; $\omega_{\mathsf{K}}(r_i)$ is the rating weight function that calculates the relevance or the reliability of the rating r_i ($\omega_{\mathsf{K}}(r_i) \geq 0$); and v_i is the value of the rating r_i. In short, the trust value is calculated as the sum of all the available ratings weighted by the rating relevance and normalised to the range of $[-1, 1]$ (by dividing the sum by the sum of all the weights). The rating weight

[3]Since each component of FIRE collects ratings from a different source, it also needs a different way to calculate the relevancy of ratings. For example, the WR component may have information about witness credibility to take into account when weighing ratings, while this information is not relevant to the IT or RT components.

[4]We choose the weighted mean method here because it allows us to take the relevance of each rating into account. Other aggregation methods could equally well be used if desired.

function $\omega_{\mathsf{K}}(r_i)$ is later defined for each component.

As we have discussed, the trust value given above lets an agent know the expected performance of the target agent. However, the trust value alone is not very useful for making task delegation decisions. For example, a trust value of +1 calculated from only 1 rating or from 10 ratings may have different effects on an agent's decision. Therefore, an agent usually also needs to know how likely it is that the target agent will perform at that expected performance (similar to the expected value and deviation measures in statistics). In other words, apart from the trust value, its reliability should also be provided by a trust model. Here, we define a reliability measure that reflects the confidence of the trust model in producing each trust value given the data it took into account. This is given in the form of a reliability value that ranges in $[0, 1]$, where 0 represents complete uncertainty and 1 total confidence. The reliability value is given based on the two following measures:

- *Rating reliability*: Since the rating weight function ω_{K} gives us the relevancy — in other words, the quality or the reliability — of each rating taken into account, the sum of all rating weights reflects the reliability of the rating set taken into account in computing $\mathcal{T}_{\mathsf{K}}(a, b, c)$ in Equation 3.1 above. Therefore, we devise a rating reliability measure based on this sum:

$$\rho_{\mathsf{RK}}(a, b, c) = 1 - e^{-\gamma_{\mathsf{K}} \cdot \left(\sum_{r_i \in \mathcal{R}_{\mathsf{K}}(a,b,c)} \omega_{\mathsf{K}}(r_i)\right)} \tag{3.2}$$

  where $\rho_{\mathsf{RK}}(a, b, c)$ is the reliability value of the rating set $\mathcal{R}_{\mathsf{K}}(a, b, c)$ and γ_{K} is a parameter used to adjust the slope of the reliability function to suit the rating weight function of each component (see Figure 3.1). Since each component has its own rating weight function, it also has a rating reliability function of its own — ρ_{RK}. As above, K is one of I, R, W, and C. R in ρ_{RK} stands for 'rating reliability'. Intuitively, the rating reliability should increase proportionally to the sum of the rating weights. However, since this sum is not limited, we choose the (increasing) function $1 - e^{-x}$ in order that the resulting reliability value is limited in $[0, 1]$. This normalisation is required because the trust and reliability values of FIRE's components will be combined later on in Section 3.7. Moreover, since each rating weight function is defined differently and may have a different range to that of another component's weight function, the parameter γ_{K} is introduced in order to adjust the rate of the rating reliability (Equation 3.2) according to each rating weight function's range. This means the rating reliability

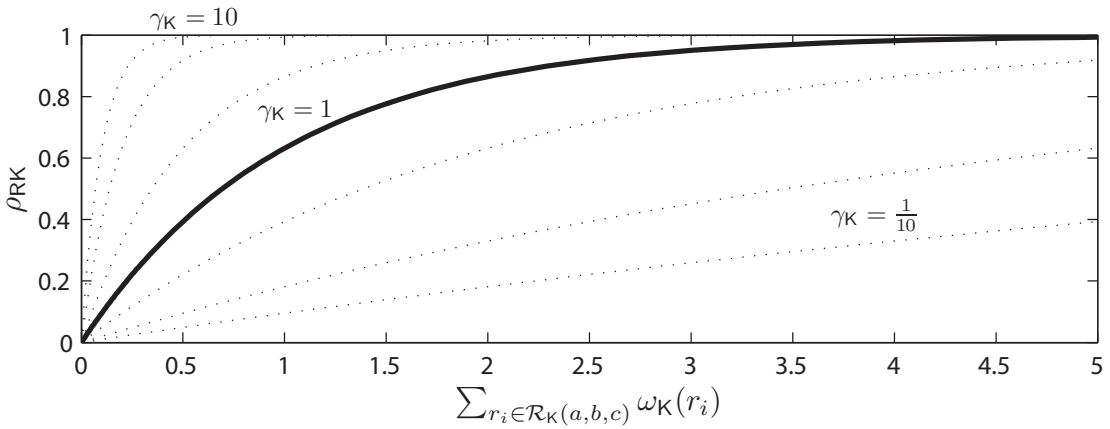

FIGURE 3.1: Rating reliability function $\rho_{RK}(a, b, c)$

function $\rho_{\mathsf{RK}}(a, b, c)$ gradually increases from 0 (the lowest reliability) to 1 (the highest reliability) when the sum of rating weights increases from 0 to $+\infty$.

- *Deviation reliability*: The greater the variability in an agent's past behaviour (which is reflected by its rating values), the more volatile it is likely to behave in future interactions. Therefore, the deviation in the rating values is also a metric that reflects a trust value's reliability:

$$\rho_{\mathsf{DK}}(a, b, c) = 1 - \frac{1}{2} \cdot \frac{\sum_{r_i \in \mathcal{R}_K(a,b,c)} \omega_K(r_i) \cdot |v_i - \mathcal{T}_K(a, b, c)|}{\sum_{r_i \in \mathcal{R}_K(a,b,c)} \omega_K(r_i)}, \tag{3.3}$$

where $\rho_{\mathsf{DK}}(a, b, c)$ is the deviation reliability value of the trust value $\mathcal{T}_K(a, b, c)$. Here, D in ρ_{DK} stands for 'deviation'. Basically, Equation 3.3 calculates the deviation of ratings' values in the set of ratings $\mathcal{R}_K(a, b, c)$ around the 'expected' value (i.e. the trust value); the calculated deviation is then normalised to $[0, 1]$. Intuitively, when there is no deviation in the rating's value (i.e. the target agent performs consistently), the deviation reliability is 1 (i.e. the most reliable); and it decreases proportionally to 0 (i.e. the least reliable) when the deviation increases.

In order to take both of these reliability factors above into account, the reliability value of the produced trust value, denoted by $\rho_K(a, b, c)$, is defined as the combination of the rating reliability and the deviation reliability measures:

$$\rho_K(a, b, c) = \rho_{\mathsf{RK}}(a, b, c) \cdot \rho_{\mathsf{DK}}(a, b, c) \tag{3.4}$$

3.3 Interaction Trust

As introduced in Section 3.1, Interaction Trust is built from the direct experience of an agent. It models the trust that ensues from the direct interactions between two agents. Here we simply exploit the direct trust component of Regret (see Appendix A.2) since this meets all our requirements for dealing with direct experiences. In more detail, each agent rates its partner's performance after every transaction and stores its ratings in a local rating database. When calculating the IT value for agent b with respect to term c, agent a has to query its database for all the ratings that have the form $(a, b, c, \_, \_)$, where the '$\_$' symbol can be replaced by any value. We call the set of those ratings $\mathcal{R}_{\mathsf{I}}(a, b, c)$.

Since older ratings may become out-of-date quickly, we use recency of the ratings as a rating weight function to give recent, and likely more up to date, ratings more weight than older ratings in IT evaluation. However, as pointed out in Appendix A.2, Regret's method of calculating rating recency has several undesirable characteristics. Therefore, we devise a new rating recency function based on the time difference between the current time and the rating time since this metric reflects precisely how old (i.e. how recent) a rating is. In order to make our rating recency function adjustable to suit the time granularity in different applications, the parameter λ, called the *recency scaling factor*, is introduced in the function (to scale time values). Our rating recency function, which is also used as the rating weight function for IT, is given by the following formula:

$$\omega_{\mathsf{I}}(r_i) = e^{-\frac{\Delta t(r_i)}{\lambda}} \tag{3.5}$$

where $\omega_{\mathsf{I}}(r_i)$ is the weight for the rating r_i (used in Equation 3.1) and $\Delta t(r_i)$ is the time difference between the current time and the time when the rating r_i is recorded. In our model, analogously to human perception, we view the time difference of two recent events as more significant than the same one of two older events (see Footnote 2, page 133 for an example). Hence, the exponential function above is chosen for rating recency because its shape over time fits our view on how the recency of ratings should affect an agent's decision about trust (see Figure 3.2). Our intuition is that new ratings are deemed to reflect the target agent's current performance more accurately than old ratings, and our recency function here is to help FIRE adapt quickly to any changes in that agent's performance. In Equation 3.5, the parameter λ is hand-picked for a particular application depending on the time unit used. For instance, if the time unit used is *day* and we want a rating obtained *five days* earlier to only have half the effect

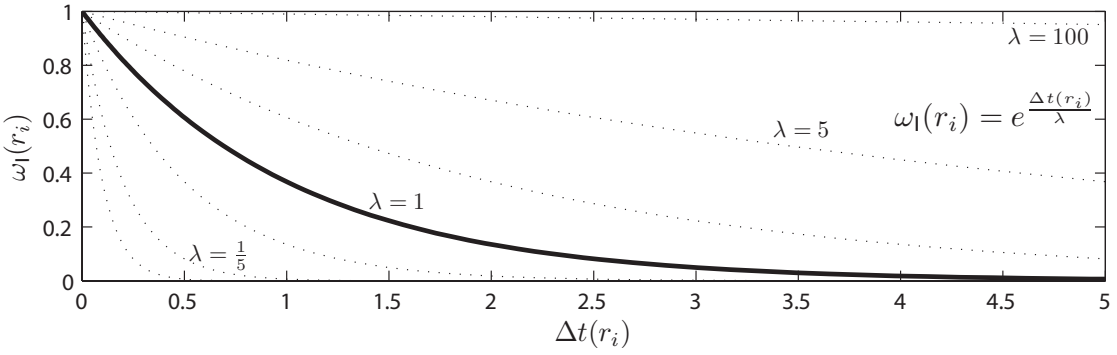

FIGURE 3.2: Rating weight function of interaction trust component.

of a new rating obtained *today* (i.e. rating weights of 0.5 and 1 respectively; $\Delta t(r_i) = 5$) then $\lambda = -\frac{5}{\ln(0.5)}$.

Given the rating set $\mathcal{R}_{\mathsf{I}}(a, b, c)$ and the rating weight function $\omega_{\mathsf{I}}(r_i)$ as specified above, the IT value $\mathcal{T}_{\mathsf{I}}(a, b, c)$ and its reliability $\rho_{\mathsf{I}}(a, b, c)$ are calculated using the general trust formula as defined in Equations 3.1 and 3.4 (Section 3.2).

3.4 Role-Based Trust

Role-based trust models the trust resulting from the role-based relationships between two agents (e.g. owned by the same company, a service provider and its registered user, friendship relationship between their owners). Since there is no general method for computationally quantifying trust based on this type of relationship, we use rules to assign RT values. As previously discussed, those rules are used to encode knowledge about the trust dynamics in the application domain. Therefore, they are usually domain-specific and must be specified by an agent's designer or its owner. In other words, this component provides the means of adapting FIRE to a particular environment and, thus, making it perform better in that environment. Here, rules are tuples of the following form: $rul = (role_a, role_b, c, e, v)$, which describes a rule that if $role_a$ and $role_b$ are the roles of agent a and b respectively, then the expected performance of b with respect to the term c in an interaction with a is v ($v \in [-1, 1]$); $e \in [0, 1]$ is the level of influence of this rule on the resulting RT value or the belief strength of agent a on the rule. For example, possible rules may be:

$rul_1 = (\text{buyer}, \text{seller}, \text{quality}, 0.3, -0.2)$,
$rul_2 = (\_, \text{government-seller}, \text{quality}, 0.8, 0.0)$,
$rul_3 = (\_, \text{team-mate}, \text{honesty}, 1.0, 1.0)$.

Thus, rul_1 expresses an agent's belief that an ordinary seller will usually sell a product of slightly lower quality than agreed, but the reliability of this belief is low (0.3); rul_2 expresses a stronger belief that an agent can expect a governmental seller to do what is agreed in terms of product quality; and rul_3 tells an agent to expect total honesty from its team mate (e.g. agents of the same owner). Here, rul_1 and rul_2 encode norms of the environment, while rul_3 is the belief based on an arrangement between agents. Such rules are given to the agent by its owner. Additional rules can naturally be added during an agent's life cycle.

Each agent has its own set of rules which are stored in a (local) rule database. In order to determine the RT of agent b with respect to term c, agent a looks up the relevant rules from its rule database. We call the set of those rules $\mathcal{R}_{\mathsf{R}}(a, b, c)$. Since the form of a rule is very analogous to that of a rating, the general trust formula in Equation 3.1 can be used to calculate the RT of b, which is denoted by $\mathcal{T}_{\mathsf{R}}(a, b, c)$, from this set. Here, the level of influence of each rule is used as the weight for that rule: $\omega_{\mathsf{R}}(r_i) = e_i$. Therefore, it should be noted that in case there exist conflicts in the applicable rules (i.e. contradicting expected performance values), all these rules will be taken into account but the deviation measure reliability (ρ_{DK}) of the resulting trust value will be low (because of the high deviation of the rules used). This, in turn, will result in a low reliability of the RT trust value, which shows that the RT trust value has a low predictive power and so it will be weighted accordingly in calculating the overall trust value (see Section 3.7).

3.5 Witness Reputation

The witness reputation of a target agent b is built on observations about its behaviour by other agents (witnesses). In order to evaluate the WR of b, an agent a needs to find the witnesses that have interacted with b. Here, it is assumed that agents in open MAS are willing to share ratings that they made and to help others search for witnesses. In order to find relevant witnesses, we implement a variant of Yu and Singh's referral system without using the VSM model[5]. Instead, our system assumes that each agent has a measure of the degree of likeliness with which an agent can fulfil an information query about witness information and witness

[5]As pointed out in Yolum and Singh [2004], the VSM model does not support hierarchy in expertise types, which can be better represented by service graphs. In this respect (i.e. modelling expertise), there is no universal model for all applications. Therefore, we leave the choice of expertise model to end users as they can evaluate which method is best suited to their particular applications.

locating. This measure needs to be defined in an application specific manner. For example, in our testbed (described in Chapter 4), an agent is assumed to know local agents (those that are adjacent to it) better and, therefore, we use the physical distance between an acquaintance and the target agent as the knowledge measure. Thus the nearer to the target agent, the more likely the acquaintance is to know it. This measure is used in the referral process to help locate witnesses. However, it should be noted that the resources available to each agent are limited (in terms of its memory and communication cost) and the evaluator (agent a) usually has limited time for trust evaluation (before it has to initiate an interaction). Thus, the process of locating witnesses should typically be limited according to an agent's time constraints, though this may result in no witnesses being found (even though appropriate agents are available in the system). Here, the parameters n_{BF} (called the *branching factor* [Yu and Singh, 2003b]) and n_{RL} (called the *referral length threshold*, or the depth of referral graphs in [Yu and Singh, 2003b]) are introduced for that purpose. Specifically, n_{BF} is used to limit the number of acquaintances to which a query is forwarded and n_{RL} to limit the length of referral chains. Besides restricting the search range of agent a due to time constraints, the referral length threshold also helps an agent not to waste its effort querying too distant agents because, intuitively, the further the witness is from a (in terms of the length of the referral chain to the witness from a), the less reliable/relevant its information. At present, n_{BF} and n_{RL} need to be hand-picked according to the an agent's resource constraints and its environment's acquaintance networks.

Specifically, the process of evaluating WR is as follows:

1. When agent a assesses the WR of agent b with respect to term c, denoted by $\mathcal{T}_{\mathsf{W}}(a, b, c)$, it sends out a query for ratings of the form $(\_, b, c, \_, \_)$ to n_{BF} acquaintances that are likely to have relevant ratings on agent b and term c (see Figure 3.3, where $n_{\mathsf{BF}} = 2$).

2. These acquaintances, upon receiving the query, try to match it to their own (local) rating databases. If they find matching ratings, it means they have had interactions with b, they will return the ratings found to a.

3. If they cannot find the requested information, they will return referrals identifying their n_{BF} acquaintances that they believe are most likely to have the relevant ratings to the query (based on the knowledge measure) so that a can look further.

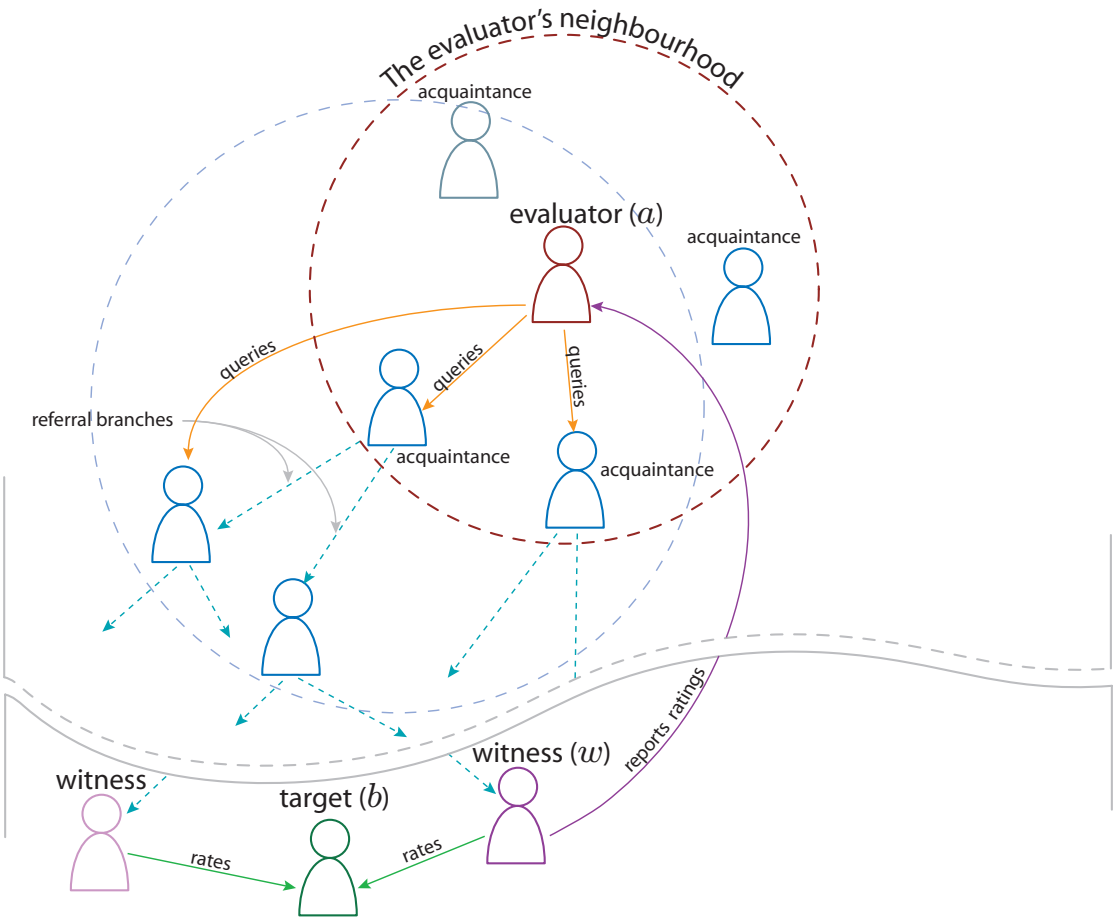

FIGURE 3.3: Referral process.

4. This process continues until a finds sufficient witnesses or the length of its referral chains reach the defined threshold n_{RL}.

It should be noted here that in this process we implicitly assume that agents in a's referral network are willing to help a find the required witness ratings. This is not a trivial assumption and needs to be guaranteed for this referral process (as for any mechanism based on third-party information) to work, especially in open MAS where agents are self-interested. However, we do not consider how such a guarantee can be obtained in this thesis because that task would very much depend on the particular application domain being considered. Thus, end-users who wish to make use of WR need to provide necessary measures for this willingness assumption to hold (e.g. obtaining an agreement between agents on sharing witness information or paying for any information request).

The set of ratings collected from the referral process, denoted by $\mathcal{R}_{\mathsf{W}}(a, b, c)$, is used to calculate the WR of agent b (i.e. $\mathcal{T}_{\mathsf{W}}(a, b, c)$) following Equation 3.1. Here, the rating weight function for WR $\omega_{\mathsf{W}}(r_i)$ is intended to reflect a witness rating's

quality which also includes the rating's credibility (since in realistic environments agents may give false/inaccurate ratings). However, as we currently assume all agents are honest, only the recency of ratings is temporarily used as per Section 3.3 (i.e. $\omega_{\mathsf{W}}(r_i) = \omega_{\mathsf{I}}(r_i)$, see Equation 3.5). A model of witness credibility is described and incorporated into FIRE in Chapter 6.

3.6 Certified Reputation

Certified reputation of a target agent b comprises certified references[6] about its behaviour from third-party agents. Such information is obtained and stored by the target agent itself and made available to any other agent that wishes to evaluate its trustworthiness for further interactions (somewhat like a reference when a person is applying for a job). The references are in the form of ratings given by agent b's partners about its performance in (past) interactions between them. These ratings allow agent b to prove its achievable performance as viewed by its previous interaction partners and then to gain the trust of its potential partners. However, since it can choose which ratings to put forward, a rational agent will only present its best ratings. Therefore, it should be assumed that CR information probably overestimates an agent's expected behaviour. Thus, although it cannot guarantee agent b's minimal performance in future interactions, the CR information does reveal a partial perspective on agent b's capabilities (which is certainly useful for trust evaluation in the absence of other sources of information).

Though CR may have lower predictive power than the other types of trust/reputation (where all bad and good ratings can be collected), it is useful because of its wide applicability. With the cooperation of its partners, agent b can obtain their references from just a small number of interactions[7]. From our evaluation, for instance, in a society where 100 agents provide a service to 500 others, agents using direct experience to evaluate trust require more than 100 interactions to achieve a reasonable level of performance, which is still less than what is achieved by agents using CR after 5 interactions (see Section 5.3 for more detail). In addition to its

[6]It is assumed that some form of security mechanism (such as a public-key infrastructure) is employed to ensure that the provided references cannot be tampered with. For instance, all references could be accompanied by digital signatures from the issuers using their private keys [Zimmermann, 1995]. By so doing, any change to a reference will be easily detected. Digital signatures are also a means to verify the references' origins.

[7]In many scenarios, such as those in the Internet, established service providers (e.g. news services or online merchants) usually have high volumes of interactions (at any time). Therefore, if they adopt the CR process outlined here, we can reasonably expect that such providers will have an abundance of performance ratings readily available.

high availability, since references are stored by the target agent and provided directly to the evaluator, CR has a very low running cost (i.e. time, communication, processing cost) compared to witness reputation. Since CR information comes from the target agent, the CR component complements the other components of FIRE, which use information collected by the evaluator, reducing the chances that they may fail to calculate trust due to lack of input (see Section 3.1). Thus, incorporating the CR component makes FIRE able to provide a trust value in most circumstances.

In more detail, the process of CR is as follows:

- After every transaction, agent b asks its partners to provide their certified ratings about its performance from which it can choose the ratings to store in its (local) rating database.
- When agent a contacts b to express its interest in using b's service, it asks b to provide references about its past performance with respect to an interested term c.
- Agent a receives the set of certified ratings of b from b, which we call $\mathcal{R}_{\mathsf{C}}(a, b, c)$ (C to denote this set is obtained via the CR mechanism), and calculates the CR of b based on this set.

In this process, since agent b relies on its interaction partner's cooperation to get references, agents may refuse to give out their ratings (as in the case of witness reputation). However, this is a much smaller problem than that in witness reputation because this information is requested far less frequently (each referee is requested to give its rating only once). Moreover, giving such information could be made a standard part of any agreement for task allocation and so agents could be forced to give it. The most notable point in this process is that when agent a makes the trust evaluation, it only involves agents a and b. Since the certified ratings about b are stored by b itself, they are immediately available to a as in the case when a uses its own experience. It should also be noted that when a referee provides references to an interaction partner, it surrenders its privacy with respect to how it values that partner's performance. This may lead to various possible reactions of that partner (e.g. it may retaliate against the referee for a bad reference or it may treat the referee differently the next time to get a better reference). However, due to the vast number of possibilities in the reactions of both agents (i.e. the referee and the referred agent), we do not consider the effects of giving up privacy in CR here and defer it to future work.

Having obtained the references of b, a can calculate the CR value of b using the formula in Equation 3.1. However, since there is no guarantee about the honesty of agents in an open MAS, we need measures to prevent or to minimise the adverse effects of lying (e.g. collusion between the target agent and its referees in producing falsely inflated references). Here, we use the rating weight function $\omega_{\mathsf{C}}(r_i)$ to reflect the credibility of a reference (i.e. rating). Again, since we are not considering the problem of lying in this chapter, the rating weight function for CR is defined based only on the recency of ratings as per Section 3.3 (i.e. $\omega_{\mathsf{C}}(r_i) = \omega_{\mathsf{I}}(r_i)$). The value of CR, $\mathcal{T}_{\mathsf{C}}(a, b, c)$, and its reliability, $\rho_{\mathsf{C}}(a, b, c)$, are calculated as defined in Section 3.2.

3.7 An Overall Value

When using FIRE to evaluate trust, an agent can decide which components it will use for trust evaluation according to its needs and situation. However, as each component produces trust values from a separate source of information, we believe that in combining the four components, and effectively the four information sources, it will in most cases yield a higher level of performance (as confirmed by the empirical evaluation in Section 5.3). Thus, we recommend combining all the aforementioned trust values into a single composite measure to give an overall picture of an agent's likely performance. As all trust values in FIRE come with reliability values, instead of averaging the trust values from the four components, we again use the weighted mean method to calculate the composite trust value, denoted by $\mathcal{T}(a, b, c)$), to take each trust value's reliability into account:

$$\mathcal{T}(a, b, c) = \frac{\sum_{\mathsf{K} \in \{\mathsf{I},\mathsf{R},\mathsf{W},\mathsf{C}\}} w_{\mathsf{K}} \cdot \mathcal{T}_{\mathsf{K}}(a, b, c)}{\sum_{\mathsf{K} \in \{\mathsf{I},\mathsf{R},\mathsf{W},\mathsf{C}\}} w_{\mathsf{K}}} \tag{3.6}$$

where $w_{\mathsf{K}} = W_{\mathsf{K}} \cdot \rho_{\mathsf{K}}(a, b, c)$, and W_{I}, W_{R}, W_{W}, W_{C} are the coefficients corresponding to the IT, RT, WR, and CR components. Here, the composite trust value is calculated from the four component trust values and each of them is weighted by both its reliability (as given by $\rho_{\mathsf{K}}(a, b, c)$) and the corresponding component coefficients (i.e. W_{K}). These coefficients are set by end users to reflect the importance of each component in a particular application. For instance, one can set W_{I} and W_{R} to be the highest in the four coefficients since the IT and RT use an agent's own information and should be the most reliable components; W_{C} can be set to be the lowest since the CR information is from third-parties and tends to

exaggerate the target agent's performance. However, these coefficients can be automatically set according to changes in an agent's environment by implementing the adaptability extension of FIRE as shown in Chapter 7.

The composite trust value also has a corresponding reliability value, denoted by $\rho_{\mathcal{T}}(a, b, c)$, which is calculated from the components' reliability values weighed by the component coefficients in a similar manner:

$$\rho_{\mathcal{T}}(a, b, c) = \frac{\sum_{\mathsf{K} \in \{\mathsf{I},\mathsf{R},\mathsf{W},\mathsf{C}\}} w_{\mathsf{K}}}{\sum_{\mathsf{K} \in \{\mathsf{I},\mathsf{R},\mathsf{W},\mathsf{C}\}} W_{\mathsf{K}}} \tag{3.7}$$

3.8 Summary

This chapter has described the basic FIRE model for trust evaluations in open MAS. The model itself is composed from four trust and reputation components: Interaction Trust, Role-based Trust, Witness Reputation, and Certified Reputation. Each component derives trust values from a separate source of information and then the component trust values are combined to provide an overall picture of an agent's trustworthiness. Thus, reviewed against our requirements for a trust model in open MAS (Section 2.5), FIRE satisfies the requirement **R1b** by making use of IT, RT, and WR. FIRE also introduces CR, a novel type of reputation, that addresses the inherent shortcomings of interaction trust (the lack of direct experience) and witness reputation (the difficulty in finding witness reports). Combining all the four trust/reputation components not only allows FIRE to produce more useful trust values than using fewer components (as confirmed from our evaluation in Section 5.3), but also makes it serviceable in the absence of some of the sources of information. In this respect, CR is particularly relevant because it greatly enhances the serviceability of FIRE by transferring the task of collecting third-party ratings (i.e. references) to target agents, who are more capable than evaluators and incentivised to do so. Therefore, an agent that newly joins an environment can evaluate the trustworthiness of others from their references even when it has not had previous experience with them and cannot find any witness for them. Thus, FIRE satisfies the requirement **R1a** by addressing the bootstrapping issue of newly joined agents[8].

[8]Obviously, there are still cases when FIRE cannot produce a trust value. Specifically, those are when a service provider newly joins the system. Hence, it does not have references about its performance and other agents do not have past experience with it. However, in a realistic scenario, in order to promote its service, that provider can join a (popular) scheme/organisation that provides quality assurance about its members' service. For example, a car dealer can obtain

In order to be compatible with the distributed nature of open MAS, FIRE is designed such that an individual agent can make trust evaluations itself without having to rely on a central authority (Requirement **R2a**). Various mechanisms are provided so that an agent can collect trust information about its peers and aggregate such information to derive trust values. Therefore, the reliability of those trust values is guaranteed (in contrast to the case where trust values are provided from a third-party). Although we have not done an analysis on the scalability of FIRE, in our experiments where 500, 1000, and 1500 agents using FIRE are deployed (Chapters 5, 6 and 7), it is observed that the execution time of those experiments varies linearly to the number of agents deployed. Thus, given its decentralised nature, we believe that FIRE is scalable to the large number of agents that may be present in an open MAS (Requirement **R2b**). The process of CR (Section 3.6) is also beneficial in this respect since it makes trust information (i.e. references) highly accessible in most circumstances. As for the adaptability requirement (Requirement **R3**), the required trust information in our model (i.e. ratings, rules) is defined in such an abstract manner that it can be applicable in various application domains. The behaviour of FIRE is also parameterised and can be fine tuned for a particular environment or application if desired. Finally, since FIRE uses only first-hand evidence of agent interactions (i.e. ratings produced by the participating agents), the problem of correlated evidence (Section 2.4.4, Requirement **R4b**) is avoided.

In summary, the basic FIRE model satisfies all the requirements for a trust model in open MAS that we outlined in Section 2.5, except that it has not considered the problem of inaccurate third-party reports (Requirement **R4a**). FIRE is extended in Chapter 6 to address this problem. In the next phase of our research, we aim to evaluate the effectiveness of FIRE in evaluating trustworthiness by helping agents select good interaction partners and, in addition, how it performs in dynamic situations (Requirement **R2c**, see Section 2.4.2). Before doing this, however, we need to describe the evaluation methodology that we use. This is detailed in the next chapter.

---

the title 'authorised dealer' from a car manufacturer. Such (popular) membership (and inherently its quality assurance) can be recognised by other agents (via rules in FIRE's RT component) and thus helps the provider to sell its service.

Chapter 4

Evaluation Methodology

In order to employ a formal and systematic evaluation of the work in this thesis, a set of experiments has been designed to evaluate FIRE's performance. In this work, since trust is an abstract and multi-faceted concept, there is no base for analytic evaluation. Instead, empirical evaluation is used as the method of measurement because it allows us to assess the performance of a trust model in terms of how much benefit it can bring to its users (which can serve as a measure to justify its use). In addition, there are a number of internal variables which control the behaviour of FIRE, as well as external variables which define the environment in which our model is being used (see Section 4.2). These variables are interrelated and need to be considered in a broad range of situations. Empirical techniques allow us to manipulate these variables, conduct the experiments, and analyse the results. Thus, they are suitable for our evaluation purpose. In particular, the evaluation technique we use is called *hypothesis testing* [Cohen, 1995, pg. 106]. With this method, hypotheses are formed to express the intuitions about FIRE's performance under a variety of situations. Experiments are then conducted and their results are used in statistical inference to either accept or reject the hypotheses.

This chapter explains in detail the procedure of hypothesis testing (Section 4.1), the testbed in which the experiments are carried out (Section 4.2), and how they are set up (Section 4.3). Finally, a summary is provided in Section 4.4.

4.1 The Methodology

As discussed above, we will evaluate FIRE's performance in terms of how much benefit an agent may gain by using it. In the context of this thesis, the aim of the trust model is to help agents distinguish good interaction partners from bad ones (Section 1.3) and, thus, to allow them to avoid losing utility by choosing not to interact with bad agents. Therefore, the difference between the utility gained[1] by an agent using a trust model and that gained by another agent using no trust model in choosing interaction partners can be interpreted as the added value of that trust model (or more concisely the performance of that trust model). Henceforth, the performance (i.e. the utility gain) of an agent using, say, FIRE is used interchangeably with the performance of FIRE. In order to be able to do so, we have to exclude all other factors than trust models that can affect an agent's performance; these include domain knowledge, negotiation issues, and planning (see Section 4.2). By removing such extraneous factors, the trust model is left as the only differentiating factor in an agent's capability (e.g. agents using no trust model, using FIRE, or using another model). This then allows us to objectively compare the performance of trust models by making comparisons between the performance of the corresponding agents using them.

In more detail, it is desirable that FIRE is evaluated in all possible situations in order to make sure that it will always behave properly. However, since the environment of an open MAS is both complex and dynamic, there are uncountably many factors that can affect FIRE's performance. For instance, it can be affected by the population of the agents in an open MAS, the interactions and relationships between them, and their behaviours. These are all unbounded external variables to FIRE. Therefore, it is impossible to exhaustively explore all the environment space in order to comprehensively evaluate FIRE. To combat this, we introduce randomness into the testbed we use to evaluate FIRE (see Section 4.2 for more details) to make it similar to a continually changing environment of an open MAS. In addition, a group containing a large number of agents (typically 500) using FIRE are evaluated at the same time, in which each agent has a particular situation defined by the environment's randomness. In so doing, FIRE is evaluated under a wide range of situations and its performance can reasonably be measured as the mean performance of all the agents in the group.

[1] Since agents are typically designed to select their actions based on the expected utility gained from those actions in order to maximise their own utility [Wooldridge, 2002], it is implicitly assumed here that the utility an agent gains from an interaction can be quantitatively measured.

In our evaluation, we are interested in answering the question:

> 'How do agents using FIRE perform in comparison to agents using no trust model and to other trust models?'

This requires us to compare FIRE's performance with that of another model. We are also interested in experiments showing how FIRE performs with and without a particular component because such results help us to confirm/reject our intuitions and to justify our design decisions. Specifically, this requires us to compare the performance of groups of agents using FIRE with different configurations. However, a mere comparison of the mean performance of two groups of agents does not allow us to conclude that one group performs better than the other in all the cases. The reason is that the population of possible situations is infinitively large and the results from one experiment are only from a small sample of that population and, moreover, it might not be a typical result for the population. Given this, statistical inference techniques should be used since they allow us to draw a conclusion about an unseen population given a relatively small sample. Thus, to the extent that a sample is representative of the population from which it is drawn, statistical inference permits generalisations of conclusions beyond the sample [Cohen, 1995, pg. 105]. In our experiments, we use a statistical inference method called *hypothesis testing*, which allows us to answer a yes-or-no question about the population and assess the probability that the answer is wrong[2].

Now suppose we need to determine whether the performance of agents using FIRE is better than the performance of agents using no trust model. Since trust models typically learn about its user's environment and gradually improve its performance through interactions (Section 2.2), it does not make sense to compare the performance of models after different periods of use. Therefore, we need to choose a test period and compare the performance of models after that same period (say, five interactions[3]).

[2]In analysing (experimental) data about two populations, say their income levels, the fact that the means of the two sample groups' incomes are different does not always indicate that the two populations have different levels of income if randomness can affect sample selection. Thus, it is possible that the means of these two particular samples are different, but the means of the two populations' incomes are not. Hypothesis testing methods allows us to confirm with a predefined confidence level whether the difference of the two means actually indicates that one group has higher income than the other, eliminating the random factor in selecting the samples (see [Cohen, 1995] for more detail).

[3]Test periods can also be chosen in other time units (e.g. 5 minutes or 100 generated clock ticks). However, since we are using the mean performance of a group of agents employing FIRE as the performance measure and in a timed period each of the agents may complete a varied

Term	Definition
FIRE	The name of the group of agents using FIRE
NoTrust	The name of the group of agents using no trust model
n	The number of finished interactions chosen as the test period
N_{FIRE}	The number of agents in group FIRE
N_{NoTrust}	The number of agents in group NoTrust
μ_{FIRE}	The population mean performance for FIRE; the mean performance obtained by measuring the utility gain of an infinitive number of agents using FIRE in their nth interaction and in all possible environments.
μ_{NoTrust}	The population mean performance for NoTrust, obtained as above.
$\overline{P}_{\mathsf{FIRE}}$	The mean performance of a sample of agents using FIRE after their nth interaction
$\overline{P}_{\mathsf{NoTrust}}$	The mean performance of a sample of agents using no trust model after their nth interaction
s_{FIRE}	The variance of the performance sample of FIRE
s_{NoTrust}	The variance of the performance sample of NoTrust

TABLE 4.1: Terms used in the hypothesis testing example.

With the terms defined in Table 4.1, the procedure of hypothesis testing used in our experiments is as follows (adapted from [Cohen, 1995]):

1. Formulate a null hypothesis and an alternative hypothesis, denoted H_0 and H_1, respectively:
   H_0: $\mu_{\mathsf{FIRE}} = \mu_{\mathsf{NoTrust}}$;
   H_1: $\mu_{\mathsf{FIRE}} > \mu_{\mathsf{NoTrust}}$.

2. Gather a sample of performance (i.e. utility gain) of agents in FIRE and NoTrust in their nth interaction, and calculate the mean performance of each group, denoted by $\overline{P}_{\mathsf{FIRE}}$ and $\overline{P}_{\mathsf{NoTrust}}$. Call N_{FIRE} and N_{NoTrust} the number of samples in group FIRE and that in group NoTrust respectively.

3. Assuming the null hypothesis is true (i.e. there is no difference between the performance of FIRE and NoTrust), calculate the probability of obtaining the sample means $\overline{P}_{\mathsf{FIRE}}$ and $\overline{P}_{\mathsf{NoTrust}}$. This probability is given by the *two-sample t-test* function that takes into account $\overline{P}_{\mathsf{FIRE}}$, $\overline{P}_{\mathsf{NoTrust}}$, N_{FIRE}, N_{NoTrust}, and

number of interactions depending on its operation, the performance measure after such a period can greatly fluctuate between experiment runs. This is because the performance of a trust model depends on the amount of information it learns after each interaction and, thus, a varied number of interactions will result in a varied level of a specific performance. Therefore, the number of finished interactions is a more suitable basis than time unit for choosing test periods. From the view point of an individual agent, it also allows the performance measure to show how quickly a trust model learns roughly in terms of an individual agent's cost (i.e. interaction cost, possible loss because of bad bootstrapping performance of the trust model), which is more relevant to the agent than how much time it has been operating in its environment.

the sample variances of the two groups (see [Cohen, 1995, pg. 127] for more details).

4. If this probability is lower than 0.05, reject the null hypothesis in favour of the alternative hypothesis. This means that we can conclude with a confidence level of 95% that H_1 is true, or the performance of FIRE is (statistically) significantly better than that of NoTrust.

For example, after an experiment, assume the following data is obtained at $n = 5$:

- Group FIRE: $\overline{P}_{\mathsf{FIRE}} = 6.3627$, $N_{\mathsf{FIRE}} = 500$, and $s_{\mathsf{FIRE}} = 3281.2384$.
- Group NoTrust: $\overline{P}_{\mathsf{NoTrust}} = -1.0543$, $N_{\mathsf{NoTrust}} = 500$, and $s_{\mathsf{NoTrust}} = 4641.8361$.

Assuming H_0 is true, the probability of obtaining this data given by the two sample t test is 2.81×10^{-242}. Therefore, we can conclude that the performance of FIRE is indeed significantly better than that of NoTrust (in this case with the confidence level of nearly 100%).

The hypothesis testing procedure above can determine that, for instance, using FIRE will yield a better utility gain than using no trust model at the 5th interaction. However, it is not clear exactly how quickly FIRE can achieve that level of performance. Moreover, we are also interested in whether FIRE can maintain the same level of performance at later interactions. Thus, instead of choosing a fixed test period, in each experiment we carry out the hypothesis testing procedure for every test period from the 1st interaction to the 200th one[4]. Thus, the mean performance of each group of agents in terms of utility gain (UG) is plotted on a chart to show the trend of performance change (see Figure 4.1 for an illustration). Now, since showing the actual hypothesis tests will include many (uninteresting) numbers, we will show only the result of the tests on the chart using the second y-axis (on the right). For example, with the result for $n = 5$ above, we assign rank 2 to FIRE and rank 1 to NoTrust. This is to show that the corresponding hypothesis test concludes that FIRE outperforms NoTrust and that the performance difference is statistically significant (using the confidence level of 95%). The rank lines are named using the group names but prefixed by R. If at some interactions the rank

[4]It is shown in all our experiments that the performance of all the models tested are stable by around the 200th interaction, or, put another way, that the interesting trends of performance change can be observed in this period. In our experiments, most agents make about 300–400 interactions. However, given this stability, we choose to show only the results from the first 200 interactions for reasons of simplicity.

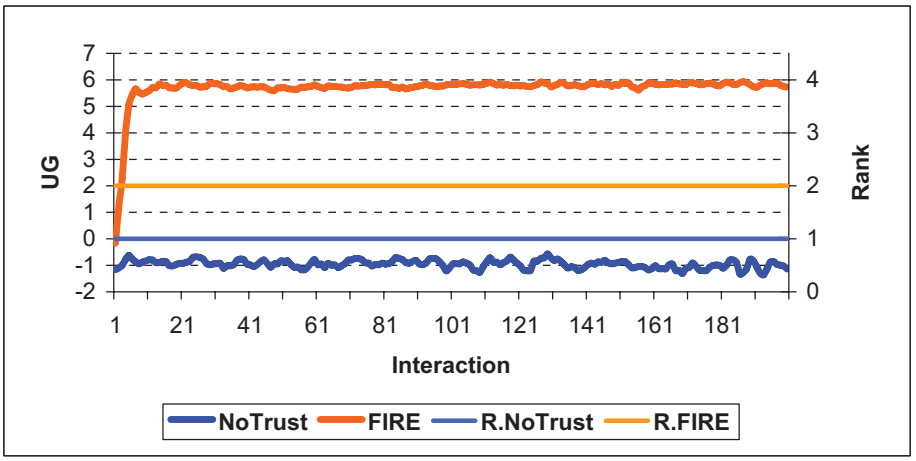

FIGURE 4.1: Hypothesis testing example chart.

lines are collapsed into one line, it means that at the corresponding test period the probability that H_0 is true is greater than 0.05. This means that we cannot reject the null hypothesis and can only conclude that the performance difference between the two groups is not statistically significant for that test period.

4.2 The Testbed

Having defined the evaluation methodology, we need a testbed to run the experiments on. This section describes the testbed we use and discusses various design decisions that aim to ensure it captures the key characteristics of open MAS (as detailed in Section 1.1). In particular, the testbed domain setup is described in Section 4.2.1, and then Section 4.2.2 presents the factors of randomness introduced into the testbed to simulate the dynamism in an open MAS.

4.2.1 The testbed domain description

The testbed environment for evaluating FIRE is a multi-agent system consisting of agents providing services (called *providers*) and agents using those services (called *consumers*). We assume that the performance of a provider (and effectively its trustworthiness) in a particular service it provides (e.g. news services) is generally independent from that in another service (e.g. weather services or banking services). Therefore, without loss of generality, and in order to reduce the complexity of the testbed's environment, it is assumed that there is only one type of service in the testbed. Hence, all the provider agents offer the same service. However, their

performance (i.e. the quality of the service) differs. The agents are situated randomly on a spherical world whose radius is 1.0 (see Figure 4.2). Each agent has a *radius of operation* (r_o — depicted by a dotted circle around an agent in Figure 4.2) that models the agent's capability in interacting with others (e.g. the available bandwidth or the agent's infrastructure) and any agents situated in that range are the agent's acquaintances. In the case of a provider, its radius of operation serves as the normal operational range in which it can provide its service at its full capability without loss of quality. For consumers outside that provider's normal operational range, the quality of service they receive from it gradually degrades. This simulates the phenomenon that each agent usually has particular circumstances (here its location) which affect service delivery. For example, two distant agents may experience significant network latency during their interactions, or a seller agent in the UK may charge another agent extra for shipping goods abroad and the goods may arrive much later than usual.

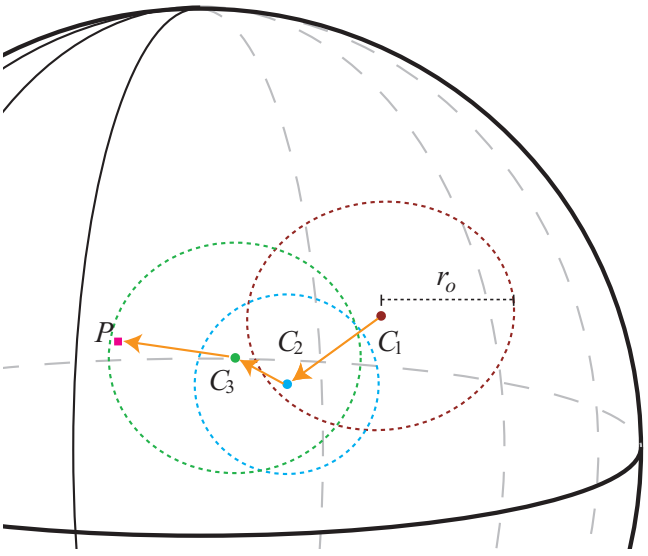

FIGURE 4.2: The spherical world and a path from consumer C_1 (through C_2 and C_3) to provider P based on neighbourhood.

Simulations are run in the testbed in rounds (of agent interactions). Events that take place in the same round are considered simultaneous. The round number is used as the time value for events. In each round, if a consumer agent needs to use the service it can contact the environment to locate nearby provider agents[5] (in terms of the distance between the agents on the spherical world). The consumer agent will then select one provider from the list to use its service. The selection

[5]This is to simulate a situation in which only a portion of the provider population is available to a given agent. For example, a retail banking agent can only serve customers in its country. In addition, as the degradation of service quality is proportional to the distance between a provider and its consumer, providers that are too distant may not be useful.

process relies on the agent's trust model to decide which provider is likely to be the most reliable. Consumer agents without a trust model randomly select a provider from the list. On the other hand, an agent with a trust model selects a provider as follows:

1. It evaluates the trustworthiness of all the providers in the list. Providers whose trustworthiness cannot be determined (due to no available rating) are placed in the set *NoTrustValue*. The rest, whose trustworthiness has been determined, are placed in the set *HasTrustValue*.

2. There can be up to two options available to the agent:
   (a_1) select the provider with the highest trust value in the set *HasTrustValue*, which according to the trust model is likely to yield the highest UG; and
   (a_2) select a random provider from the set *NoTrustValue*, allowing it to learn about the performance of an unknown provider (i.e. exploring the provider population).

3. Obviously, if the set *HasTrustValue* is empty, it can only choose (a_2); if the set *NoTrustValue* is empty, it can only chose (a_1).

4. Otherwise, it needs to determine which action it should take. Choosing (a_2) allows it to explore more about the provider population although it may risk losing utility if it encounters a bad provider. In contrast, choosing (a_1) can somewhat guarantee the expected UG. However, it may not be the optimal performance the agent can get because it has not learnt enough about the provider population. This *exploit-vs-explore* dilemma is addressed in this work by using a standard Boltzmann exploration strategy [Kaelbling et al., 1996]. Using this strategy, an agent tends to explore its environment first and then gradually move its stance towards exploitation when it learns more about the environment. Thus, the agent chooses an action a_k with the probability of:

$$P(a_k) = \frac{e^{\frac{ER(a_k)}{T}}}{\sum_{a_i} e^{\frac{ER(a_i)}{T}}} \tag{4.1}$$

   where $ER(a_i)$ is the expected return from choosing action a_i, and T is a parameter that is set to decrease over time to decrease exploration (termed the *temperature* parameter in [Kaelbling et al., 1996]). In brief, the probability that an action a_k is selected is biased by the expected return of that action. Moreover, when an agent's level of exploration is decreased (by decreasing T over time) the action with the highest expected return is more likely to be

selected (i.e. the agent is more likely to exploit the knowledge it has learnt about the performance of provider agents). Here, the expected return for (a_1) is the expected UG of the highest trusted provider as calculated from its trust value, and that for (a_2) is the average UG of the provider population that has been observed by the consumer agent.

Having selected a provider, the consumer agent then uses its service and gains some utility from the interaction (i.e. UG). The value of UG is in $[-10, 10]$ and depends on the level of performance of the provider in that interaction. A provider agent can serve many users at a time. As in realistic situations, a consumer agent, however, does not always use the service in every round. The probability it needs and requests the service, called its *activity level* and denoted by α, is selected uniformly randomly when the consumer is created. In other words, the activity level of a consumer determines how frequently it uses the service[6].

After an interaction, the consumer agent rates the service of the provider based on the level of performance, or the quality of the service, it received. It records the rating for future trust evaluations and also informs the provider about the rating it made. The provider may record the rating as evidence about its performance to be presented to potential consumers (as discussed in Section 3.6). Since the basic FIRE, which is going to be evaluated, assumes that all agents exchange their information honestly, an agent (as a witness or as a referee) provides its true ratings as they are without any modification. The testbed is extended to simulate various types of disinformation later in Chapter 6.

So far, there is no difference between provider agents. However, in order to test the ability of a trust model in helping a consumer select good providers, we need to introduce different types of provider agents with various levels of performance. By so doing, the actual UG of a consumer agent from an interaction (which is determined by the performance of the provider it selects) will reflect how good its trust model is in evaluating the trustworthiness (i.e. the expected performance) of the providers. Here, we consider four types of provider agents: good, ordinary, bad, and intermittent. They are to simulate the cases in real world, where, in a particular market, there are usually a (small) number of very good service providers, many ordinary providers who cannot perform exceptionally as those in the first group, and some bad providers that cheat. The intermittent providers

[6]This is to simulate the phenomena that each agent has an individual frequency of requiring a particular service. For example, an ordinary person may need to check the news once a day, while a stock broker may do so once every hour.

Profile	Range of μ_{P}	σ_{P}
Good	[PL_GOOD, PL_PERFECT]	1.0
Ordinary	[PL_OK, PL_GOOD]	2.0
Bad	[PL_WORST, PL_OK]	2.0

Performance level	Utility gained
PL_PERFECT	10
PL_GOOD	5
PL_OK	0
PL_BAD	−5
PL_WORST	−10

TABLE 4.2: Profiles of provider agents.

are introduced to simulate the cases of some online servers whose performance is affected by extraneous factors such as unreliable Internet connections or technical difficulties so that they perform unpredictably, good in some instances and bad in the others.

In the real world, it is rarely the case that one service provider can always maintain a fixed performance. There are usually always some (minor) fluctuations of their performance due to various reasons (e.g. late delivery due to traffic conditions, varying food quality depending on weather). Therefore, it is unreasonable to set a fixed performance level for a provider agent in the testbed. Instead, we only set a provider's mean performance and later vary its actual performance based on a random variable. The normal distribution is chosen for this purpose since it models the random nature that we look for and also allows us to control the variation range of the variable fluctuations (by setting the standard deviation parameter). Hence, good, ordinary, and bad providers are assigned a mean level of performance, denoted by μ_{P}. Its actual performance then follows a normal distribution around this mean. The values of μ_{P} and the associated standard deviation of these types of providers, denoted by σ_{P}, are given in Table 4.2. Intermittent providers, since we want to model intermittent behaviours, on the other hand, are set to yield unpredictable (random) performance levels in the range [PL_BAD, PL_GOOD].

As in our example above, a consumer might experience a better service from a provider in the same country than it does from those that are in a different country. Here, we use the distance between a consumer and a provider on the sphere world to model the particular situations between them. If a consumer agent is situated outside of the provider's normal operational range (i.e. r_o) the service quality of that provider is then set to degrade in proportion to the distance between them.

4.2.2 The dynamism factors

Although the testbed described in the previous section covers the basics of interactions between agents in MAS, it does not reflect the continuously changing nature of an open MAS (as discussed in Section 2.4.2). Therefore, in order to verify that FIRE can cope with various changes that can happen in an open MAS (Requirement **2c**), dynamism is introduced into the testbed by changing a number of its factors after each round[7]:

- *The population of agents*: In an open MAS, agents can come and leave the system at anytime. This is simulated by removing a number of randomly selected agents from the testbed and adding new agents into it. The numbers of agents added and removed after each round vary, but have an upper limit of some predefined percentage of the whole population. The population change limits for the consumer and the provider populations are denoted respectively by p_{CPC} and p_{PPC}. Since in the real world, providers are usually more established than consumers, p_{PPC} is set to be lower than p_{CPC} in our simulations. The profile of the newly added agents are set randomly but they are uniformly distributed over the initial agent populations (i.e. the proportions of providers of different profiles and that of consumers in different groups are maintained) in order to maintain the characteristics of the population in which trust models are being tested.

- *The locations of agents*: During their life cycle, agents break old relationships and make new ones (reflecting the notion of continual change that is inherent in open MAS). In our testbed, this type of change is reflected by the change in an agent's location on the spherical world. When a consumer changes its location, it will have a new set of acquaintances according to its r_o. In addition, the location of an agent in the testbed also reflects its individual situation covering things such as its knowledge about other local agents (see Section 3.5) and the service delivery between providers and consumers (see Section 4.2.1). Therefore, changing an agent's location changes its relationships with others, as well as its individual situation. Specifically, we use polar coordinations (r, φ, θ) for agent locations on the spherical world. Then in order to change an agent's location, amounts of angular changes $\Delta\varphi$ and $\Delta\theta$ are added to φ and θ respectively. In this case, $\Delta\varphi$ and $\Delta\theta$ are selected randomly in $[-\Delta\phi, +\Delta\phi]$. Thus, $\Delta\phi$ limits the variability of agents'

[7]These factors are chosen based on the types of changes that were identified in Section 2.4.2.

locations. Not every agent changes its locations every round and, in particular, p_{CLC} and p_{PLC} are used to denote the probabilities that a given consumer or provider, respectively, changes its location in a round.

- *The behaviour of the providers*: In many environments, provider performance may alter (for better or worse) over time. A provider may even change its behaviour completely (e.g. a provider may take advantage of its good reputation and decide to perform selfishly to obtain better utility). In our testbed, the average performance of a provider (μ) can be changed by an amount of $\Delta\mu$ randomly selected in $[-\mathsf{M}, +\mathsf{M}]$, and this happens in each round with the probability of $p_{\mu\mathsf{C}}$. Moreover, after each round, a provider can switch to a completely new provider profile with a probability of $p_{\mathsf{ProfileSwitch}}$.

The above changes to the testbed's environment are applied only after each round of interactions finishes. The nature and degree of dynamism vary depending on the experiment and are therefore specified for each specific experiment. Now, in some experiments where, because of their objectives, none of the above changes is carried out, we call the testbed *static*.

4.3 The Experimental Setup

In each experiment, the testbed is populated with provider and consumer agents. Each consumer agent is equipped with a particular trust model, which helps it select a provider when it needs to use a service. Since the only difference among consumer agents is the trust models that they use, the utility gained by each agent through simulations will reflect the performance of its trust model in selecting reliable providers for interactions. Therefore, the testbed records the UG of each interaction along with the trust model used.

Before each experiment, the testbed is set up to simulate a particular environment of interest using the parameters defined in the previous sections. These parameters are called the *experimental variables* and their default values are presented in Table 4.3. These default values will be used in all the experiments unless otherwise specified. Although a 'typical' provider population may differ in various applications, the space of possibilities is vast and exploring it completely would be impossible. Therefore, we choose provider populations which we believe are more common than others for our experiments Here, a typical provider population according to our view consists of about half profitable providers (i.e. yielding

Simulation variable	Symbol	Value
Number of simulation rounds	N	500
Total number of provider agents:	$\mathsf{N_P}$	100
+ Good providers	$\mathsf{N_{PG}}$	10
+ Ordinary providers	$\mathsf{N_{PO}}$	40
+ Intermittent providers	$\mathsf{N_{PI}}$	5
+ Bad providers	$\mathsf{N_{PB}}$	45
Number of consumer agents in each group	$\mathsf{N_C}$	500
Range of consumer activity level	α	$[0.10, 1.00]$

TABLE 4.3: Experimental variables.

Parameters	Symbol	Value
Local rating history size	H	10
IT recency scaling factor	λ	$-\frac{5}{\ln(0.5)}$
Branching factor	n_{BF}	2
Referral length threshold	n_{RL}	5
Component coefficients:		
+ Interaction trust	W_{I}	1.0
+ Role-base trust	W_{R}	1.0
+ Witness reputation	W_{W}	0.5
+ Certified reputation	W_{C}	0.25
Reliability function parameters:		
+ Interaction trust	γ_{I}	$-\ln(0.5)$
+ Role-base trust	γ_{R}	$-\ln(0.5)$
+ Witness reputation	γ_{W}	$-\ln(0.5)$
+ Certified reputation	γ_{C}	$-\ln(0.5)$

TABLE 4.4: FIRE's default parameters.

positive UG) and half exploiting providers (i.e. yielding negative UG, including intermittent providers). However, good and intermittent providers are usually exceptional cases and, thus, they take only a small portion of each half. Except in the experiments where we evaluate FIRE with different provider populations, this typical provider population is used throughout[8].

Here, we also show the default parameters of FIRE set for the experiments in Table 4.4. These parameters were introduced to allow end users of FIRE to customise FIRE's behaviour according to their own needs and application. For example, here, we know that the IT component deduces trust from ratings in which the agent does the rating itself and, thus, is more reliable than the WR and CR components, which use information from third-parties. We expect the CR information to exaggerate an agent's true performance, hence, the CR component has

[8]We have explored with different population mixes and we observe the same broad trends.

the lowest reliability. The RT component provides rules encoding knowledge and beliefs about the agent's environment to customise the trust model, so it should also have a high reliability. Therefore, the component coefficients are set to reflect these beliefs. Likewise, since the performance of an agent may change quickly from round to round and given the time unit used in the test bed (round of interactions), we set the IT recency scaling factor such that a 5-round old rating has half (0.5) the effect of a new rating (1.0). It should be noted that the space of possible parameter assignments is vast and comprehensively evaluating FIRE in that space is impossible. Therefore, these parameters are chosen here on a reasonable basis as explained above. In this regard, Chapter 7 investigates a number of learning techniques so that some of these parameters can be adjusted dynamically according to an agent's actual situation.

4.4 Summary

This chapter has presented our evaluation methodology which is followed throughout in evaluating FIRE. The two-sample t-test, a hypothesis testing method, is used to ensure that our conclusions about the evaluation results are correct with a minimum confidence level of 95%. We also show in detail how our testbed is constructed and set up to reflect the main features of open MAS as identified in Chapter 1. The testbed and the evaluation methodology will be used in the subsequent chapters (Chapters 5, 6, and 7) to run and analyse various experiments on FIRE's performance.

Chapter 5

Empirical Evaluation

Having presented the testbed and the methodology for FIRE's evaluation in the previous chapter, we now turn to the experiments themselves. In particular, we concentrate on the two following questions:

1. How much is the benefit (in terms of UG) of using FIRE for selecting interaction partners compared to not using a trust model and to other models?
2. How do FIRE's individual components contribute to its overall performance?

The experiments in this chapter are designed to give the answers for the two questions in a variety of environment types (e.g. static environments where there is no significant dynamism and dynamic environments where there are changes in various factors in the testbed as described in Section 4.2.2). Experiments dealing with the first question are presented in Sections 5.1 and 5.2. The former shows experiments with static environments, while the latter focusses on dynamic ones. Subsequently, Section 5.3 presents experiments dealing with the second question. A summary is then provided in Section 5.4

5.1 Performance in Static Environments

In order to evaluate the overall performance of FIRE, we compare it with the SPORAS model (whose operation is described in Appendix A.1) and a group of agents with no trust model. Hence, there are three groups of consumer agents: FIRE, SPORAS, and NoTrust. SPORAS is chosen as the control benchmark for two

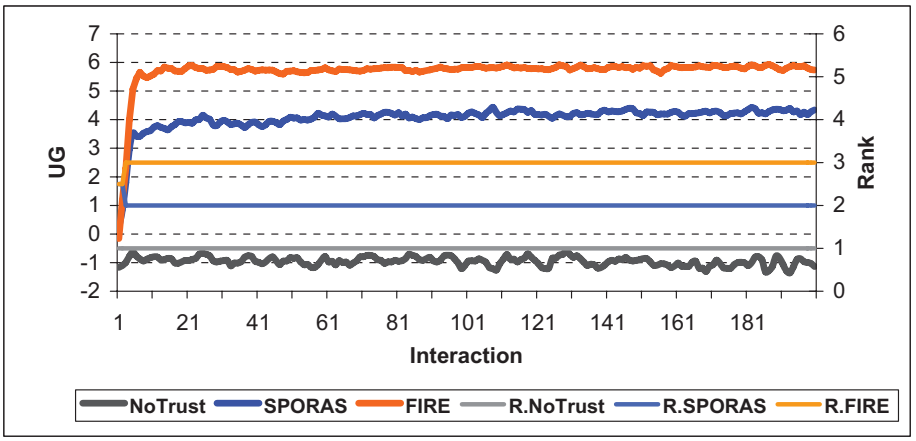

FIGURE 5.1: Overall performance of FIRE in the typical provider population.

reasons. First, it is a successful independently developed trust model that several other researchers have used for benchmarking (e.g. [Carbo et al., 2003], [Sabater and Sierra, 2001]). Second, other than SPORAS, most other notable trust models make assumptions that are incompatible with open MAS, or require additional knowledge, and, thus, they will not operate as intended in our testbed.

Now, the first thing to test is whether FIRE helps consumer agents select profitable providers (i.e. those yielding positive UG) from the population and, by so doing, helps them gain better utility than without FIRE (i.e. the NoTrust group). In this section, the testbed's environment is static (as defined in Section 4.2.1).

In more detail, Figure 5.1 shows that the NoTrust group, selecting providers randomly without any trust evaluation, performs consistently the lowest (as we would expect). On the other hand, both SPORAS and FIRE prove to be beneficial to consumer agents, helping them to obtain significantly higher UG. This shows that the tested trust models can learn about the provider population, and allow their agents to select profitable providers for interactions. However, the chart, as well as the t-test ranking, also shows that FIRE outperforms SPORAS, the second rank, throughout the interactions by about 40%[1]. This is despite the fact that SPORAS, being a centralised model, gathers much more information than FIRE (a decentralised model)[2]. The performance difference of FIRE and SPORAS is ac-

[1]Here, the average UG of the NoTrust group in an experiment (i.e. -1) is used as the baseline performance. The UG of SPORAS and FIRE is then compared to this baseline performance to show the benefit of using SPORAS and FIRE.

[2]After every interaction, the consumer reports its rating about the provider's service in that interaction to SPORAS. Therefore, as a centralised service, SPORAS collects all the available ratings from its users. In contrast, consumers employing FIRE only have ratings from a limited set of witnesses (from the WR component) and those presented by providers (from the CR component) in addition to their own ratings. Typically in our experiments, after the first 10 rounds the average number of ratings taken into account in each trust evaluation request to

Interaction:	1	2	3	4	5	6	7	8	9	10
SPORAS	0.20	0.85	1.80	2.96	3.53	3.42	3.42	3.52	3.58	3.62
FIRE	−0.16	1.20	2.30	4.00	5.06	5.44	5.66	5.52	5.47	5.53

TABLE 5.1: The performance of SPORAS and FIRE in the first 10 interactions

counted for by the fact that FIRE separates direct experiences from others' experiences (i.e. ratings) in trust evaluation, while SPORAS treats all types of ratings equally. Therefore, SPORAS suffers from noise in ratings (resulting from different degrees of degradation of service quality due to different provider-consumer distances). In contrast, FIRE reduces rating noise by giving more weight to direct experiences (see Table 4.4), which are more relevant to an individual agent's situation. Another noticeable point is that in the first few interactions, FIRE can learn about the providers quicker than SPORAS as the FIRE group achieves its superiority from the first interaction (see Table 5.1) despite much less information being available to it. As we show in Section 5.3, this is achieved thanks to the WR and, in particular, the CR components.

Having shown FIRE performs well in a mixed population, we now check whether FIRE performs consistently with a particular type of providers. Therefore, we re-ran the same experiment but with provider populations consisting of providers of only one profile (e.g. all good, all ordinary, all bad, and all intermittent). Specifically, the experiment is run with 100 good providers, then with 100 ordinary providers, 100 bad providers, and 100 intermittent providers. The result in the case of intermittent providers is not shown here because the performance of all three groups is indistinguishable; fluctuating randomly in $[-1, 0]$ (this is expected because of the random behaviour of intermittent providers). In the rest of the experiments, we observe similar results (see Figures 5.2, 5.3, and 5.4) to that in our first experiment with the typical provider population. FIRE maintains its superiority in all the three types of provider population. This shows that FIRE can work well in a wide range of provider population.

In sum, through the experiments on FIRE's overall performance, we confirm that FIRE is beneficial to agents in selecting interaction partners in a variety of types of provider populations. In addition, despite being decentralised and having less information than SPORAS, FIRE outperforms the model in all the cases thanks to its differential treatment of each source of trust information.

FIRE is 3.28, and that to SPORAS is 15.55. After 20 rounds the corresponding numbers are 4.05 and 28.47 respectively. This suggests that FIRE may be advantageous in environments in which rating information costs some premium.

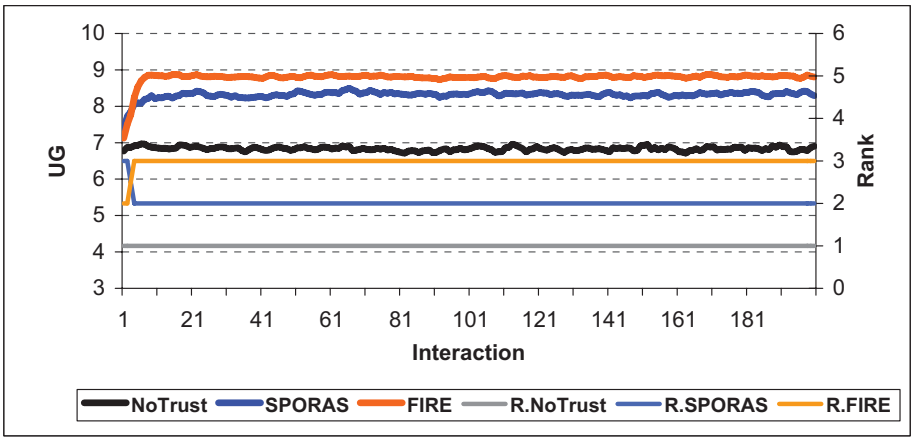

FIGURE 5.2: Overall performance of FIRE – 100% good providers.

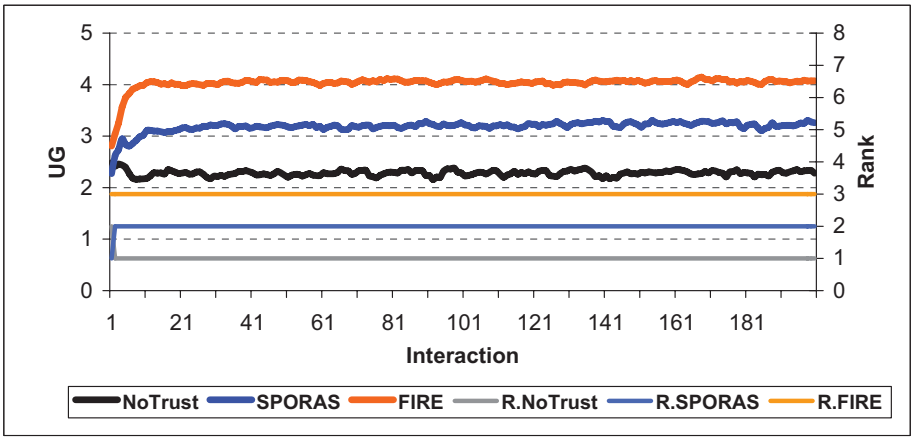

FIGURE 5.3: Overall performance of FIRE – 100% ordinary providers.

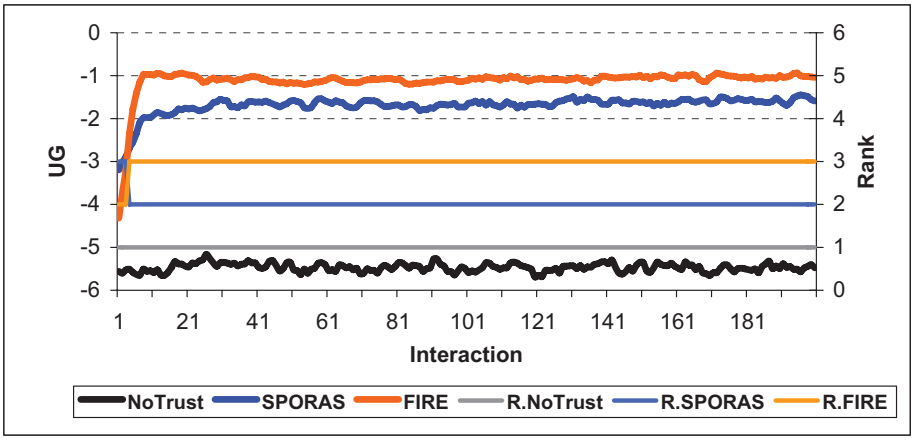

FIGURE 5.4: Overall performance of FIRE – 100% bad providers.

5.2 Performance in Dynamic Environments

The environment of a realistic open MAS is always changing because of its openness (as discussed in Sections 1.1 and 4.2.2). Hence, a trust model designed for open MAS should be able to function effectively in such a dynamic environment. Given this, this section concentrates on testing the hypothesis that FIRE still maintains its desirable properties (i.e. being beneficial to agents in selecting interaction partners) in a changing environment. Similarly to the experiments in Section 5.1, there are three groups of consumer agents in the experiments: NoTrust, SPORAS, and FIRE. The provider population is the typical one. Each experiment will test the hypothesis with only one dynamic factor in effect (see Section 4.2). Specifically, the same experiments will be run but with each of the following conditions[3]:

1. The provider population changes at maximum 2% every round ($p_{\mathsf{PPC}} = 0.02$).
2. The consumer population changes at maximum 5% every round ($p_{\mathsf{CPC}} = 0.05$).
3. A provider may alter its average level of performance at maximum 1.0 UG unit with a probability of 0.10 each round ($p_{\mu\mathsf{C}} = 0.10$, $\mathsf{M} = 1.0$).
4. A provider may switch into a different (performance) profile with a probability of 0.02 each round ($p_{\mathsf{ProfileSwitch}} = 0.02$).
5. A provider may move to a new location on the spherical world at a maximum angular distance of $\frac{\pi}{20}$ with a probability of 0.10 each round ($p_{\mathsf{PLC}} = 0.10$, $\Delta\phi = \frac{\pi}{20}$).
6. A consumer may move to a new location on the spherical world at a maximum angular distance of $\frac{\pi}{20}$ with a probability of 0.10 each round ($p_{\mathsf{CLC}} = 0.10$, $\Delta\phi = \frac{\pi}{20}$.

These experiments are named Experiment 1 to 6, respectively, and their results are shown in Figures 5.5 to 5.10. Since the NoTrust group still has the lowest performance, we omit its results from the charts for reasons of clarity. A general observation from all the results is that both FIRE and SPORAS still maintain

[3]These are what we consider to be reasonable values of variation. We have conducted similar experiments with both greater and lesser degrees of dynamism and we see the same broad trends as we report here.

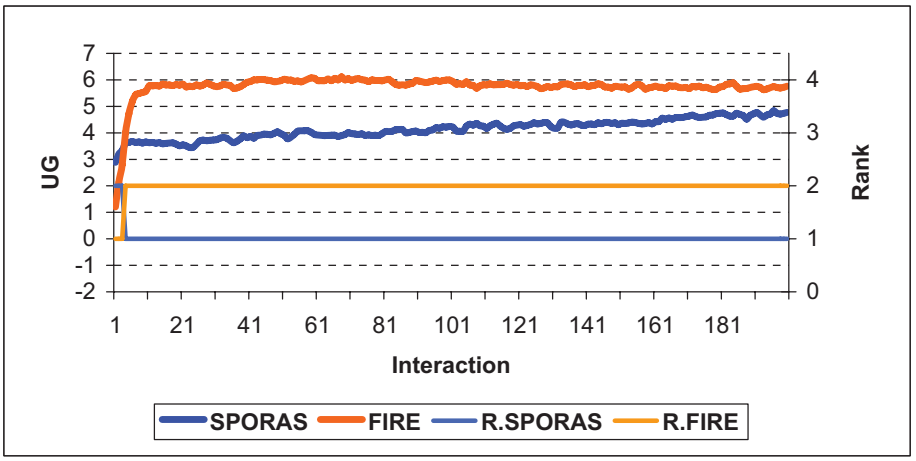

FIGURE 5.5: Experiment 1: Provider population change: $p_{\mathsf{PPC}} = 0.02$.

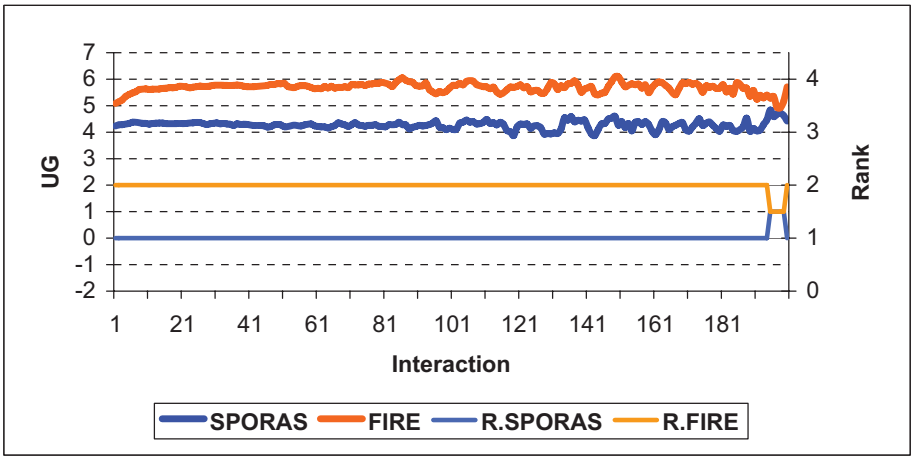

FIGURE 5.6: Experiment 2: Consumer population change: $p_{\mathsf{CPC}} = 0.05$.

positive UG from about 3.0 to 6.0 (except SPORAS in Experiment 4, Figure 5.8). However, dynamism, as it introduces noise to the environments, adversely affects the performance of both of them in all the experiments reported here. Specifically, and as we would expect, their performance is lower than that in the static environment (Figure 5.1) and the performance plots also evolve differently over time. Nevertheless, although having lower levels of performance than in a static environment, the shape of FIRE's performance plots are generally maintained after they reach their stable level in the first few interactions in all the experiments. This shows that FIRE is able to maintain a stable performance regardless of the various types of changes in the environment. In other words, FIRE can learn about the changes and adapt quickly to them.

In more detail, the experiments here can be put into two categories: (i) dynamism on the consumer side (Experiments 2 and 6), and (ii) dynamism on the provider side (Experiments 1, 3, 4, 5). In the first group, the results (Figures 5.6 and 5.10)

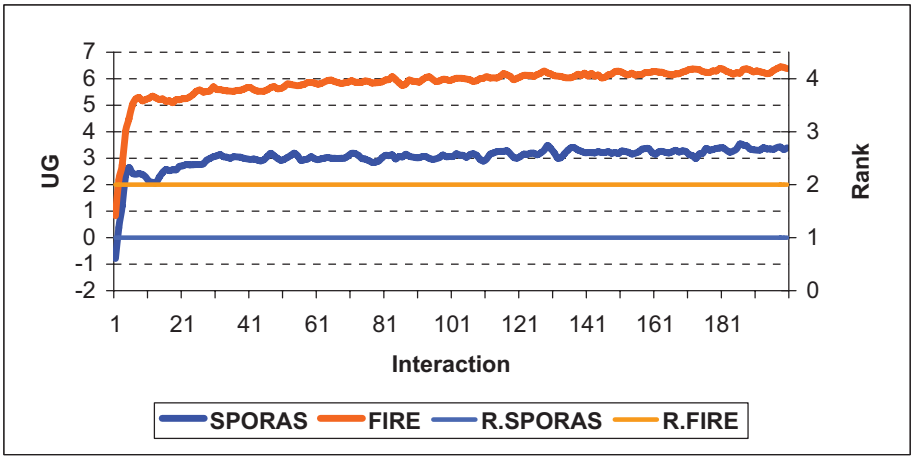

FIGURE 5.7: Experiment 3: Providers change their performance: $p_{\mu\mathsf{C}} = 0.10$, $\mathsf{M} = 1.0$.

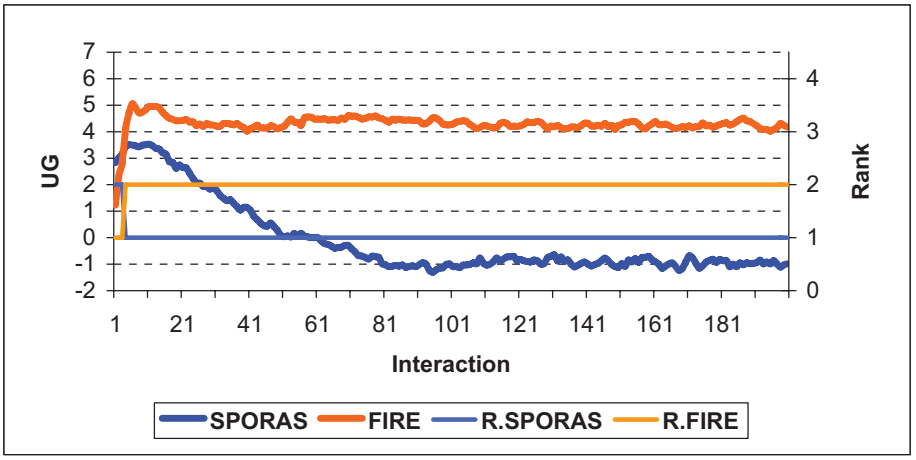

FIGURE 5.8: Experiment 4: Providers switch their profiles: $p_{\mathsf{ProfileSwitch}} = 0.05$.

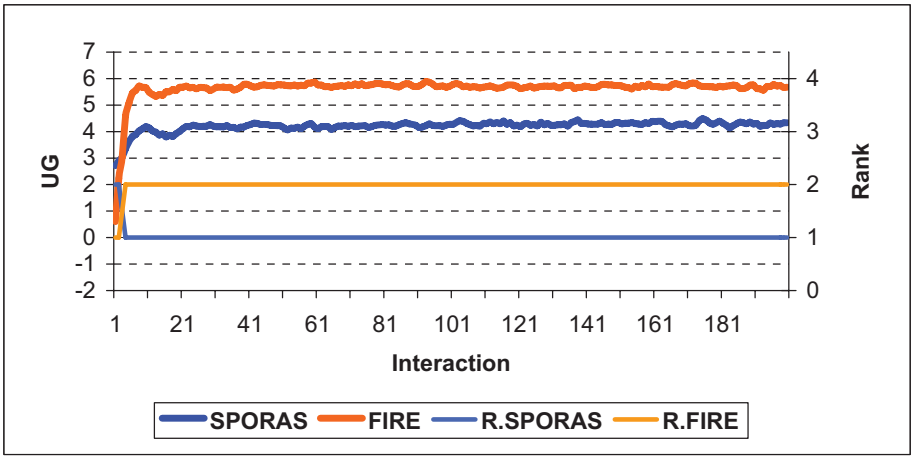

FIGURE 5.9: Experiment 5: Providers change their locations: $p_{\mathsf{PLC}} = 0.10$, $\Delta\phi = \frac{\pi}{20}$.

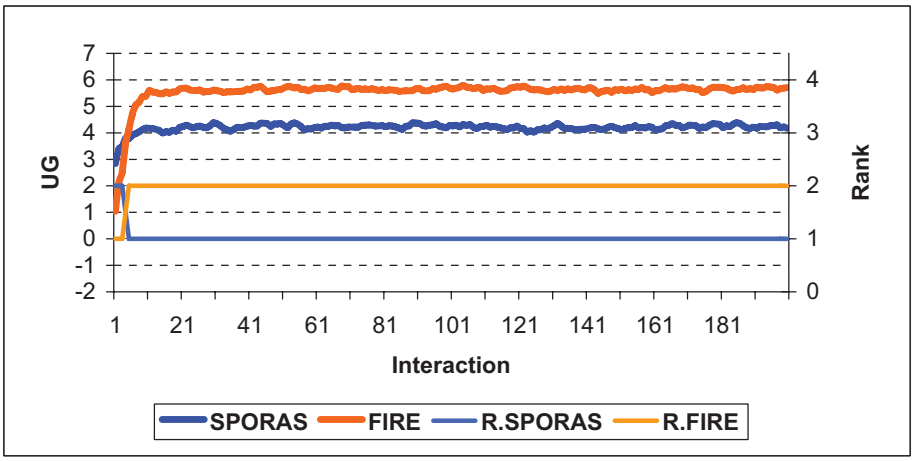

FIGURE 5.10: Experiment 6: Consumers change their locations: $p_{\mathsf{CLC}} = 0.10$, $\Delta\phi = \frac{\pi}{20}$.

show that SPORAS can cope well with these types of changes. This is because SPORAS collects ratings centrally from all consumers and, thus, small changes on the consumer side do not have a significant impact on its performance as its learned knowledge about the provider population is still useful. Particularly in Experiment 2, where new consumers are added to the testbed, newly joined agents using SPORAS take advantage of the existing knowledge of the centralised model and perform well right from the start. In contrast, FIRE relies on the consumer community for witness reputation and, thus, has a slightly lower performance than that in the static environment. However, it still outperforms SPORAS in these experiments.

The situation is somewhat different in the experiments of the second category. SPORAS's performance is significantly affected when providers change their performance levels (Experiments 3 and 4, whose results are shown in Figures 5.7 and 5.8), most noticeably in Experiment 4, where providers switch their performance profiles completely. In this experiment, although FIRE is also affected greatly by the steep changes in the provider population, it still maintains a generally high and stable performance, while SPORAS's performance degrades disproportionately to that of the NoTrust group. It should be noted that the trust models' duty here is to learn and predict the behaviour (i.e. the performance) of providers and, therefore, their performance in an environment where there are changes on the provider side reflects their ability to adapt to such changes. Hence, the results suggest that SPORAS can quickly learn about the environment (because of its centralised nature), but has difficulty adapting to the continual changes of the providers. In Experiments 1 and 5 (Figures 5.5 and 5.9), where the provider population changes and where the providers move around, respectively, the performance of FIRE and

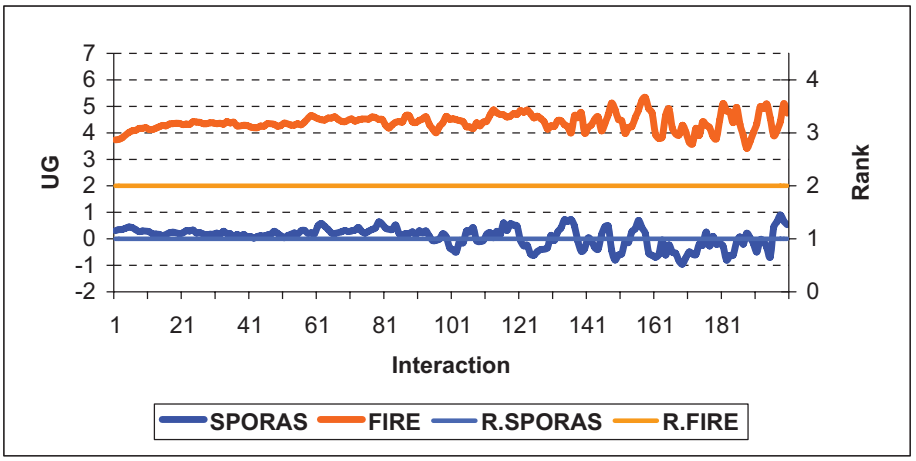

FIGURE 5.11: Experiment 7: Performance of FIRE in an environment where all dynamic factors are in effect.

SPORAS are only slightly affected. In general, FIRE performs consistently in all these experiments. Its average UG is stable around 6.0 in Experiments 1, 3, and 5; and around 4.0 in Experiment 4 (which has the most abrupt changes). This confirms our intuition that the recency function of FIRE helps it adapt quickly to changes in the environment.

Since a realistic open MAS usually has a combination of all the dynamic factors considered here, we also want to test how FIRE performs in such situations. Therefore, we ran an additional experiment with all the dynamic factors active at the same time. The result, in Figure 5.11, shows that although both FIRE and SPORAS suffer from the continual changes of various types in the testbed's environment, FIRE manages to maintain a rather stable performance in the range $[4.0, 5.0]$. SPORAS, however, cannot cope with all the types of changes at the same time and its performance degrades significantly compared to its own performance in a static environment and is near to that of the NoTrust group through all interactions (see Figure 5.1). This again confirms the adaptability of FIRE in a complex dynamic environment.

In summary, the experiments in this section show that FIRE is able to perform consistently in various dynamic environments, maintaining a high level of utility gain for its agents (in all the experiments).

5.3 Impact of the Individual Components

In Section 3.1, we argued that each component of FIRE plays an important role in exploiting trust information from a particular source and this, in turn, contributes to the effectiveness of the overall model. In order to confirm this, here we benchmark FIRE with and without various components to evaluate the contribution of that component to the whole model. However, since the IT component is mostly reused from Regret, we will focus on evaluating the novel components (i.e. WR and CR). Role-based trust is also not considered here because it is typically highly domain specific. Experiments in this section evaluate the WR and CR components with the typical provider population in a static environment.

First, we benchmark the WR component. In this experiment, there are two groups of consumer agents. The first one uses only the IT component[4] (called the Control group). The second makes use of the WR component in addition to the IT component (called the WR group). This experiment's hypothesis is that the performance of the WR should be higher than that of the Control group.

The result of the experiment, presented in Figure 5.12, shows that the WR component does substantially improve the performance of agents in the WR group compared to that of those in the Control group. The t-test ranking confirms this by showing that agents using the WR component outperform agents using only the IT component in all interactions, and, effectively, also verifies our hypothesis. More importantly, however, the WR group achieves its higher performance quicker than the Control group. This means that WR speeds up an agent's learning about its environment by propagating trust in the agent's community (here the community of consumer agents is connected to one another via acquaintances).

In the next experiment we evaluate the CR component (using a similar setting). Here, there are two groups of consumer agents. The Control group employs the IT component, and the other employs the CR component in addition (called the CR group). The hypothesis in this experiment is that the CR group, which additionally makes use of the CR component, should outperform the Control group. The result presented in Figure 5.13 shows a similar result to that of the previous experiment.

[4]It should be noted that if Regret is employed in our testbed, its performance will be similar to that of FIRE's IT component only. This is due to the fact that there is no information about a social network of the agents in the testbed available for Regret. Therefore, other than the direct trust component, the other components of Regret will not be able to work due to a lack of supporting information (e.g. its witness reputation component will not be able to locate witnesses, and the neighbourhood reputation component will not be able to locate neighbouring agents of the target agent).

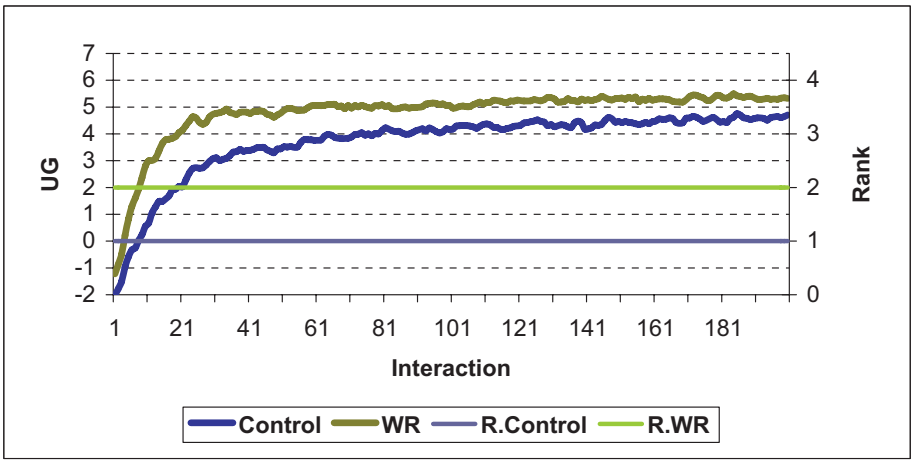

FIGURE 5.12: Performance of the WR component.

Hence, all of the claims made above for the WR component are still valid for the CR component. However, the most noticeable thing about this experiment is its execution time. Without having to look for witnesses as in the process of WR, the process of CR is more direct, resulting in an execution time for this experiment being about 15 times faster than that of the previous one. This confirms our intuition about the high serviceability of the CR component.

A subtler point shown in this experiment is the quick learning time of the CR group. Comparing the first interactions of the WR group in Figure 5.12 with those of the CR group, Figure 5.13 shows that the CR group starts off better than the WR group right from the first few interactions. In order to verify this observation, eliminating the random factor that may affect the results of the two independent experiments above, we ran another experiment to compare the performance of FIRE with and without the CR component. In this experiment, there are also two groups of consumer agents: the WR group employing the IT and WR components and the FIRE group employing the IT, WR, and CR component. Our hypothesis is that the addition of the CR component to FIRE is beneficial, or, equivalently, the FIRE group should outperform the WR group. The result in Figure 5.14 shows that the performance of FIRE is indeed always higher than that of the WR group. In more detail, Table 5.2 shows the FIRE group's performance reaches its stable level of around 6.0 in only 8 interactions, while that of the WR group only reaches 2.57 after 10 interactions. This shows that the CR process propagates trust in an agent community more quickly than the WR process. Taking into consideration its very quick execution time, compared to that of the WR component in the previous experiments, the CR component is clearly useful in situations where an agent needs to have a quick trust evaluation in order to expedite decisions, or when

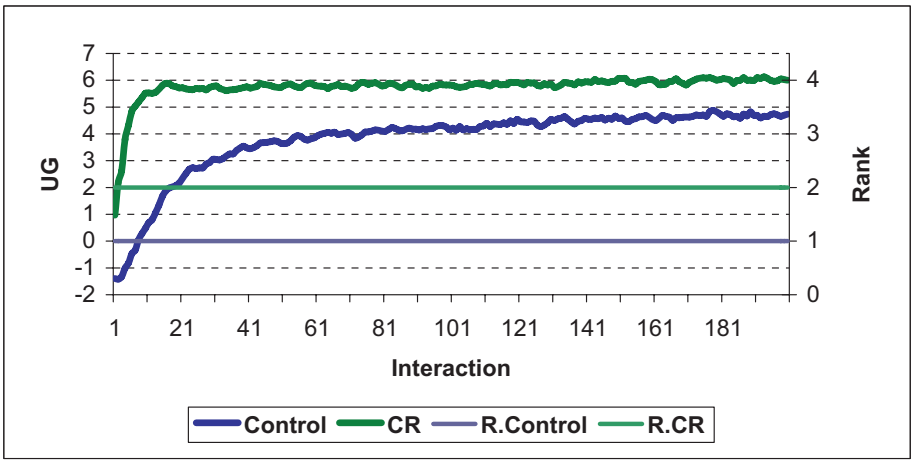

FIGURE 5.13: Performance of the CR component.

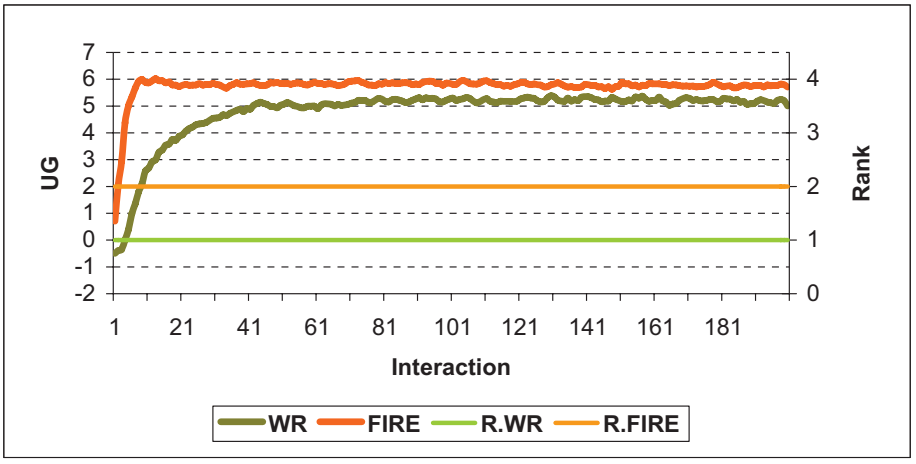

FIGURE 5.14: Performance of FIRE with and without the CR component.

WR information is scarce and difficult to locate. We conclude that the addition of the CR component to FIRE is beneficial both in terms of its robustness (reflected by its higher level of performance) and its serviceability.

In summary, it has been shown that taking various sources of trust information into account not only helps FIRE be able to make trust evaluations in a wide variety of situations, but also increases its usefulness; and that both the WR and CR components contribute a significant amount to its overall performance.

Interaction:	1	2	3	4	5	6	7	8	9	10
WR	−0.50	−0.39	−0.35	0.01	0.40	0.99	1.36	1.80	2.10	2.57
CR	0.71	2.08	2.94	4.38	5.02	5.34	5.67	5.92	5.99	5.89

TABLE 5.2: The performance of WR and FIRE in the first 10 interactions

5.4 Summary

This chapter has presented various experiments to investigate the behaviour of FIRE. These experiments were designed to assess the effectiveness of FIRE in a wide range of situations. Specifically, we showed that FIRE helps its users to select appropriate partners for interactions. In terms of utility gain, agents using FIRE outperform those using SPORAS, a centralised model, despite the fact that SPORAS collects more information than FIRE. This result remains even when FIRE is used in different provider populations. More importantly, we showed that FIRE copes well with various changes that are typically present in a realistic MAS. It is able to learn changes in the environment quickly and to maintain a stable and high performance.

In addition, we showed that the WR and CR components contribute highly to FIRE's performance and their mechanisms allow trust information to be propagated quickly in an agent society. Particularly, FIRE with the CR component achieves a significantly higher level of performance than without it. Moreover, CR also speeds up FIRE's bootstrapping and has a very low running cost (in terms of time and resources required for trust evaluations).

Overall, it was shown that FIRE is a robust trust model and that all its components contribute significantly to its performance. However, in the experiments reported in this chapter we assumed that all agents are honest in exchanging information, which is not realistic in open MAS (as noted in our requirements in Section 2.5). In the next phase of our research, we extend FIRE with the ability to detect lying and inaccuracy in third-party ratings. This allows FIRE to filter out inaccurate ratings when evaluating trust based on WR and CR. The details of this are given in the next chapter. Subsequently, Chapter 7 shows how learning techniques can help to adapt FIRE's parameters to an agent's own situation.

Chapter 6

Handling Inaccurate Reports

As we have shown in the previous chapter, third-party reports (i.e. witness reports and certified ratings) propagate trust information in an agent society. They are particularly useful in speeding up an agent's learning about its peers' behaviours and, in so doing, they help it gain high utility by choosing to interact with appropriate partners despite its lack of direct experiences. However, a key problem in this area, and one that is exacerbated in open MAS, is that these reports can be inaccurate. This can happen because of the differing views of the reporters (e.g. 'on-time good delivery' may indicate an 'excellent service' for one person but may only indicate a 'satisfactory service' to another, close business partners may receive preferential services compared the normal services that other ordinary customers receive). However, it can also happen due to the conflicting interests that stem from the fact that there are multiple stakeholders (e.g. some reporters may deliberately provide false information to serve their own interests). In both cases, third-party reports that differ from the actual performance an agent receives are viewed as inaccurate and their inaccuracy[1] is reflected by the magnitude of the differences. Now, since these reports are central building blocks for reputation systems, if their inaccuracy is not recognised and dealt with, it will adversely affect the function of these systems. Worse still, they may become a means for malicious agents to gain unwarranted trust which may then allow them to benefit to the detriment of others.

Given its importance, there have been several attempts to tackle the inaccurate

[1]It should be noted that, in this context, inaccuracy is according to the view of the agent receiving third-party reports. It does not reflect the true honesty/accuracy of a reporting agent. Rather it should be viewed as the subjective measure of the usefulness of third-party information provided by that agent which is assessed by another particular agent.

reports problem (see Section 2.4.3), but none of them are well suited to open MAS. In particular, in order to operate as intended, they typically make assumptions about the target environment that are incompatible with the characteristics of open MAS or they require additional domain knowledge that clearly limits their applicability (see Section 2.4.3 for examples). To this end, we devise a credibility model that can be used by trust and reputation models in open MAS to assess the accuracy of a reporter. In so doing, we extend FIRE to deal with situations where agents may provide inaccurate reports. Against this background, this chapter describes our credibility model (Section 6.1) and provides an empirical evaluation of its performance (Section 6.2).

6.1 Credibility Model

The credibility of a witness or a referee in reporting its ratings about another agent can be derived from a number of sources. These include knowledge about:

- the relationships between the reporter and the rated agent (e.g. cooperating partners may exaggerate each other's performance, competing agents may underrate their opponents, no relationship may imply impartial ratings);
- the reputation of the reporter for being honest and expert in the field in which it is doing the rating (e.g. a reputable and independent financial consultant should provide fair ratings about the services of various banks);
- the relationships between the reporter and the querying agent (e.g. agents with the same owner should provide honest reports to one another);
- norms in the reporter's society (e.g. doctors usually recommend a drug to the benefits of patients, rather than to the benefit of its pharmaceutical companies).

Unfortunately, however, these types of information are very application specific and may not be readily available in many cases. Therefore, although they could certainly be used to enhance the performance of a credibility measure, they are not suitable as a generic basis (although they could complement a generic measure in particular contexts).

In contrast, in our credibility model, we view providing third-party reports as a service an agent provides. Thus its performance (i.e. trustworthiness and reliability) can be evaluated and predicted by a trust model. By so doing, the credibility model can benefit from a trust model's (usually sophisticated) ability of learning and predicting an agent's behaviour (in this case, the behaviour of providing accurate reports) without having to implement its own method. Here, we use FIRE's IT component for this purpose[2]. Although witness reports in WR and references in CR are both third-party reports and their accuracy can be modelled in the same way, an agent can take the roles of a witness and a referee at the same time and behave differently in each role. This is because witness reports in WR are only revealed to a selected set of agents, while references in CR are public as the refereed agent can give its references to any other agents. Knowing this, an agent may have different lying strategies in each role. For example, it may lie more when providing witness reports than when providing certified references because there is more chance that it can get away with lying (since significantly fewer agents receive its witness reports). Therefore, we consider that it is necessary to evaluate the credibility of an agent in terms of providing witness reports (called *witness credibility*) and that in terms of providing references (called *referee credibility*) separately. However, since we use the same credibility model for both these tasks, for the sake of simplicity, henceforth, we shall only describe the evaluation of witness credibility and how WR can make use of it in this section. The same procedure is applied for referee credibility.

In more detail, after having an interaction with agent b, agent a records its rating about b's performance, denoted by r_a ($r_a = (a, b, i_a, c, v_a)$). Now, if agent a previously received a witness report from agent w about b, it then rates the credibility of w by comparing the actual performance of b (i.e. v_a) with w's rating about b. The smaller the difference between the two rating values, the higher agent w is rated in terms of providing witness reports (mutatis mutandis for bigger differences). For each rating that a received from w in evaluating the trustworthiness of b (denoted by $r_k = (w, b, i_k, c, v_k)$), the credibility rating value v_w for agent w is given in the following formula:

$$v_w = \begin{cases} 1 - |v_k - v_a| & \text{if } |v_k - v_a| < \iota \\ -1 & \text{if } |v_k - v_a| \geq \iota \end{cases} \tag{6.1}$$

[2]It should be noted that the credibility model is not necessarily limited to just using IT only. Other components of FIRE could equally well be used for the same purpose. For example, suppose that an agent knows that the reporter belongs to its owner, it can use rules to give a high credibility to that reporter; or if a witness is known to be reliable, witness reports about the reporter can also be taken into account.

where ι is the inaccuracy tolerance threshold ($0 \leq \iota \leq 2$, 2 is the maximal difference since $v_k, v_a \in [-1, 1]$). Thus if the difference between a witness rating value and the actual performance is higher than ι, the witness is considered to be inaccurate or lying, and, therefore, receives a negative rating of -1 for its credibility. On the other hand, if the difference is within the tolerance threshold, it can be viewed as resulting from a subjective viewpoint and is deemed acceptable. In this case, the credibility rating value v_w is set to be inversely varied to the difference (e.g. higher difference, lower credibility). Since $0 \leq |v_n - v_a| \leq 2$, v_w is also in the range $[-1, 1]$ regardless of ι. The rating about w's credibility — $r_w = (a, w, i_w, \text{term}_{\mathsf{WCr}}, v_w)$ — is then recorded in a's rating database, where $\text{term}_{\mathsf{WCr}}$ is the rating term of providing witness reports and i_w is the interaction of agent w providing agent a the rating r_k about agent b.

Here, as agents whose inaccuracy exceeds the tolerance threshold are considered lying and are heavily fined (by giving a -1 credibility rating), honest witnesses may be falsely penalised if their (honest) ratings are too different from that of a (e.g some agents may have substantially varying levels of performance which result in varying, though honest, ratings). However, since in the case of acceptable inaccuracy ($|v_k - v_a| < \iota$) an agent's credibility is also penalised according to the degree of its inaccuracy (i.e. $|v_k - v_a|$), it is always safe to set ι to higher values to reduce the probability of falsely classifying honest witnesses. Nevertheless, doing so inevitably allows ratings from marginally lying reporters be accepted. In such cases, although the credibility of such witnesses may be low, it may never be low enough, or it may take a longer time, for them to be considered lying and be disregarded (see below). This means, in general, that there will be a lower performance of the credibility model. Therefore, it is important to choose a threshold value that closely reflects the variability of the agents' performance in the target environment and the relative costs of considering lying witnesses versus falsely classifying them. To this end, Section 7.3 proposes a technique to automatically tune ι according to the agents' performance deviation.

Having recorded ratings about w's performance on providing witness reports, a can evaluate w's credibility based on those ratings. As mentioned above, a can use its own trust model for so doing. Specifically, here we evaluate a's trust on w's capability of providing accurate reports using the IT component to calculate the credibility trust value of w from the set of ratings about w's witness credibility (called $\mathcal{R}_{\mathsf{I}}(a, w, \text{term}_{\mathsf{WCr}})$, which is retrieved from the rating database as per Section 3.3). If no such ratings have been recorded, and, thus, the IT value is not available, w will be assigned the default witness credibility trust value, denoted

by $\mathcal{T}_{\mathsf{DWCr}}$.

$$\mathcal{T}_{\mathsf{WCr}}(a,w) = \begin{cases} \mathcal{T}_{\mathsf{I}}(a,w,\text{term}_{\mathsf{WCr}}) & \text{if } \mathcal{R}_{\mathsf{I}}(a,w,\text{term}_{\mathsf{WCr}}) \neq \emptyset \\ \mathcal{T}_{\mathsf{DWCr}} & \text{otherwise} \end{cases} \tag{6.2}$$

where $\mathcal{T}_{\mathsf{WCr}}(a,w)$ is the witness credibility of w evaluated by a.

Having defined the credibility measure for witnesses, we extend the WR component to weight each witness rating received according to the credibility of the witness. Suppose that agent a is evaluating the WR of agent b and that the rating r_i is collected from witness w, then the weight for r_i is obtained as follows:

$$\omega_{\mathsf{W}}(r_i) = \begin{cases} 0 & \text{if } \mathcal{T}_{\mathsf{WCr}}(a,w) \leq 0 \\ \mathcal{T}_{\mathsf{WCr}}(a,w) \cdot \omega_{\mathsf{I}}(r_i) & \text{otherwise} \end{cases} \tag{6.3}$$

This means the new rating weight function disregards any rating provided by witnesses that have negative credibility (by giving 0 as the weight for their ratings). The rest are taken into account in producing the WR of b, but are weighted by the credibility of their providers and by their recency (provided by the function $\omega_{\mathsf{I}}(r_i)$ of the IT component). In so doing, ratings from the more accurate witnesses (as judged by the accuracy of their past ratings) make a bigger impact on the WR value than those from the less accurate ones. In cases where all the witness ratings collected are disregarded, due to negative credibility of their providers, the WR component will produce no trust value (as in the case where it fails to collect witness ratings). In addition, at first, every witness receives the default credibility value since it has not provided witness ratings to agent a before. Hence, end users can set the value of $\mathcal{T}_{\mathsf{DWCr}}$ to reflect their policy towards newly encountered witnesses. For example, $\mathcal{T}_{\mathsf{DWCr}}$ can be set to 0 so that newly encountered witnesses are disregarded until they prove to be credible (by providing ratings in the acceptable accuracy threshold) or it can be set to 1 to reflect the policy that all witnesses are considered to be accurate and honest until proven otherwise. However, it can also automatically adjust based on the general level of witness credibility in the environment as shown in Section 7.2.

Similarly, we extend the CR component to make use of the credibility model. In brief, after every interaction, each reference of the target agent is compared with the actual outcome and a rating is recorded for the referee in terms of providing accurate references (denoted by $\text{term}_{\mathsf{RCr}}$) as in Equation 6.1. Then the referee

credibility of an agent is given by the following formula:

$$\mathcal{T}_{\mathsf{RCr}}(a,w) = \begin{cases} \mathcal{T}_{\mathsf{I}}(a,w,\text{term}_{\mathsf{RCr}}) & \text{if } \mathcal{R}_{\mathsf{I}}(a,w,\text{term}_{\mathsf{RCr}}) \neq \emptyset \\ \mathcal{T}_{\mathsf{DRCr}} & \text{otherwise} \end{cases} \tag{6.4}$$

where $\mathcal{T}_{\mathsf{RCr}}(a,w)$ is the referee credibility of w evaluated by a, $\mathcal{R}_{\mathsf{I}}(a,w,\text{term}_{\mathsf{RCr}})$ is the set of credibility ratings of w that a recorded in terms of providing accurate references, and $\mathcal{T}_{\mathsf{DRCr}}$ is the default referee credibility which is given when no such ratings has been recorded. Then the rating weight function for CR is redefined similar to Equation 6.3:

$$\omega_{\mathsf{R}}(r_i) = \begin{cases} 0 & \text{if } \mathcal{T}_{\mathsf{RCr}}(a,w) \leq 0 \\ \mathcal{T}_{\mathsf{RCr}}(a,w) \cdot \omega_{\mathsf{I}}(r_i) & \text{otherwise} \end{cases} \tag{6.5}$$

6.2 Empirical Evaluation

Having extended FIRE to deal with possible inaccuracy and disinformation in third-party reports, we now turn to evaluating its performance in such situations (i.e. those where some agents provide dishonest reports). The main question here is how the new WR and CR components cope with inaccurate reports. We survey the effect of them on WR in Section 6.2.1 and on CR in Section 6.2.2, respectively.

6.2.1 The effect of inaccurate reports on WR

In order to evaluate how the WR component copes with inaccurate witness reports, we need to extend the testbed in Section 4.2 to simulate lying in giving out witness ratings. Specifically, we introduce five types of witnesses: Hon, Pos1, Pos2, Neg1, and Neg2. Agents in the Hon group always reveal their actual ratings truthfully. Those in Neg1 and Neg2, however, provide to the querying agent ratings that are lower than those they actually recorded. Conversely, those in Pos1 and Pos2 give falsely higher ratings. The difference between an actual rating value and its fabricated one in Neg1 and Pos1 is randomly set in the range $[0.3, 1.0]$ (i.e. representing marginally inaccurate witnesses) and the respective range of Neg2 and Pos2 is $[1.0, 2.0]$ (i.e. representing extremely inaccurate witnesses). In the testbed, we also define levels of witness inaccuracy at the level of the overall system (-100 to 100) so that various configurations of *witness population* can be conveniently referred in our experiments. The proportions of witness types in

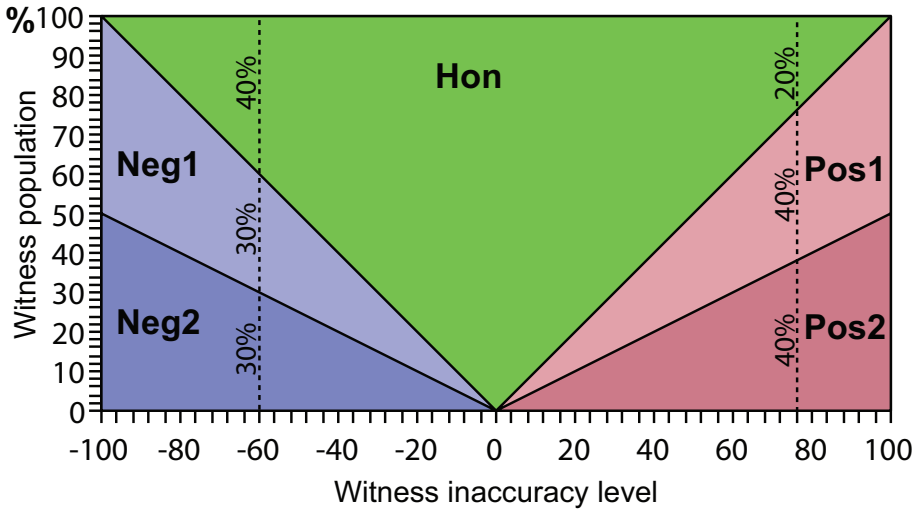

FIGURE 6.1: The proportions of witness types at various levels of witness inaccuracy.

each level of witness inaccuracy are given in Figure 6.1 (where vertical dotted lines represent specific example configurations of witnesses). For example, level 0 means that all the witnesses are of the Hon group (and provide their ratings honestly). Level +80 means that in the witness population, the proportions of Hon, Pos1, and Pos2 are 20%, 40%, and 40% respectively. It also means that 80% of the witness population is providing positively exaggerated reports (because 80% of the witnesses are either Pos1 or Pos2). Similarly, level −60 means that 60% of the witness population (30% Neg1, 30% Neg2) is providing falsely negative reports.

In our experiments, there are three groups of consumers: NoTrust, SPORAS, and WR. As in our previous experiments in Chapter 5, agents in the NoTrust group select provider agents for interaction randomly, those in SPORAS use the trust values provided by SPORAS for selecting providers, and those in WR use the WR and IT components of FIRE. The WR component is extended to make use of the credibility model as described in Section 6.1. The default witness credibility $\mathcal{T}_{\mathsf{DWCr}}$ is set to 0.5 so that all ratings from newly encountered witnesses will be taken into account in calculating WR, but their weights are smaller than those of any proven accurate witness (which is typically greater than $\iota = 0.5$, see Equation 6.2) and larger than that of a proven inaccurate one (which is typically negative). The value of ι is handpicked based on the actual variability of honest rating values in the testbed (which never exceeds 0.5).

Now, we look at how various levels of witness inaccuracy affect the performance of each consumer group using the methodology described in Section 4.1. The experiments are run with the following witness inaccuracy levels: −100, −80, . . . , 0,

..., +80, +100. In these experiments, inaccurate witnesses always provide inaccurate reports. After plotting the performance of the three consumer groups in each experiment (for example, see Figure 6.2), it can be seen that the performance of NoTrust is consistently low (around −1.0). On the other hand, thanks to the trust models used, the performance of SPORAS and WR are always higher than that of NoTrust. However, the performance of WR is always superior to that of SPORAS. In this experiment, since the performance of all providers are equally exaggerated, it is still the case that good providers generally have better ratings than others. Hence, in the first few interactions, they are selected by WR, and this accounts for an increase of WR's UG in this period. Now, after several interactions with these providers, WR is able to record their actual performance, which is generally lower than the reported performance of the remaining providers (calculated only from falsely positive reports). Thus, remaining providers (i.e. ordinary and bad ones) are then selected in later interactions. As a result, WR's UG is decreased. Nevertheless, because of the witness credibility model, during these interactions, WR is able to realise that all reports are inaccurate, and, thus, future (false) reports are disregarded. Effectively, WR resorts to the IT component for evaluating providers' trustworthiness. As for SPORAS, since it cannot filter out inaccurate reports, it cannot improve its performance over time. Now, due to the large number of experiments and associated settings that were conducted and because we are interested in the effects of varying levels of inaccuracy on WR, we only present a general analysis of the experiment's results. In more detail, the chart in Figure 6.3 shows the average performance of the three groups in each experiment with various levels of witness inaccuracy. Here, the average performance of each group is calculated as the average utility gained in each interaction of an agent in that group. This average performance is calculated from data of the first 200 interactions in each experiment (by which time the average UG of all groups is stable in all experiments).

From Figure 6.3, it can be seen that the performance of both WR and SPORAS suffer as the witness inaccuracy increases (as we would expect). However, the performance of WR is more robust. In particular, SPORAS suffers greatly from exaggerated positive ratings (because of the reason mentioned above). On the other hand, although WR also suffers from false positive ratings at the beginning (see Figure 6.2), it can gradually learn and disregard inaccurate witnesses and, generally speaking, it maintains a high performance. In the cases of falsely negative ratings (see Figure 6.3, witness inaccuracy level −100 to −20), SPORAS does not suffer as much as in the cases of exaggerated positive ratings. The reason is

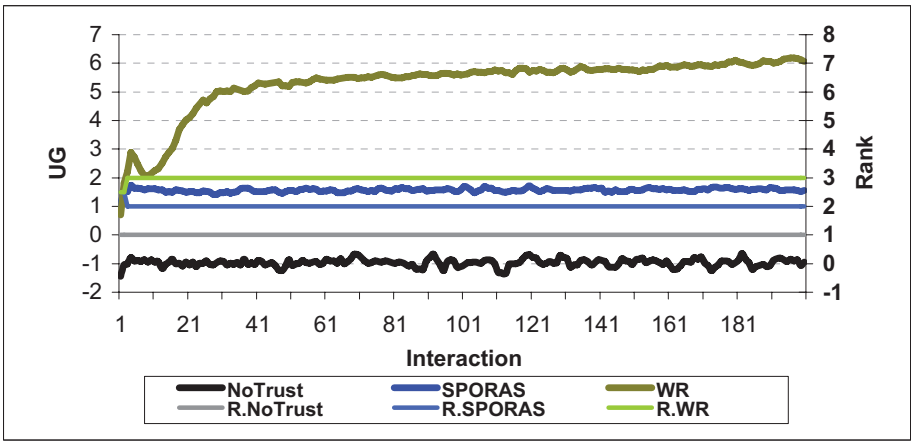

FIGURE 6.2: Performance of NoTrust, SPORAS, and WR at witness inaccuracy level +100.

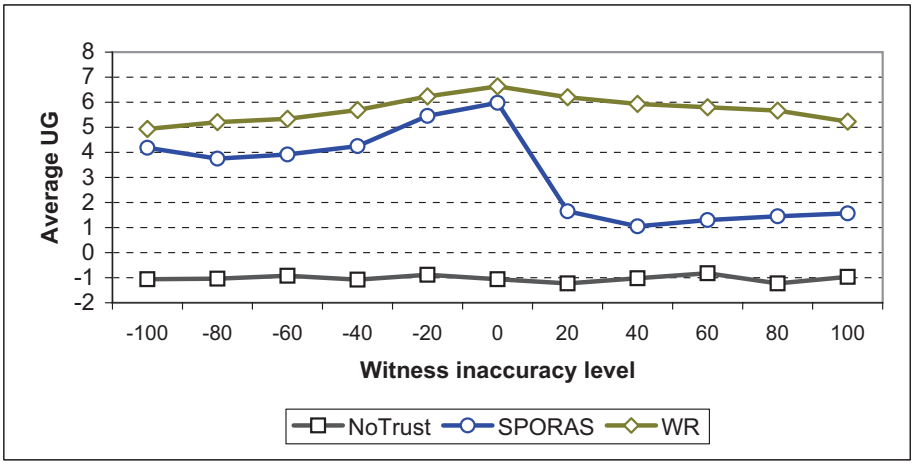

FIGURE 6.3: Performance of NoTrust, SPORAS, and WR at various levels of inaccuracy.

that falsely negative ratings not only lower the rating values of good providers, but also lower those of bad and ordinary ones by similar amounts. Hence, it is still the case that good providers have better reputations in SPORAS, and, thus, they are more likely to be selected for interaction. This means that SPORAS can perform normally in scenarios where all lying witnesses provide negative ratings. However, this is because of the nature of that specific lying population rather than SPORAS's ability of dealing with inaccurate reports. As for WR, as mentioned above, its ability to detect and penalise inaccurate witnesses also works in such scenarios and allows WR to maintain a generally high performance.

Next, we seek to determine whether our model can cope with situations where witnesses have more subtle lying behaviour — they lie sometimes and provide their honest ratings at other times. Here, we investigate two scenarios: the lying witnesses provide false ratings: (1) 25% of the time (i.e. being mostly honest, lying

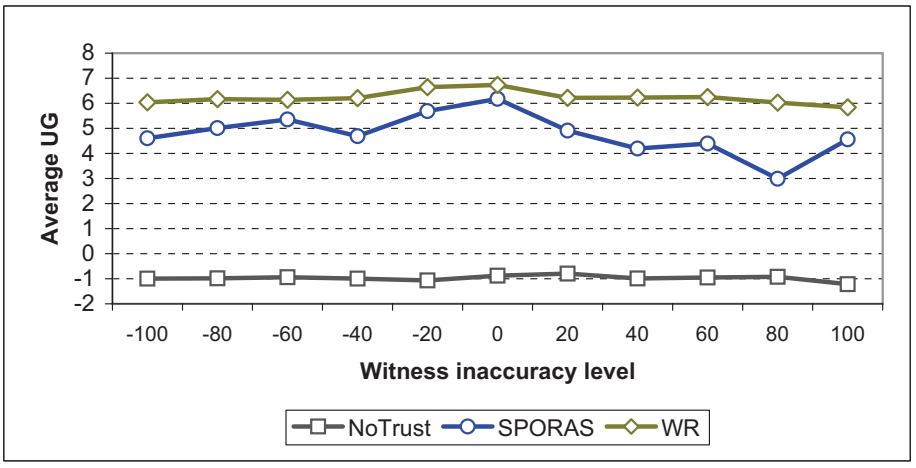

FIGURE 6.4: Lying 25% of the time.

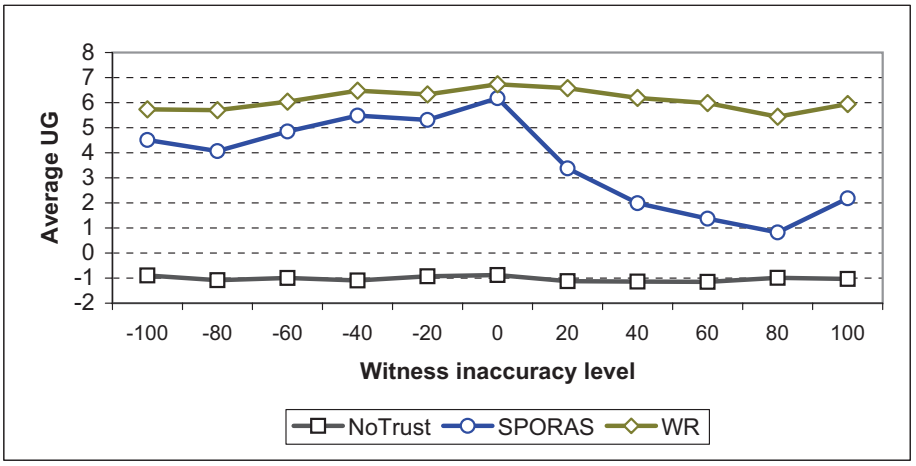

FIGURE 6.5: Lying 75% of the time.

sometimes) and (2) 75% of the time (i.e. mostly lying, being honest sometimes). The two cases are interesting because some agents may try to fool a reputation system by lying only a few times to maintain their credibility (case 1) or by giving reports honestly to increase its (bad) credibility (case 2). The set of experiments are re-run for the two scenarios and their results are presented in Figures 6.4 and 6.5. From these, it can be seen that the performance of all groups have a broadly similar pattern in scenarios of negative lying. However, as in the scenarios of lying 100% of the time, SPORAS suffers adversely from exaggerated positive reports (as in the previous experiment). It can also be seen that SPORAS's performance decreases in proportion to the amount of lying (i.e 25%, 75%, 100%). In contrast, in all scenarios presented, our witness credibility model can learn the witnesses' lying behaviour (thanks to the adaptive nature of FIRE's IT component), and this accounts for the robust performance of WR throughout in these scenarios.

6.2.2 The effect of inaccurate reports on CR

Now, we turn to assessing the effect of inaccurate reports on the CR component. Similar to the previous section, in order to do so, we also introduce lying behaviours when providing references into the testbed. However, since references are given and disclosed directly to the refereed agents, a rational referee typically does not simply give bad references to its interaction partners. The reasons are that (1) the badly refereed agent might retaliate and (2) it can disregard such references. Moreover, a rational referee normally does not give exaggerated references for everyone. Rather, it is more likely to give them only to certain agents — called its *friend agents* (e.g. those from the same organisation or having common interests). Thus, in each experiment in this section, a consumer agent has a maximum number of friend providers ($\mathsf{N_{FP}}$) that it may collude with when providing references about their performance. Such providers are selected randomly from a consumer's nearby providers when it enters the testbed. In addition, we introduce five types of referees: Hon, Exag1, Exag2, Extr1, and Extr2. Agents in the Hon group always give out their actual ratings as references (as per Section 6.2.1). *Exaggerating referees* in groups Exag1 and Exag2, however, give falsely higher ratings than those they actually recorded for their friend providers[3] (and their actual ratings for the others). In addition to giving falsely inflated references for their friends, *extreme referees* in Extr1 and Extr2 also deliberately underrate the other providers. The difference between an actual rating value and its inaccurate one in Exag1 and Extr1 is randomly set in the range $[0.3, 1.0]$ (i.e. representing marginally inaccurate referees) and the respective range of Exag2 and Extr2 is $[1.0, 2.0]$ (i.e. representing extremely inaccurate referees).

In this section, we carry out several experiments to evaluate the new CR component in a wide range of environments. In these experiments, there are three groups of consumer agents: NoTrust, SPORAS, and CR. The NoTrust and SPORAS groups are defined as in our previous experiments. The CR group consists of agents using the CR component extended to deal with inaccurate references (Section 6.1). Similar to the experiments with WR, the default credibility level of referee $\mathcal{T}_{\mathsf{DRCr}}$ is 0.5 and the lying threshold ι is 0.5. The consumers in each experiment consist of honest referees and those from only one of the four colluding referee groups (Exag1, Exag2, Extr1, and Extr2). Each consumer is set to have a maximum of four friend providers ($\mathsf{N_{FP}} = 4$). The proportion of colluding referees is set to be 0%, 20%, 40%, 60%, 80%, and 100% in these experiments. For example, Figure 6.6 presents

[3]This is motivated by examples where referees provided exaggerated reports about their friends.

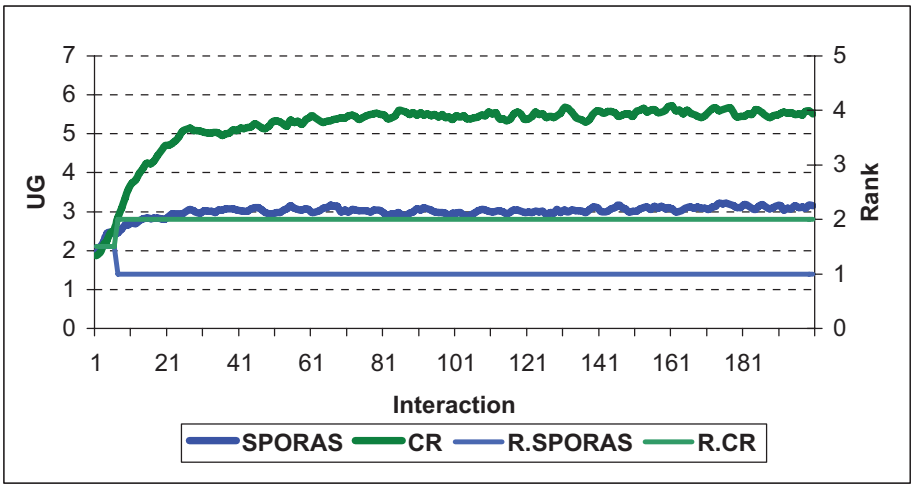

FIGURE 6.6: 80% Extr2 referees.

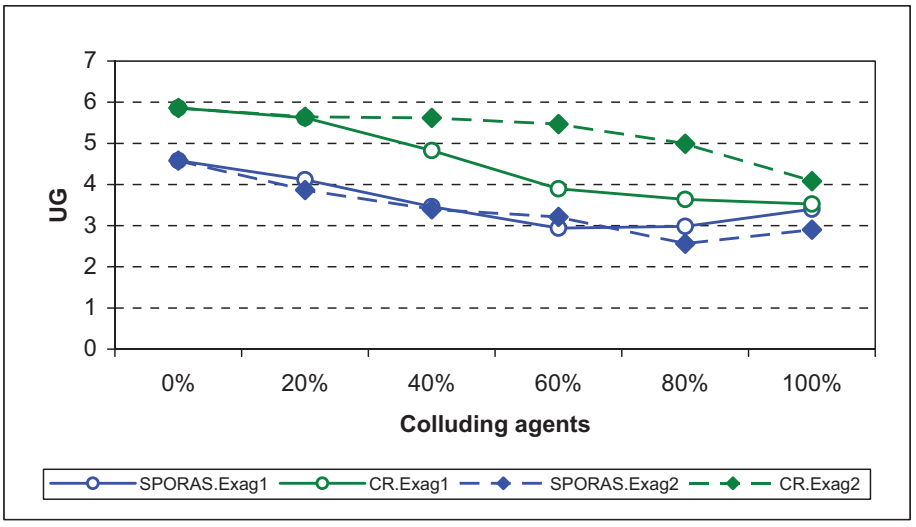

FIGURE 6.7: Performance with exaggerating referees.

the result from the experiment where the consumers consist of 20% honest referees and 80% colluding referees from the Extr2 group. Since NoTrust performs consistently poorly in all the experiments, its results are omitted from our charts for the sake of simplicity. Moreover, instead of providing a detailed result of every experiment as in Figure 6.6, for an overview of the effect of various proportions of colluding referees versus honest referees, we plot the average UG per interaction of SPORAS and CR in each experiment on the summary charts in Figures 6.7 and 6.8. In these charts, the plots are named as GroupName.RefereeType. For example, SPORAS.Extr1 is the plot for the average UG of agents in the SPORAS group when the colluding referees in the testbed are of the Extr1 group.

The first thing these charts show is that collusion adversely affects the performance of trust models (as we would expect). For instance, Figure 6.6 shows that

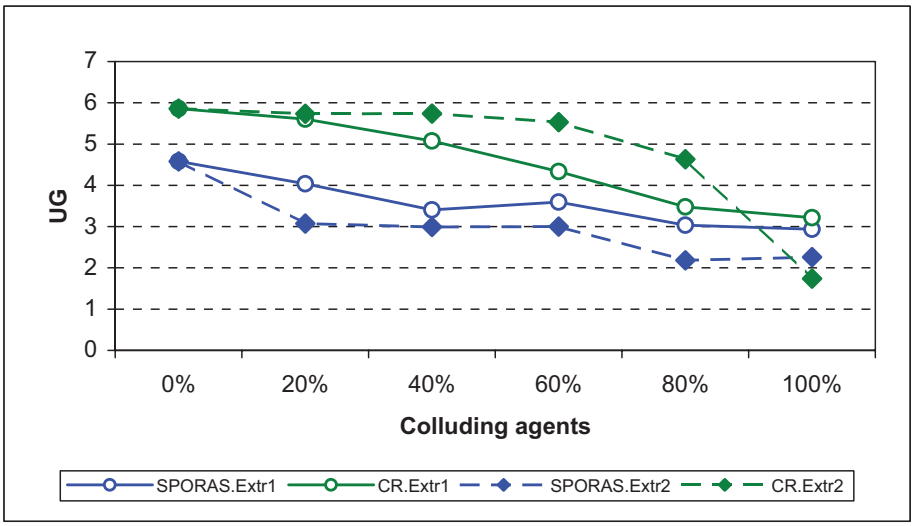

FIGURE 6.8: Performance with extreme referees.

it takes longer for CR to reach a stable level of performance (i.e. to learn about the colluding agents) and this performance is also lower than that in an honest environment (see Figure 5.13, Section 5.3). In such circumstances, SPORAS also yields a low UG and, without a credibility model, it cannot improve its performance over time. We can also see that the average performance of both SPORAS and CR generally decreases when the number of colluding agents increases (Figures 6.7 and 6.8). However, CR always outperforms SPORAS except in the case when 100% of the consumers are Extr2 agents. In this particular experiment, since all consumers are identified by CR as lying (because all of them provide highly distorted references for all the providers), CR stops using their references, thus, depressing performance. However, this particular case (i.e. 100% extreme collusion) is highly unlikely to happen in practice (and if it did one might just retract trust from the population altogether). In the remaining experiments, CR can easily detect referees providing highly distorted references and maintain a generally high performance (see plots CR.Exag2 and CR.Extr2). CR is less effective in filtering out the colluding agents in Exag1 and Extr1 since their colluded references are less distorted than in the case of Exag2 and Extr2 (i.e. more difficult to detect lying). This suggests that the inaccuracy tolerance threshold ι should be carefully selected to reflect the nature of biased behaviours in a particular environment, or better, learning techniques could be used to enable an agent to adjust this parameter according to the prevailing context. In this regard, Chapter 7 presents such a technique in which FIRE monitors the general level of deviation in the providers' performance in the environment and accordingly adjust the inaccuracy tolerance threshold on-the-fly.

In summary, the results shows that the credibility model of CR enables it to outperform SPORAS in dealing with biased behaviours. Specifically, it allows agents using CR to maintain a robust and high performance in a wide variety of cases, especially when the level of collusion is less than 50% (which is, in our opinion, the most likely case in realistic scenarios).

6.3 Summary

The basic FIRE model presented in Chapter 3 is not able to recognise inaccurate third-party reports and, thus, in its evaluation (Chapter 5) we had to assume that all agents were honest. However, this is not realistic in open MAS (as discussed in our requirements in Section 2.5). Given this, this chapter describes our novel model of credibility which extends the basic FIRE model in that it can now detect such inaccuracy. This is done based solely on an agent's own experience, and, therefore, no additional domain knowledge is required. As a result, an agent is able to keep track of the quality of third-party reports and assess the credibility of a witness or a referee accordingly. Making use of the credibility model, our extended WR and CR components take into account the credibility of reporters in weighing their information. Hence, reports from inaccurate reporters can quickly be filtered out.

Using empirical evaluation, we have shown that with the credibility model our WR and CR components perform well in a variety of contexts. In all the experiments where the percentages of inaccurate reporters are less than 50%, the performance of the WR and CR components is maintained at a comparable level to that in honest environments. Therefore, with the addition of the credibility model, FIRE satisfies the requirement **R4a** on being robust against inaccuracy reports (see Section 2.5). This completes all our requirements for a trust model in open MAS.

In developing a trust model that satisfies our initial requirements, we introduced various parameters so that FIRE can be adapted and fine tuned for a wide range of applications. However, choosing the right parameters for FIRE to work efficiently in a particular application currently requires a good understanding of how FIRE operates and the application domain. To overcome this, we believe it is possible to use a variety of learning techniques to enable FIRE to adapt to its operating environment and to automatically adjust its parameters accordingly. This work is detailed in the next chapter.

Chapter 7

Adapting FIRE's Parameters

FIRE is a parameterised model; that is, its behaviour is defined by a set of parameters. Such parameterisation is necessary in order to make FIRE customisable to suit particular environments and/or applications. For example, in environments where there are a high level of lying agents, the component coefficients for the WR and CR components should be decreased because third-party information is highly unreliable; or the recency scaling factor λ (see Section 7.1) can be adjusted to suit the time unit used in a particular environment. However, choosing the right set of parameters for FIRE to work efficiently in a particular environment or application currently requires a good understanding of how FIRE operates and also of the application domain. To overcome this, we believe it is possible to use a variety of learning techniques to enable FIRE to adapt to its operating environment and to automatically adjust its parameters accordingly. Having been equipped with such techniques, FIRE is able to adapt itself to changes in an environment on-the-fly. The benefit of this is twofold. First, a self-adaptive model requires minimal administration because parameters which are initially set to wrong values can be readjusted automatically by the model without requiring human intervention. Second, such a model is more robust in a dynamic environment like an open MAS since it can adapt itself to changes that were unforseen by the model's designer. This chapter surveys the parameters of FIRE and possible techniques to make it an adaptive model. In the next section, an analysis of FIRE's parameters is carried out to choose the parameters to become adaptive. Sections 7.2, 7.3, and 7.4 then describe our learning algorithms for the chosen parameters. Various experiments are then carried out to empirically evaluate the proposed techniques in Section 7.5. Finally, a summary is provided in Section 7.6.

Parameter	Symbol
Rating history size	H
Rating reliability scaling factor	γ_{K}
Recency scaling factor	λ
Branching factor	n_{BF}
Referral length threshold	n_{RL}
Component coefficients	W_{K}
Inaccuracy tolerance threshold	ι
Default witness credibility	$\mathcal{T}_{\mathsf{DWCr}}$
Default referee credibility	$\mathcal{T}_{\mathsf{DRCr}}$

TABLE 7.1: FIRE's parameters.

7.1 Determining the Parameters to Adapt

FIRE's parameters (Table 7.1) are designed to enable end users to control its behaviours and to allow it to be customised for a particular application. Some of them can be set according to the end users' subjective considerations, while the others need to be set according to the (objective) constraints of the application. For example, an end user can set FIRE to rely more on the IT component than the others because he believe IT is more reliable than WR and CR, but he can only set the rating history size according to the available memory of his agents. Since objective parameters can only be set according to an agent's individual situations that are out of FIRE's control, they are not suitable to become adaptive. Therefore, in what follows, we identify the subjective parameters and study how they can made adaptive in the subsequent sections:

- *Rating history size* (H): This parameter defines the maximum number of ratings of a partner (from direct interactions) that are stored in an agent's memory (Section 3.1). Thus, for a given agent, only its latest H ratings are stored; its older ratings are discarded if the number of its ratings exceeds H. Generally speaking, the more ratings that are stored, the longer the history, or the more knowledge, about an interaction partner is retained. However, in practice, this is constrained by an agent's memory capacity. Therefore, the agent's designers must choose H according to the memory resource available to it.

- *Rating reliability scaling factors* (γ_{K}[1]): Each component of FIRE has its own rating reliability measure to evaluate the reliability, or the quality, of the

[1] As in previous chapters, K is one of I, R, W, and C, which represent the IT, RB, WR, and CR components, respectively.

ratings it receives (Section 3.2). In each component, the reliability values of the ratings taken into account are used to calculate the reliability value of the trust value produced from those ratings. The reliability values of trust values need to conform to the same kind of representation (i.e. 0 for completely uncertainty and 1 for total confidence) because they are also used as weights for trust values produced by the trust components that are combined into an overall trust value (Section 3.7). However, since end users are free to replace FIRE's rating reliability measure with their own (to suit a particular application) and because there is no constraint on such measures except that they are required to produce a non-negative reliability value for each rating, the range of rating reliability values can vary greatly. Therefore, a rating reliability scaling factor (γ_{K}) is devised for each component to scale (and to normalise) the range of the rating reliability measure it uses; and, thus, it needs to be set according to the range of the rating reliability measure used in each component.

- *Recency scaling factor* (λ): This parameter is devised to scale the time unit that an application uses (Section 3.3) and, thus, allows FIRE to work with various time systems. For example, artificial time ticks might be used in one application, while real time seconds might be used in another. Hence, this parameter needs to be set to suit the time system used in a particular application.

- *Branching factor* (n_{BF}) and *Referral length threshold* (n_{RL}): The Witness Reputation component uses these two parameters to define the search range in which it will look for the required witnesses (Section 3.5). The former limits the breadth of the search range (i.e. how many acquaintances an agent will pass on a query onto) and the latter limits how far away it is (i.e. to query agents at how many links away from the querying agent). Although a broader and further search range potentially yields a better result (i.e. more witnesses found) than a smaller one, as a trade-off, it also requires more resources in terms of (searching) time, communication costs, and, sometimes, information costs (e.g. a queried agent may request payment for its information). In light of this, the two parameters above were designed to enable an agent to limit the number of agents to be contacted in a witness search to suit its resource constraints. Therefore, these parameters can only be set by the agent or its designer according to its own situation.

- *Component coefficients* (W_{K}): In order to evaluate the trustworthiness of an agent, FIRE uses its four components and combines the trust values that

these components produce into an overall value (Section 3.7). Since each component derives trust values from an independent source of ratings, its accuracy can differ from the others'. Therefore, trust values from the four components are not equally taken into account when an overall trust value is produced, but, rather, they are weighted according to an agent's belief on how accurate they are deemed to be (Equation 3.6). For instance, an agent might believe that the ratings it makes itself are more accurate than third-party ratings and, thus, it gives more weight to the trust values produced by the Interaction Trust component than to those produced by the Witness and Certified Reputation components. Here, the component coefficients (w_{K}) are the weights for the components and are set by an agent's designer according to such beliefs. However, presetting fixed component coefficients might yield bad results when the actual situation does not correspond with the agent's beliefs. For example, an agent might interact very infrequently and, thus, might not have sufficient ratings for the IT component to produce accurate trust values. In that situation, relying heavily on IT rather than WR or CR would possibly give an unreliable overall trust value. Another possible situation is where the number of inaccurate reporters (i.e. witnesses, referees) in an agent's environment increases or decreases; and this will affect the accuracy of the WR or CR components. In order to deal with such uncertainties, FIRE should be able to adjust the component coefficients to reflect the actual accuracy of the corresponding components.

- *Inaccuracy tolerance threshold* (ι): In our credibility model (Section 6.1), the inaccuracy tolerance threshold ι is used to speed up the process of filtering out clearly unreliable (e.g. lying) witnesses or referees by classifying those that provide ratings whose inaccuracy exceeds ι as liars and giving them the minimum credibility rating value (-1). However, incorrect classification is always possible and, if made, can be a significant problem. For example, a provider's performance can vary so highly that it makes an honest and accurate recent rating too different to that provider's next performance and since the difference is greater than the (low) ι value, the rating's reporter is (wrongly) classified as lying. This would result in that reporter's future (honest) ratings be disregarded. Conversely, an unnecessarily high ι value is also inefficient as it allows lying reporters to 'get away' (i.e. to be undetected) and slows down the learning process of FIRE's credibility model. In such situations, knowing the typical performance deviation of providers in an environment can help in choosing a suitable value for ι. Therefore, we

propose to monitor the performance deviation of all the providers an agent encounters and to use it to adjust the inaccuracy tolerance threshold ι.

- *Default witness credibility* ($\mathcal{T}_{\mathsf{DWCr}}$) and *Default referee credibility* ($\mathcal{T}_{\mathsf{DRCr}}$): These two parameters are the default values for the credibility of a witness and a referee, respectively, when FIRE cannot evaluate the credibility of them because of a lack of past experience (Section 6.1). They are normally set (by a designer) to reflect the policy of the agent for handling information from a newly encountered witness or referee. Setting $\mathcal{T}_{\mathsf{DWCr}}$ to 0, for example, means that witness information from newly encountered witnesses are initially believed not to be accurate and, therefore, not used; thus, all witnesses need to prove their credibility by presenting accurate information to the agent first. Conversely, setting the parameter to 1 means that all newly encountered witnesses are believed to be accurate unless proved otherwise (i.e. detected giving false ratings). $\mathcal{T}_{\mathsf{DRCr}}$ is set similarly. However, either setting can be inefficient. Setting the default credibility to 0 in an honest environment (where most agents are honest and accurate) will initially render valued information from newly encountered agents useless, while setting it to 1 in an environment where there are many lying agents will feed inaccurate information to FIRE. Therefore, we propose to monitor the level of inaccurate witnesses/referees in an agent's environment to calculate the likely credibility of a newly met agent.

In summary, FIRE's subjective parameters are: the component coefficients, the inaccuracy tolerance threshold, and the default witness/referee credibility. In the subsequent sections (7.2, 7.3, and 7.4), we study present our approaches to make these parameters adaptive in order to push FIRE towards a self-adaptable trust model.

7.2 Component Performance Learning

As discussed above, in order for FIRE to efficiently make use of the trust values produced by its components, the component coefficients W_{K} should be set to reflect the accuracy of the corresponding components and they should also be updated accordingly when the components' performance changes. To this end, we use FIRE itself to monitor the performance (i.e. accuracy) of its components. In order to do so, we introduce four virtual agents into the FIRE model of each agent; they are

named comp_{I}, comp_{R}, comp_{W}, and comp_{C} representing the IT, RB, WR, and CR components respectively. The performance of each component is then recorded as the performance of the corresponding agent. It is measured by how accurate its trust values are in predicting the performance of other agents as follows. Suppose that agent a uses FIRE to predict the performance of agent b and then interacts with b. After the interaction, a can observe the actual performance of b and can use this to determine how accurate each of FIRE's component was. Specifically, let v_{K}^b be the trust value of b that component K produced before the interaction and v_o be b's actual performance observed by a; the accuracy of component K in that particular interaction is calculated from the difference between v_{K}^b and v_o:

$$v_i = 1 - |v_o - v_{\mathsf{K}}^b| \tag{7.1}$$

where v_i is the accuracy of component K. Since $-1 \leq v_o, v_{\mathsf{K}}^b \leq 1$, v_i is also bound in $[-1, 1]$, where 1 is the maximum possible accuracy (when $v_o = v_{\mathsf{K}}^b$) and -1 is the minimum (when $|v_o - v_{\mathsf{K}}^b| = 2$). The accuracy of component K is then recorded as a rating for comp_{K}: $r_i = (a, \text{comp}_{\mathsf{K}}, te_i, \text{term}_{\mathsf{CP}}, v_i)$, where te_i refers to the trust evaluation that a made using FIRE and $\text{term}_{\mathsf{CP}}$ is the term for trust component performance.

Having recorded ratings for each component, FIRE can now evaluate their performance. Since these ratings are the direct experience of agent a, we use the IT component for this purpose:

$$\mathcal{T}_{\mathsf{CP}}(a, \text{comp}_{\mathsf{K}}) = \mathcal{T}_{\mathsf{I}}(a, \text{comp}_{\mathsf{K}}, \text{term}_{\mathsf{CP}}) \tag{7.2}$$

where $\mathcal{T}_{\mathsf{CP}}(a, \text{comp}_{\mathsf{K}})$ is the trust agent a has in the performance of comp_{K}; or, in other words, the expected performance of comp_{K} given a's experience (i.e. ratings). The trust value for comp_{K} is then used to adjust the component coefficient of component K; the potential value for W_{K} (denoted by W'_{K}) is taken from $\mathcal{T}_{\mathsf{CP}}(a, \text{comp}_{\mathsf{K}})$ if it greater than 0 (i.e. the trust values produced by component K are expected to be somewhat accurate), otherwise (i.e. $\mathcal{T}_{\mathsf{CP}}(a, \text{comp}_{\mathsf{K}}) < 0$, component K is too inaccurate) trust values from component K should be ignored (W'_{K} is set to 0):

$$W'_{\mathsf{K}} = \max\{0, \mathcal{T}_{\mathsf{CP}}(a, \text{comp}_{\mathsf{K}})\} \tag{7.3}$$

Here, the potential value W'_{K} is not assigned directly to W_{K} because it might fluctuate significantly from interaction to interaction. This is because the accuracy of a trust component can sometimes vary greatly depending on the target agent it

evaluates. For instance, an agent may interact with others frequently and its IT component generally performs well, but for a target agent it has never interacted with before, the IT component's performance can be very low (due to the lack of direct experience). In order to avoid the situation in which a trust component is temporarily disabled because of such fluctuations, FIRE partly 'remembers' the previous value of a component coefficient; thus, it adjusts W_{K} towards the potential value W'_{K} instead of using W'_{K} directly:

$$W_{\mathsf{K}} = q_{\mathsf{CPL}} \cdot W_{\mathsf{K}}^{pre} + (1 - q_{\mathsf{CPL}}) \cdot W'_{K} \tag{7.4}$$

where W_{K}^{pre} is the previous value of W_{K} (i.e. before this update) and $q_{\mathsf{CPL}} \in [0, 1]$ is the memory parameter for component performance learning defining how strongly FIRE remembers previous values of W_{K}. If q_{CPL} is set to 0, FIRE will 'forget' the previous value completely; the nearer q_{CPL} is to 1, the less W_{K} is adjusted towards its potential value W'_{K}.

7.3 Inaccuracy Threshold Learning

Classifying a reporter as honest or lying only based on the information it provides is difficult. Typically, the evaluator's view in an open MAS is limited (see Section 1.1) and there is usually no way it can confirm whether the ratings it receives are honest and based on the actual observation of the target agent's performance. Thus, in order to assess the accuracy of such third-party ratings, the evaluator can only compare them with its own observations of the target agent. In our credibility model (Section 6.1), the difference between a third-party rating of the target agent and the performance the evaluator receives from it serves as the measure of the reporter's accuracy. Based on this measure, the reporter is then classified as lying/inaccurate or honest/accurate. However, this accuracy measure alone is not enough. The difference between a third-party rating and the performance the evaluator observes can be attributed to many things: (1) the variation of the target agent's performance, (2) the reporter's (in)ability of making accurate ratings, and (3) the reporter's intentional manipulation of the rating's value. Ideally, only when (2) or (3) is the case should the reporter be classified as lying/inaccurate and its future ratings be filtered out (as a result). However, there is no way for the evaluator to tell whether the difference is due to (1) or not because it cannot know the performance the reporter received from the target agent. Therefore, it can only (reasonably) expect the performance variation of providers to be relatively

small and that any inaccuracy greater than the typical performance variation of providers is deemed to be due to (2) or (3). Here, the inaccuracy tolerance threshold ι serves as the borderline to separate between cases (1) and cases (2) or (3) and it needs to be set based on the typical performance variation of providers in an agent's environment. As discussed at the beginning of the chapter, fixed values for ι can be inefficient if the agent's designer chooses a wrong value. Moreover, an agent can interact with only a small set of providers in an open MAS whose typical performance variation is different to that of another set or that of the whole open MAS. Hence, in this section, we propose an algorithm to monitor the performance variation of the providers an agent encounters and to adjust the inaccuracy tolerance threshold accordingly.

In order to do so, we first need to calculate the performance deviations of the providers that the evaluator, say, agent a, has encountered. The performance deviation of a provider, say, agent b, in term c is calculated by a as follows:

$$\mathsf{dv}(a,b,c) = \frac{\sum_{r_i \in \mathcal{R}_\mathsf{I}(a,b,c)} |v_i - \overline{v}|}{|\mathcal{R}_\mathsf{I}(a,b,c)|} \tag{7.5}$$

where $\mathsf{dv}(a,b,c)$ is the performance deviation of b in term c that is observed by a, $\mathcal{R}_\mathsf{I}(a,b,c)$ is the set of ratings of b in term c that a made from past interactions with b, v_i is the value of the rating r_i, and $\overline{v}$ is the mean value of all the rating values in the set $\mathcal{R}_\mathsf{I}(a,b,c)$. The performance deviation is used to estimate the variations in providers' performance only when it is calculated from at least two ratings. Otherwise, the deviation from one rating is always 0 and does not give any information regarding the variation of a provider's performance. Next, the average performance deviation of all the provider agents a encountered at least twice (denoted by the set P) is used as the potential value for the inaccuracy threshold (denoted by ι'):

$$\iota' = \frac{\sum_{p \in \mathsf{P}} \mathsf{dv}(a,p,c)}{|\mathsf{P}|} \tag{7.6}$$

where $|\mathsf{P}|$ is the number of agents in the set P. If this number is too small, ι' may not be representative and the inaccuracy tolerance threshold ι should not be replaced by it completely. Otherwise, the value of ι can fluctuate significantly when an agent has just joined an environment and interacted with a small number of agents. Hence, ι is updated as follows:

$$\iota = \begin{cases} \frac{|\mathsf{P}| \cdot \iota' + (n_{\mathsf{MD}} - |\mathsf{P}|) \cdot \iota^{pre}}{n_{\mathsf{MD}}} & \text{if } |\mathsf{P}| < n_{\mathsf{MD}} \\ \iota' & \text{otherwise} \end{cases} \tag{7.7}$$

where n_{MD} is the minimum number of deviation values for ι' to be considered representative and ι^{pre} is the previous value of ι (i.e. before this update). In short, if the potential value ι' is calculated from the deviation values of sufficient agents (i.e. $\geq n_{\mathsf{MD}}$) then ι' is assigned to ι. Otherwise, ι is only partly updated based on ι' and the impact of ι' on the new value of ι is given by the number of deviation values taken into account ($|\mathsf{P}|$) when ι' is calculated. The inaccuracy tolerance threshold ι is updated every time an agent has new information about the performance variation of its providers. This means that an ι update happens (after) every time a interacts with a provider agent (i.e. FIRE received a new direct rating) that it has interacted with before (so that a has at least two ratings of the rated agent to calculate its performance deviation).

7.4 Default Credibility Learning

In this section, FIRE is extended to monitor the level of inaccuracy of the witnesses and referees that an agent encounters. It can then adjust the default credibility values for newly met witnesses and referees so that it does not ignore potentially good information or use potentially bad information (as discussed in the examples provided in Section 7.1). In order to do so, the latest credibility of every agent calculated by the evaluator is stored in a *credibility cache*. FIRE is equipped with two such caches; one for witnesses, denoted by WCr, and the other for referees, denoted by RCr. In our notation here, WCr and RCr are treated as sets of credibility values and $\mathsf{WCr}(w)$ and $\mathsf{RCr}(r)$ are used to refer to the latest credibility values of witness w and referee r, respectively. The process of adjusting the default witness credibility is as follows:

1. When evaluating the credibility of a witness w, if the evaluator has credibility ratings of w (see Section 6.1), w's credibility value can be calculated and is then stored in WCr (i.e. the evaluator's witness credibility cache).

2. If the evaluator does not have any credibility ratings of w and, thus, FIRE cannot calculate the credibility of w. In this case, we would expect that the credibility of w is similar to that of the witnesses the evaluator had experiences with. Therefore, the default credibility for w should be calculated as the mean credibility from all the credibility values of the witnesses the

evaluator knows (i.e. which are stored in the witness credibility cache):

$$\overline{\mathsf{WCr}} = \frac{\sum_{r \in \mathsf{WCr}} \mathsf{WCr}(r)}{|\mathsf{WCr}|} \tag{7.8}$$

where $\overline{\mathsf{WCr}}$ is the mean credibility of all witnesses in the set/cache WCr.

3. As in the previous section, $\overline{\mathsf{WCr}}$ can only serve as the potential value for the default witness credibility $\mathcal{T}_{\mathsf{DWCr}}$ since it may not be representative if calculated from a small number of witnesses' credibility values. Likewise, we introduce the minimum number of witness credibility values, denoted by n_{MWCr}, required for $\overline{\mathsf{WCr}}$ to be representative. The default witness credibility value is then updated as follows:

$$\mathcal{T}_{\mathsf{DWCr}} = \begin{cases} \frac{|\mathsf{WCr}| \cdot \overline{\mathsf{WCr}} + (n_{\mathsf{MWCr}} - |\mathsf{WCr}|) \cdot \mathcal{T}^{pre}_{\mathsf{DWCr}}}{n_{\mathsf{MWCr}}} & \text{if } |\mathsf{WCr}| < n_{\mathsf{MWCr}} \\ \overline{\mathsf{WCr}} & \text{otherwise} \end{cases} \tag{7.9}$$

where $\mathcal{T}^{pre}_{\mathsf{DRCr}}$ is the previous value of $\mathcal{T}_{\mathsf{DWCr}}$ (i.e. before this update).

Similarly, for the default referee credibility ($\mathcal{T}_{\mathsf{DRCr}}$), the mean credibility is calculated from the credibility values stored in the credibility cache RCr:

$$\overline{\mathsf{RCr}} = \frac{\sum_{w \in \mathsf{RCr}} \mathsf{RCr}(r)}{|\mathsf{RCr}|} \tag{7.10}$$

where $\overline{\mathsf{RCr}}$ is the mean credibility of all referees in the set/cache RCr. The default referee credibility $\mathcal{T}_{\mathsf{DRCr}}$ is then updated based on this mean value:

$$\mathcal{T}_{\mathsf{DRCr}} = \begin{cases} \frac{|\mathsf{RCr}| \cdot \overline{\mathsf{RCr}} + (n_{\mathsf{MRCr}} - |\mathsf{RCr}|) \cdot \mathcal{T}^{pre}_{\mathsf{DRCr}}}{n_{\mathsf{MRCr}}} & \text{if } |\mathsf{RCr}| < n_{\mathsf{MRCr}} \\ \overline{\mathsf{RCr}} & \text{otherwise} \end{cases} \tag{7.11}$$

where $\mathcal{T}^{pre}_{\mathsf{DRCr}}$ is the previous value of $\mathcal{T}_{\mathsf{DWCr}}$ (i.e. before this update) and n_{MRCr} is the minimum number of referee credibility values required for calculating $\overline{\mathsf{RCr}}$ so that it can be considered representative.

7.5 Empirical Evaluation

Having presented the learning techniques to automatically adjust the component coefficients, the inaccuracy threshold, and the default witness/referee credibility, we now turn to their evaluation to determine whether these techniques are in fact

effective (i.e. they can choose the right values for the parameters in question in a variety of changing situations). First, Section 7.5.1 starts with the experiments to evaluate the learning algorithm for component coefficients. Those for adjusting the inaccuracy threshold and the default witness/referee credibility are subsequently presented in Sections 7.5.2 and 7.5.3, respectively. All the experiments use the standard settings as defined in Section 4.3 unless otherwise stated.

7.5.1 Component performance learning

In this section, the learning algorithm for FIRE's component coefficients is evaluated to see whether it helps FIRE to cope with the situation in which a component that used to function well fails. The WR and CR components are chosen as the target for the experiments in this section since their performance depends significantly on the inaccuracy of the ratings they receive and this can easily be manipulated in our testbed. Specifically, in these experiments, none of the consumer agents is an honest witness or an honest referee. They all provides both false witness and certified ratings. However, their lying percentage can be externally control. For example, in the first experiment, at first the WR lying percentage is set to 100% and the CR lying percentage is set to 0%. This means that each consumer lies 100% of the time when providing witness ratings and always provides honest certified ratings (0% lying). This effectively renders the WR component useless (because all witnesses are lying all the time) and leaves the CR component in tact. During the experiment, the WR and CR lying percentages are changed often to create a situation where the components' performance is affected by the changes in the agent's environment. In order to simulate drastic changes in an environment that can render one trust component useless, the WR and CR lying percentages are set to be alternating between 100% and 0% every 60 rounds (Figure 7.1). For example, in the first 60 rounds, the WR component is almost useless (because all witnesses are lying), while all the information the CR component receives is honest and accurate. The situation of the two components are then switched after every 60 rounds. It should be noted that this is the worst case scenario which is somewhat unlikely to happen in practice. However, this case is chosen for our first experiment here to demonstrate the maximum benefit of component performance learning and to test its ability to cope with such drastic changes.

In order to evaluate the new learning algorithm, we introduce two groups of consumer agents into this experiment. Both of them use FIRE without the IT component (since we are interested only in the performance of the WR and CR com-

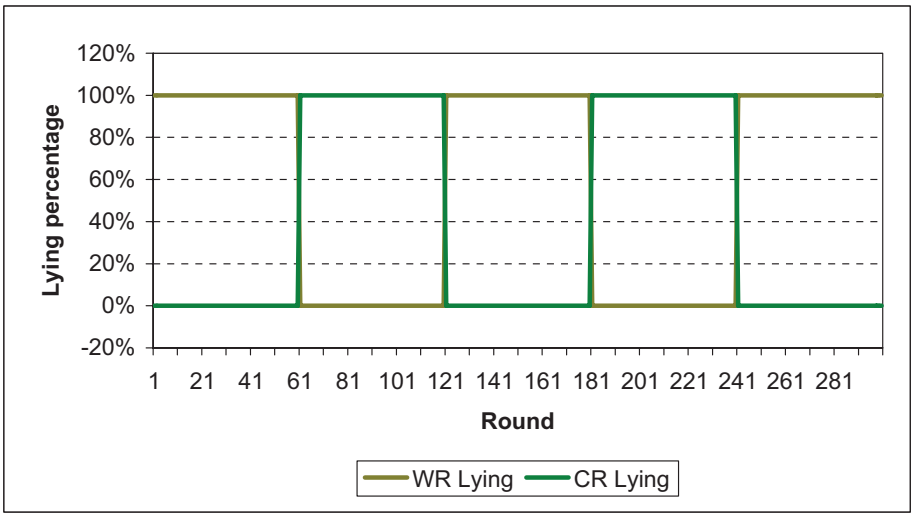

FIGURE 7.1: Lying rate change in the testbed's environment.

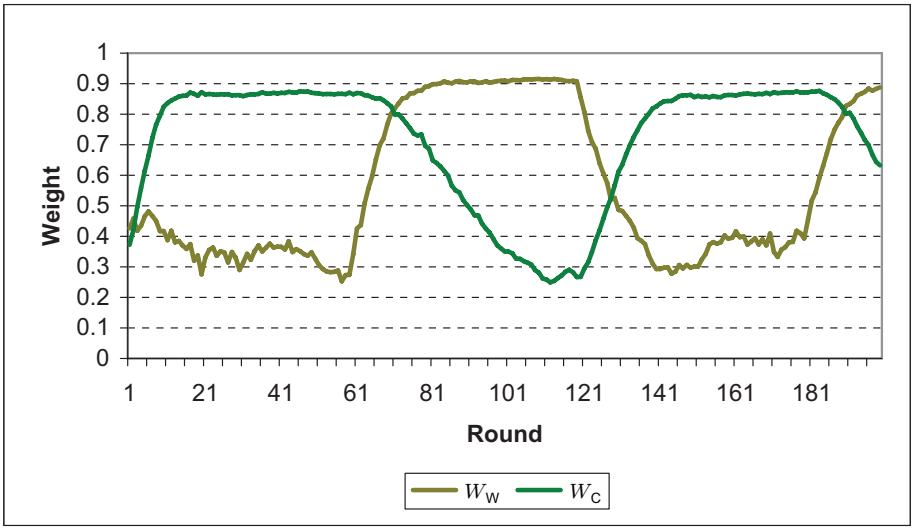

FIGURE 7.2: Component weight learning — Fixed initial weight values.

ponents) with the credibility model extension (to detect lying, Chapter 6) as their trust models. However, only one of them implements the new learning algorithm. We name that group WCRL and the other WCR ('L' signifies learning abilities). The performance of both groups are monitored as in our previous experiments. In addition, we also monitor the average component coefficients of WCRL. The component coefficients of both groups are initially set to the same value 0.5 (the mid value in the range of component coefficients). This means that both the WR and CR components are initially treated the same (the same initial weights). After the experiment, the average component coefficients of WCRL are plotted in Figure 7.2 and the performance of both groups is plotted in Figure 7.3.

Comparing the chart in Figure 7.2 (WCRL's component coefficients) with that

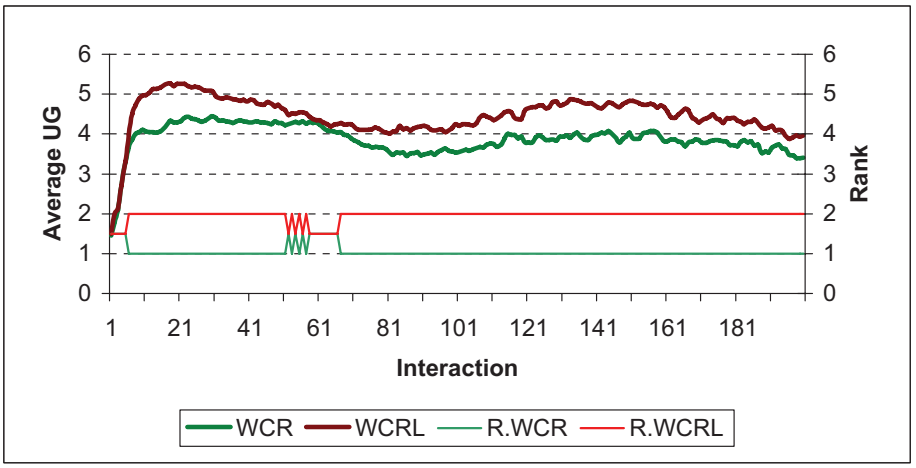

FIGURE 7.3: Learning performance — Fixed initial weight values.

in Figure 7.1 (the WR and CR lying percentages), as we would expect, we can see that the coefficient for the WR of WCRL (W_{W}) is low when the WR lying percentage is 100% (i.e. when WR performs badly) and it quickly increases to near 0.9 (high) when the WR lying percentage is changed to 0%. Similarly, the coefficient of WCRL's component (W_{C}) initially goes up because the CR component is given accurate information and produces accurate trust values. When the CR lying percentage is 100% (e.g. from round 61 to 120), W_{C} decreases because CR cannot produce accurate trust values anymore. Generally, the lines of W_{W} and W_{C} in Figure 7.2 correspond closely to the changes of the witness and referee populations in Figure 7.1 that affect the performance of the WR and CR components[2]. This shows the new learning algorithm can effectively track the changes in the performance of FIRE components and adjust the corresponding weights (i.e. the component coefficients) accordingly, even though it does not directly track the level of lying in the testbed. This shows that the method used for learning component performance is generic and it can reasonably be expected to work with other types of changes that affect the components' performance, not just the lying percentages as in our experiment. As a result, the performance plots in Figure 7.3 and the t-test rankings show that the learning algorithm helps WCRL to cope with the changes introduced in the testbed and outperforms WCR in most interactions[3].

[2]It should be noted that there is a gradual decrease of W_{C} from round 61 to 120 in Figure 7.2 that does not correspond closely to the abrupt change in the CR lying percentage (in the same period in Figure 7.1). However, this does not mean that the algorithm cannot track the changes well. In fact, since providers store references they receive, good (i.e. honest) references (from the previous period where there is no CR lying) are still stored and presented for a little while before they are all replaced by false references. Thanks to these (lingering) honest references, the performance of CR does not drop abruptly and this is reflected in the gradual decrease of W_{C} as shown in Figure 7.2.

[3]WCRL and WCR have a similar level performance in the bootstrap period (the first five rounds) and around round 60 when the lying percentages changes. These are the times in which

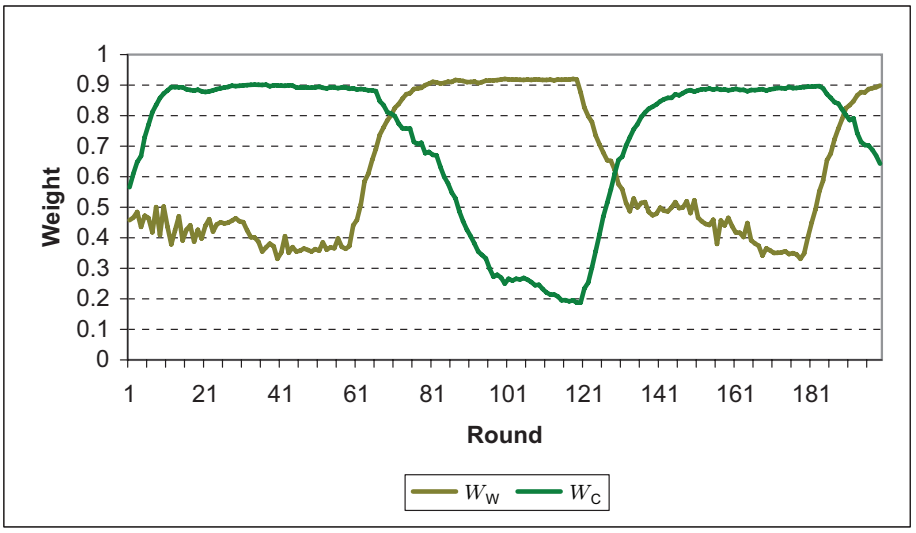

FIGURE 7.4: Component weight learning — Randomly initialised weight values.

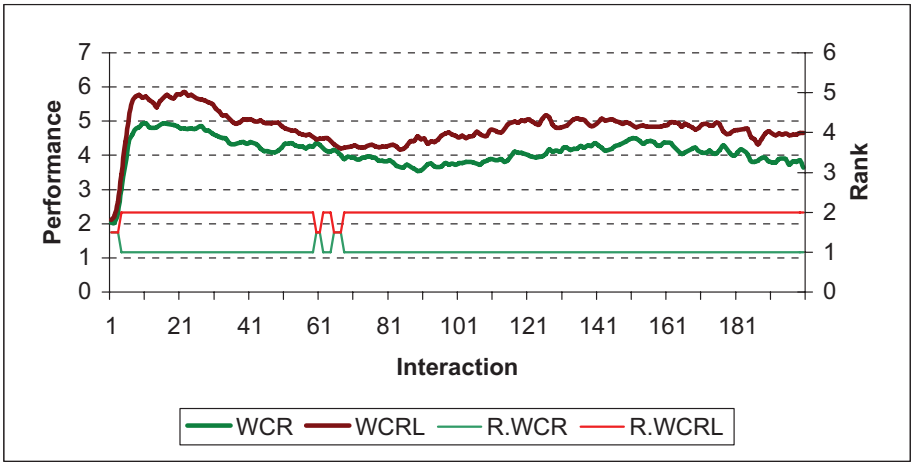

FIGURE 7.5: Learning performance — Randomly initialised weight values.

In order to ensure our learning algorithm still works in various situations, we repeat the experiment above except that the component coefficients W_W and W_C of the group are not initially fixed but randomly initialised in $[0, 1]$. This is to show that the learned parameters can still converge from random initial values. The results presented in Figures 7.4 and 7.5 are very similar to that of our first experiment and, thus, it shows our algorithm can work well even when the component coefficients are wrongly set to arbitrary values.

In the two previous experiments, the abrupt changes introduced in the testbed are useful for verifying our algorithm's effectiveness. However, they are not typical in realistic scenarios because it is unusual that all the agents (of various ownerships) to switch their behaviours at the same time. Therefore, our first experiment is

WCRL is adjusting to the new situations (hence a lower performance).

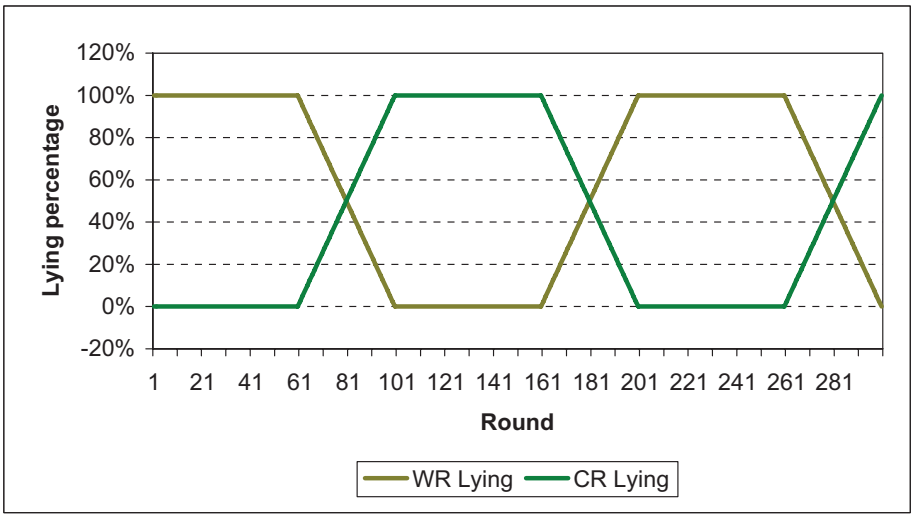

FIGURE 7.6: Gradual change of lying percentages in the testbed's environment.

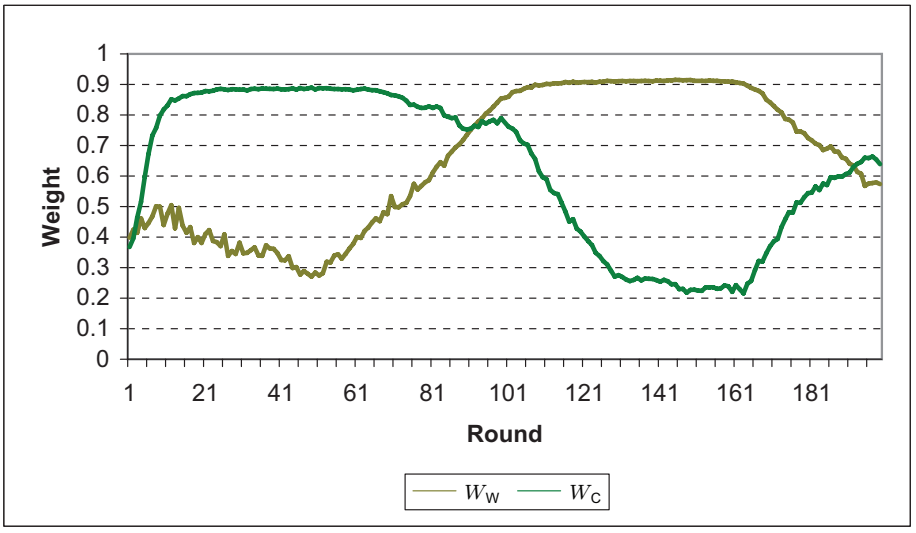

FIGURE 7.7: Component weight learning — Gradual change.

further repeated but the changes in the WR and CR lying percentages are now made gradually (Figure 7.6). This models the case in which the behaviours of the agents gradually change through time (which is more likely the case in a real-world environment). The result plotted in Figures 7.7 and 7.8 again shows that our algorithm can still track well the (gradual) changes and help FIRE to obtain better performance than without the algorithm.

7.5.2 Inaccuracy tolerance threshold learning

As discussed in Section 7.3, there are three causes to rating inaccuracy: (1) the variation of the target agent's performance, (2) the reporter's (in)ability to make

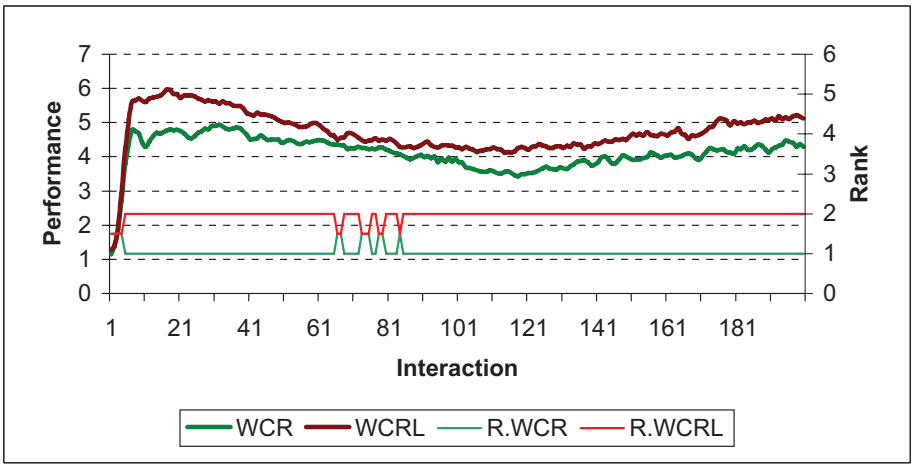

FIGURE 7.8: Learning performance — Gradual change.

accurate ratings, and (3) the reporter's intentional manipulation of the rating's value. Given these, the inaccuracy tolerance threshold ι serves as the borderline for the credibility model to distinguish between inaccurate ratings due to cause (1) (i.e. honest ratings) and those due to (2) and (3) (i.e. inaccurate/false ratings). Based on the performance deviation of the providers an agent encounters, the algorithm presented in Section 7.3 is designed to automatically adjust the inaccuracy tolerance threshold to help the credibility model make an accurate classification of honest and lying reports. Therefore, in this section, we evaluate how this algorithm performs in terms of its improvements to the classification's accuracy. In order to measure this accuracy, as our testbed knows which ratings are honest and which are fabricated, we extend the current testbed to count the number of ratings that are correctly classified; the percentage of correctly classified ratings (over the total number of third-party ratings an agent receives) is then used as the classification accuracy measure.

Generally speaking, if the performance of agents varies significantly, a low ι value would result in a high number of wrongly classified ratings, and vice versa. Therefore, in this section, the changes we introduce into the testbed to evaluate the adaptability of the ι parameter are the various levels of provider's performance variations. Since the performance of the providers in our testbed are simulated following the normal distribution (Section 4.2.1), such changes can simply be obtained by setting the standard deviation σ_{P} of the provider's performance to the desired values. In this experiment, the σ_{P} of all providers are set according to the chart in Figure 7.9 (the right y-axis); thus there are four levels of performance deviation: 0.1 (very low), 2.0, 4.0, and 6.0[4](very high).

[4]It should be noted that the performance of a provider (UG units) is in the range $[-10, 10]$.

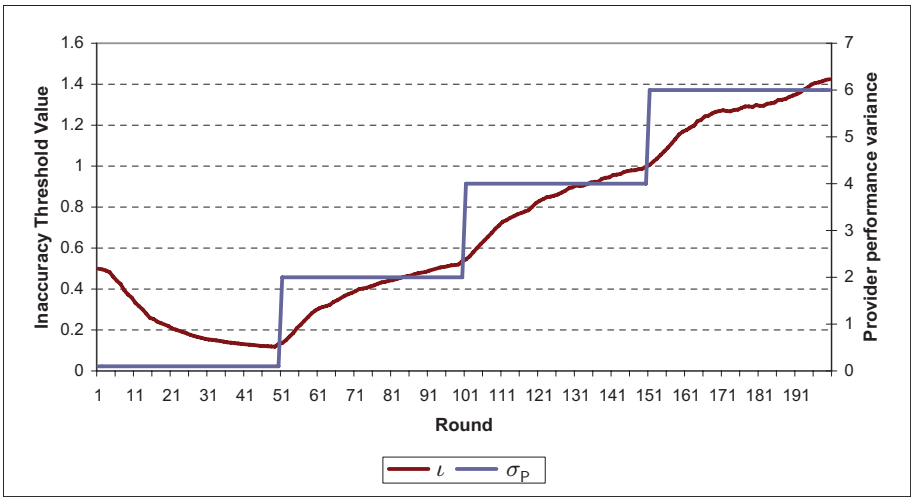

FIGURE 7.9: Variation of provider performance and the adaptation of inaccuracy threshold.

In this experiment, there are three groups of consumer agents, all of which use only FIRE's CR component with the credibility model extension[5]. The first two groups have fixed $\iota = 0.5$ and $\iota = 1.0$ (i.e. low and high inaccuracy thresholds, respectively). The agents in the third group (called ι auto) implement the algorithm in Section 7.3 to update their ι parameter automatically. There are also three profiles of referees in the consumer's population: Hon, Extr1, and Extr2 (see Section 6.2.2). In order to build a balanced (typical) referee population, the proportions of honest and lying referees are equal (50% are Hon); and among the lying referees, half of them lie mildly (25% are Extr1) and the other half are extremely inaccurate (25% are Extr2).

The results of this experiment is plotted in Figure 7.10, which shows the percentages of ratings correctly classified by each consumer group in each round. Comparing these to the changes of providers' performance deviation shown in Figure 7.9, there is a clear correlation. Specifically, the second group with a fixed $\iota = 1.0$ performs worst in the first 100 rounds because the actual variations of the providers' performance are low. Now, it appears that the group's classification accuracy is low, because it classified mildly lied ratings as honest, due to its high value of ι. It can only achieve its maximum classification accuracy when the providers' performance deviation is raised up in rounds 101 to 200. The situation is reversed

Therefore, $\sigma_P = 6.0$ is a very high level of deviation (about 32% of the time the actual performance of a provider falls outside the range $[\mu_P - 6, \mu_P + 6]$ around its mean performance μ_P).

[5]Since this experiment is to evaluate the accuracy of third-party rating classification, the IT component is not relevant. Both the WR and CR could equally well be used here. However, we choose the CR component because it has a much faster running speed than the WR component.

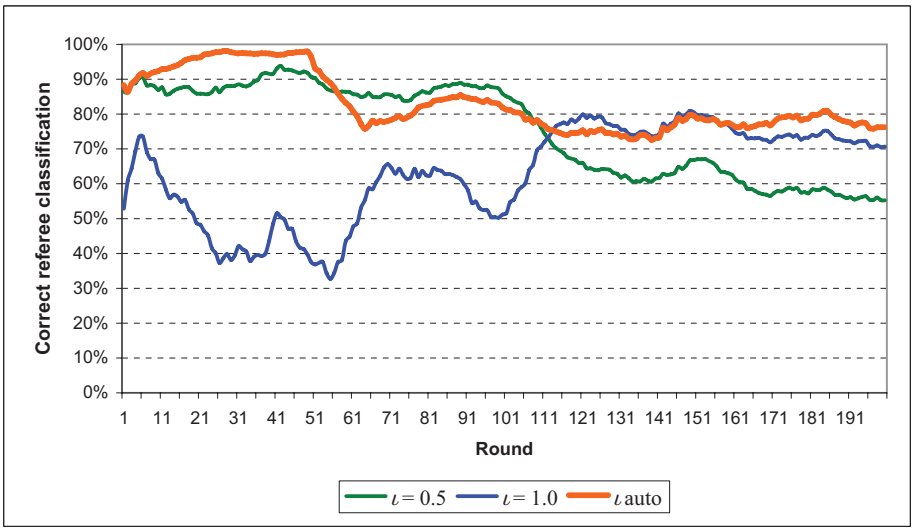

FIGURE 7.10: The accuracy of referee classification using fixed ι versus using automatically learned one.

with the first consumer group (fixed $\iota = 0.5$). Having a fixed low ι, it performs best in the first 100 rounds (when the providers' performance variation is low) and worst in the latter 100 rounds (low inaccuracy threshold, high providers' performance deviation). By having the ability to monitor the actual variations of the providers' performance, only the third group manages to maintain a high level of rating classification accuracy throughout the 200 rounds of simulation. Moreover, its accuracy is always higher than, or at least comparable to, that of the other two groups. This shows that the algorithm we introduce helps the third group to effectively adapt its ι parameter according to changes in its environment. This is confirmed by the experiment's result, in which the evolution of the automatically updated ι of the third group (i.e. the black line) plotted in Figure 7.9 corresponds well to the changes in providers' performance deviation (i.e. the grey line).

7.5.3 Default credibility learning

We now turn to the evaluation of the last learning algorithm introduced in this chapter. This algorithm is designed to monitor the general level of credibility of witnesses or referees in an environment and to update the default witness/referee credibility value. Hence, the effectiveness of this algorithm is going to be tested in situations where the general level of lying in the environment changes. In this experiment, there are two main groups of consumer agents; both of them using the CR component with the credibility model extension as their trust model. However, one of them is equipped with the default credibility learning algorithm, called CRL.

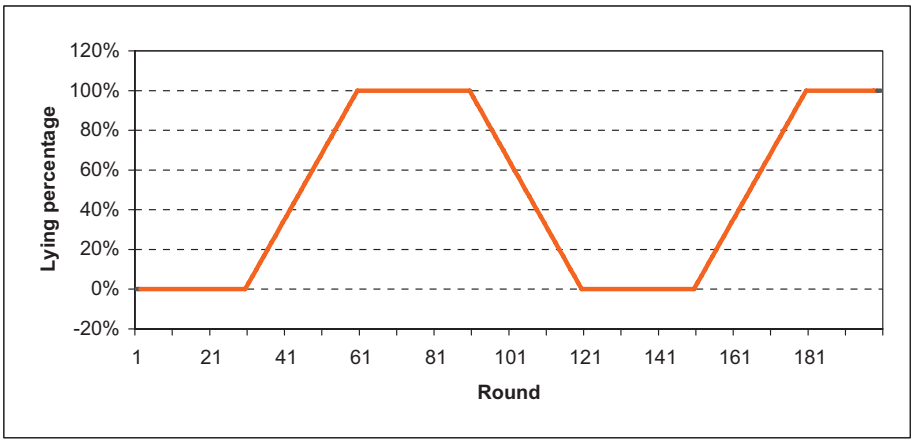

FIGURE 7.11: Learning $\mathcal{T}_{\mathsf{DRCr}}$ — CR lying percentage.

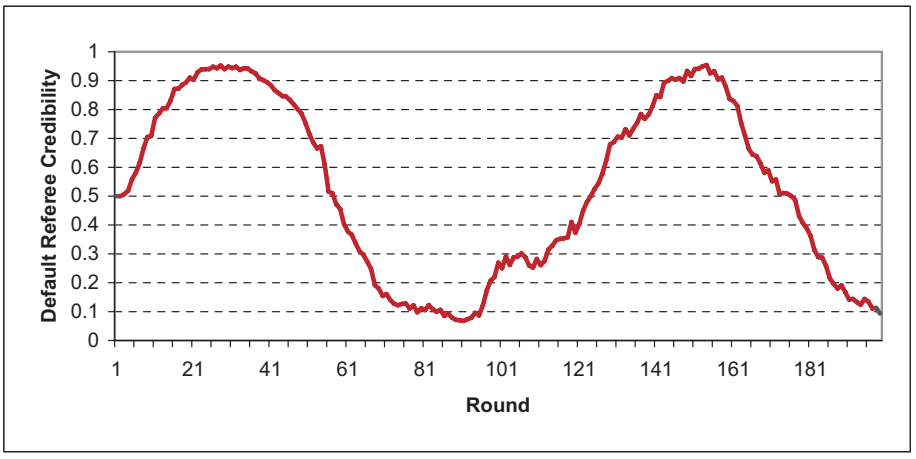

FIGURE 7.12: Learning $\mathcal{T}_{\mathsf{DRCr}}$ — $\mathcal{T}_{\mathsf{DRCr}}$'s evolution.

The other group is called CR. Since the default credibility is only used for referees that are met for the first time, new referees need to be continuously added into the testbed in order to evaluate this facet of the model. Therefore, we introduce a dummy group of consumers whose tasks are simply interacting with the providers and providing references for them. All of the consumers in this group are lying referees of either Extr1 or Extr2 type. Similar to Section 7.5.1, the level of lying in the testbed is controlled by using the CR lying percentage, which is set following the chart in Figure 7.11. To ensure the abundance of new referees in every round, 30% of the agents in the dummy group are replaced by new dummy consumer agents after each round. Since the dummy consumers do not use a trust model (they randomly select providers for interaction), we only monitor the performance of CR and CRL.

The experiment's results are presented in Figures 7.12 and 7.13. The $\mathcal{T}_{\mathsf{DRCr}}$ value of group CRL, plotted through time in Figure 7.12, closely corresponds to the CR

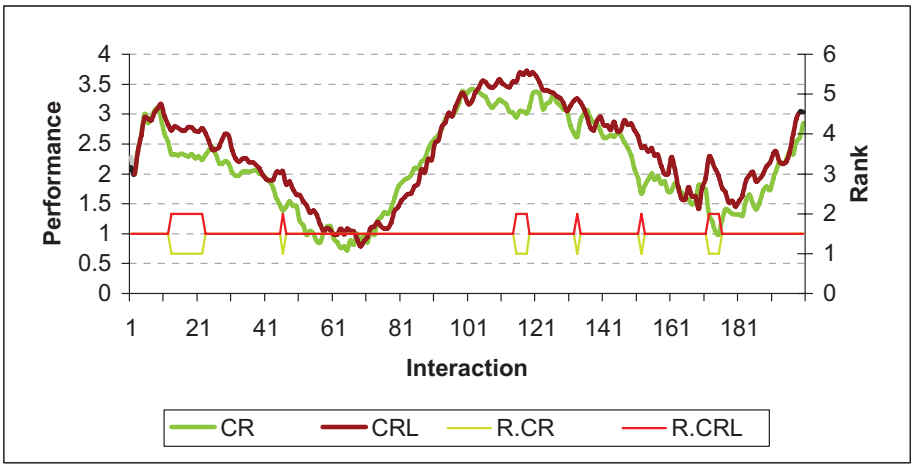

FIGURE 7.13: Learning $\mathcal{T}_{\mathsf{DRCr}}$ performance.

lying percentage in Figure 7.11. Thus, $\mathcal{T}_{\mathsf{DRCr}}$ is raised to its near maximum value (i.e. 1.0) when the CR lying percentage is minimum and vice versa. It is apparent here that the default credibility learning algorithm can effectively track the level of lying/inaccuracy in the environment and update the default referee credibility accordingly. Figure 7.13 shows that, with the new learning algorithm, CRL can outperform CR in several instances while it has similar performance with CR in the other cases. The slight performance improvement is expected because $\mathcal{T}_{\mathsf{DRCr}}$ is only used for new referees and, thus, it cannot significantly affect the overall performance of FIRE.

7.6 Summary

This chapter has further extended FIRE towards a more flexible and adaptable trust model. We have explored several learning techniques and partly automated the process of choosing the right parameters in order to ensure that FIRE operate effectively in a range of environments. Specifically, and most importantly, we devised an algorithm to monitor the performance of each of FIRE's components (based on the accuracy of their trust values). This is a novel approach that allows any changes in an agent's environment that affect the performance of one or more components (e.g. lack of ratings or changes in the quality of ratings from a particular source of information) to be indirectly detected and the weights for the corresponding components (i.e. the component coefficients) to be accordingly adjusted. In addition, appropriate algorithms are also devised for choosing the right values for the inaccuracy tolerance threshold and the default witness/referee credibility. When taken together, all these new techniques enhance FIRE's

robustness and resilience in facing unforeseen circumstances. Through empirical evaluation, we have shown that the new algorithms are in fact effective and significantly improve FIRE's adaptability.

This chapter has concluded the research work of designing a generic and adaptable trust and reputation model for applications in open MAS in the scope of this thesis. The next chapter will summarise the contributions this research has made and outline the directions for the future work.

Chapter 8

Conclusions

This chapter summarises the findings of this thesis in enabling agents in an open multi-agent system to assess the trustworthiness of their peers for selecting good interaction partners. In order to do so, a novel trust and reputation model — FIRE — was developed, which takes into account the main characteristics of an open MAS to ensure its robustness and applicability in such environments. More specifically, this thesis presents and evaluates a framework for evaluating trustworthiness of agents based on multiple sources of information: direct experience, role-based relationships, witness reports, and certified references. By using this framework, agents in an open MAS are able to obtain the trust values of their peers in most circumstances. This is possible because FIRE does not rely solely on one source of information (its four components complement and back up one another) and particularly because it exploits the high availability of the novel Certified Reputation component.

In more detail, Section 8.1 reviews the contributions of this research to the state of the art. Section 8.2 then discusses the main ways in which this research can be carried forward in the future.

8.1 Research Contributions

This thesis has presented FIRE, a novel decentralised model for trust evaluation that is specifically designed for general applications in open MAS. Before going on to FIRE's contributions to the state of the art, we recap the requirements for a trust model for applications in open MAS (discussed in Section 2.5):

R1a It should deal with the bootstrapping issue of newly joined agents.

R1b It should make use of role-based trust, interaction trust, and witness reputation when the required information for these dimensions of trust is available.

R2a Each agent should be able to collect observations and calculate the reputation values by itself.

R2b The trust model should be scalable to a large number of agents that might be present in open MAS.

R2c It should reasonably maintain its normal effective operation in situations where there are various changes in its environment.

R3 It should be adaptable to different domains of applications that an open MAS may have.

R4a It should be robust against possible lying from agents and

R4b the correlated evidence problem.

In what follows, we are going to show how these requirements are met by FIRE and highlight its novelties. In an overview, the novel mechanisms developed in this research can be classified into the following areas:

- *evaluating trust*: A generic framework is built which allows a variety of sources of trust information to be integrated to provide a collective and precise trust measure. The model is able to predict closely the behaviour of an agent. In addition, Certified Reputation, a novel type of reputation, is introduced. (Chapter 3)
- *dealing with inaccurate reports*: A model of the reporter's credibility is developed, allowing FIRE to weight third-party reports according to their provider's credibility and filter out inaccurate reporters. (Chapter 6)
- *adapting to the environment*: Learning techniques were implemented to adapt a number of FIRE's parameters to the prevailing context, allowing it to operate robustly under unforseen circumstances in the environment. (Chapter 7)

The remainder of this section discusses the results of the work in this thesis focusing on the above aspects in turn (in Sections 8.1.1, 8.1.2, and 8.1.3, respectively).

8.1.1 Evaluating trust

Enabling agents in open MAS to evaluate the trustworthiness of their peers and, thus, to be able to select reliable ones for interactions is the main aim of this thesis. To this end, FIRE was developed based on a number of potential sources of trust information. These sources include: direct experiences of an agent from its interactions, witness reports, third-party references, and rules provided by end users encoding beliefs or knowledge about the environment. This breadth is important because, given the dynamic factors that inherently exist in an open MAS, some sources may not be available, or adequate, for deducing trust. Moreover, the multiple sources are used in FIRE not only to back one another up, but also to complement one another in order to produce more precise trust values (c.f. just using one of them). In order to combine trust values derived from different sources of information, a generic framework was developed to standardise trust calculations. This includes:

- *a standardised rating form*: to represent trust information from any source which is used for exchanging trust information. A rating is not given for an agent for its performance in general, but for its performance in a particular interaction. Therefore, each rating is linked with a particular interaction, providing further contextual information (e.g. value of the interaction, service provided in that interaction) if needed. Moreover, this also eliminates the correlated evidence problem (Requirement **R4b**) because no overall opinion (i.e. opinions given based on results of more than one interaction) is exchanged.

- *a general trust formula*: to aggregate the trust information (i.e. ratings) that a trust component collects, which is used in all FIRE's components. However, depending on the information source used, each component can define its own weight function to reflect the quality of each rating taken into account.

- *reliability measures*: to produce reliability values for each trust values based on the quality of the ratings taken into account and their deviation.

This framework is generic because it works independently of any specific application and it does not rely on any assumption or information which is not widely available in an open MAS. Thus, FIRE can be instantiated and applied in a wide range of applications (Requirement **R3**).

Under this framework, each component of FIRE is developed to process trust information from each of the sources mentioned above. The components are: Interaction Trust, Role-based Trust, Witness Reputation, and Certified Reputation. First, based on the principles of Regret's Direct Trust component, the IT component is built to produce trust values from an agent's own ratings from its direct interactions. A new rating weight function is devised to calculate the reliability of a rating based on its recency. Second, a formalisation of role-based rules is presented in order for the RT to retrieve relevant rules and calculate the role-based trust based on those rules. Third, the referral process was implemented for the WR component to locate witness ratings for witness reputation calculations. Finally, and most importantly, a novel mechanism was developed for making use of third-party references in the CR component, in which the target agents obtain references themselves and present those to the evaluator when requested. The addition of this new type of reputation greatly enhances the serviceability of FIRE, allowing a trust measure to be available in most circumstances because:

- its mechanism addresses the problem of the lack of direct experience (since agents can typically collect a large number of references themselves and they are incentivised to present these to establish new trust relationships) in the IT component, and
- using the CR component, agents are freed from the various costs involved in locating witness reports (e.g. resource, time, and communication costs).

Making use of all the four components, FIRE effectively combines their particular strengths in building a robust trust measure: the reliability of direct experiences, the domain knowledge from role-based rules, and the abundance of third-party information via witness reports and certified references. Moreover, having the four sources of information at its disposal (especially certified references thanks to their high availability) means that FIRE can provide a trust measure that is sufficiently precise to be used in a wide range of situations (Requirements **R1a** and **R1b**). Obviously, there are still cases when FIRE cannot produce a trust value. Specifically, those are when a service provider newly joins the system. Hence, it does not have references about its performance and other agents do not have past experience with it. However, in a realistic scenario, in order to promote its service, that provider can join a (popular) scheme/organisation that provides quality assurance about its members' service. For example, a car dealer can obtain the title 'authorised dealer' from a car manufacturer, or a commercial site can assure its potential customers about its security reliability by showing

a 'HackerSafe' [1] certification. Such (popular) memberships (and inherently their quality assurance) can be recognised by other agents (via rules in FIRE's RT component) and, thus, helps the provider to sell its service.

In addition to the above, a notable characteristic of FIRE is that all of its mechanisms are decentralised. This means that individual agents can use FIRE to make trust evaluations without the need of a centralised authority. This is important for making FIRE compatible with the 'no central authority' of open MAS (Requirement **R2a**).

In order to verify our claims, empirical evaluation was carried out and it was demonstrated that:

- Agents using the trust measure provided by FIRE are able to select reliable partners for interactions and, thus, obtain better utility gain compared to those using no trust measure. This result was reconfirmed with various types of provider population.
- Each component of FIRE plays an important role in its operation and significantly contributes to its overall performance.
- FIRE is able to cope well with the various types of changes in an open MAS and can maintain its properties despite the dynamism possible in an environment (Requirement **R2c**).
- Although decentralised, to suit the requirements of a trust model in open MAS, FIRE still outperforms, or at worst maintains a comparable performance level with SPORAS, a centralised trust model.
- In our experiments (Chapters 5, 6, and 7), 500, 1000, and 1500 agents using FIRE have been deployed and we observe the execution time of those experiments varies linearly to the number of agents deployed. Thus, given its decentralised nature, we believe that FIRE is scalable to the large number of agents that may be present in an open MAS (Requirement **R2b**).

To sum up, FIRE satisfies all our requirements (plus Requirement **R4b**, which is dealt with in the following section) for a trust model in open MAS. Its behaviour can be customised via its set of parameters to suit a particular application. Hence, FIRE is ready to be used in real world contexts.

[1] 'HackerSafe' certifications are provided by ScanAlert (a security company, `www.hackersafe.com`) to certify that the certified sites' servers are regularly tested (by real security attacks) and shown to be hacker-proof.

8.1.2 Dealing with inaccurate reports

The ability to detect and handle inaccurate reports appropriately is critical in order for the WR and CR components to produce reliable trust values (because these components rely on third-party reports to work). This is particularly important in an open MAS where agents that are owned by various stakeholders can disseminate disinformation to their own benefit (Requirement **R4b**). To this end, this thesis presented a novel credibility model that allows FIRE to assess the reliability of information providers (i.e. reporters) and to weight, or to filter out, their information accordingly. More specifically, using our credibility model, an agent rates the credibility of a reporter based on the difference between the reports it receives and the *actual* interaction result it observes later. Hence, reporters' credibility is not objectively assessed based on how honest they are in revealing the interaction result they received, but rather it is subjectively judged based on their capability to give reports close to the actual results that a particular agent would receive. By so doing, an agent can detect not only inaccurate/false reports, but also honest, but useless, reports that result from the different views of the reporters. For example, one reporter may receive preferential treatment from a particular service provider and give out good (and honest) ratings about this provider. Such ratings, though honest and accurate (in the view of that reporter), are not useful for other agents because they would receive only normal treatment from that provider. By taking an agent's individual situation (i.e. the actual performance it receives during interactions) into account, our credibility model can deal with cases similar to the one in this example appropriately. This is the main difference that separates our approach from others in the literature where reporters are usually judged on their honesty (e.g. [Sabater, 2003], [Sen and Sajja, 2002]).

Through empirical evaluation, our credibility model was shown to be effective in handling inaccurate reports, enabling the WR and CR components to maintain a stable level of performance in a wide range of situations where various levels of inaccuracy were introduced. In particular, the WR and CR components were extended with the credibility model and tested in environments where marginally inaccurate and extremely inaccurate reporters were introduced at various percentages. The results show that by using the credibility model, the WR and CR components are able to maintain their performance at a comparable level to that in honest environments in all the experiments where the percentages of inaccurate reporters are less than 50%. Furthermore, compared to SPORAS, the extended WR and CR components are significantly better in dealing with inaccurate reports

and their performance degrades more gracefully when the level of inaccuracy in the environment increases.

8.1.3 Adapting to the environment

Given the continuously changing nature of an open MAS, it is desirable that a trust model can cope well with unforseen changes that might take place in the environment and that it is able to maintain its normal effective operation under such circumstances. Therefore, learning techniques have been implemented in order to enhance FIRE's adaptivity to a number of possible changes in an open MAS. In particular, FIRE is extended to weight the trust values produced by its components according to the components' performance. In order to do so, the performance (i.e. the accuracy) of each component is continuously monitored during FIRE's operation. Should there be extraneous factors that affect the performance of one or more of FIRE's components (e.g. the lack of ratings or changes in the quality of ratings from a particular source of information), the corresponding component coefficients are automatically adjusted in an appropriate manner. By so doing, the actual reliability of each of FIRE's component is taken into account when FIRE calculates the overall trust values from its components.

In addition, algorithms are also devised for choosing the right values for the inaccuracy threshold and the default witness/referee credibility (used in the credibility model). When taken together, all these techniques not only enhance FIRE's robustness and resilience in facing unforseen changes, but also reduce FIRE's administrative burden since these parameters, if initially wrongly set, can be re-adjusted automatically by the model without requiring human intervention. Finally, empirical evaluation has shown that the learning techniques implemented are effective and significantly improve FIRE's adaptability.

8.2 Future Directions

Although FIRE makes a number of advances to the state of the art, there are still a number of ways in which this work can be further extended. These are now detailed in the remainder of this subsection.

First, in the RT component, role-based rules need to be entered by agent owners and when these rules are matched the target agent will be given a pre-determined

trust value (Section 3.4). However, this can be extended to make use dynamic rules that give trust values based on the prevailing context. An example rule is "if a provider offers an x-year guarantee for its products then its product reliability is $\min\{0.2 \times (x-1), 1.0\}$", or another one is "if a provider shows y references about its previous successful interactions whose value exceeds £1000 then its capability is $\min\{0.1 \times y, 1.0\}$". Here, the first rule means that if a provider does not offer a product guarantee at all ($x = 0$) then its product reliability is likely to be bad (trust value is -0.2), a 1-year guarantee is the standard (a neutral trust value, 0.0), and over that, each additional guarantee year offered increases its product's reliability by 0.2 up to the maximum of 1.0. Similarly, the second rule means that the more high-value interactions (i.e. larger than £1000) a provider has finished successfully, the more capable that provider is (here each such transaction adds 0.1 to the provider's capability rating up to the maximum of 1.0). Such rules are here called "semi-automatic" because they still need to be produced by humans, but the trust values they give are not fixed, rather they are calculated according to the context in which they are applied. These rules are more powerful and expressive than those that are currently accommodated by the RT component, allowing the agent designer to encode far more complex rules than their current role-based counterparts. In order to process and apply such semi-automatic rules, the current RT component needs to be significantly extended and new mechanisms are also needed in order for an agent to understand and to extract data from contextual information. However, such an extension would greatly enhance the RT component and, as a result, the FIRE model, in terms of its customisability since it allows a much wider range of domain knowledge to be encoded and used.

The next potential extension is considered with the CR component. The CR mechanism presented in this thesis allows a target agent to actively establish trust relationships with its potential interaction partners by presenting references about its past performance when requested. Although this works well, this process can be further improved by introducing an element of negotiation into it. For example, having examined the references provided by the target agent b, agent a may not be satisfied with the quality or the relevance of these references (e.g. because they are about a different kind of service than the one a needs, because they are too old, or because the values of the corresponding interactions are insignificant). In that case, agent a can explain to b the reasons why the provided references cannot be accepted and ask b to provide more relevant ones. From the reasons given by a, agent b can select and present another set of references and tell a why it should believe in those new ones; or b can argue with a that the ones it provided

are actually significant and relevant to it (e.g. because a particular reference r was given by a big name company that has a very strict set of standards, and so on). In this example, compared to our original CR mechanism, the way CR is calculated from provided references does not change, but a trust relationship between a and b can be easier to establish because both the agents have the opportunity to understand the needs of each other and also the opportunity to correct possible misunderstandings (e.g. from the arguments provided by b, a might adjust its initial weight for reference r). One of the main concerns over the use of CR is that the references provided by the target agent itself might be misleading because it can choose the best references to present. To overcome this, argumentation-based negotiation [Rahwan et al., 2004] could be used to clear up the evaluator's possible doubts on the references it receives. In order to enter into such a negotiation, however, an agent (or the trust model) needs to be able to evaluate contextual information associated with a reference and to understand and generate appropriate arguments. This requires an investigation into the possible ways to apply the work from the area of argumentation-based negotiation into the CR component.

Understanding contextual information comes up as one of the main additional requirements for FIRE in both of the above suggested extensions. However, achieving this is not a simple task because contextual information (which is indefinite and varies significantly) needs not only to be provided, but also to be expressed in a standardised way. To this end, a suitable ontology[2] [Smith, 2003] needs to be developed to be used as a standardised basis for trust information exchange between agents. Although the development of an ontology specifically for trust information does not directly extend the work in this thesis, its existence would greatly benefit FIRE in several ways. This is because it allows contextual information to be exchanged to accompany ratings (in the interaction component of a rating), which, in turn, opens up a wide range of possible enhancements in trust evaluation as demonstrated in our examples above.

Coming back to the CR component, it should be noted that when a referee gives its references to an interaction partner, it effectively surrenders its privacy with respect to how it values that partner's performance. This may lead to various possible reactions of that partner (e.g. it may retaliate against the referee for a bad reference or it may treat the referee differently the next time to get a better

[2]The term *ontology* refers to a data model that represents a specific part of the real-world and is used to reason about the relationships of objects in the world. Ontologies contain abstract representations of objects and their relationships [Smith, 2003].

reference). However, due to the vast number of possibilities in the reactions of both agents (i.e. the referee and the referred agent) and the limited scope of this thesis, the effects of giving up privacy in the CR component have not been considered. However, a fuller investigation on this is needed in order to understand the possible problems and to put appropriate measures in place should they have undesirable effects on the performance of the CR component.

In the RT component, when looking for relevant rules to apply for a given pair of agents a and b, it is assumed that information about the roles of a and b (and, thus, their relationships) is already given in some way. Although some types of role information may actually be readily available from an agent's knowledge (e.g. b is owned by the same organisation, b is the seller, b is a financial institution), there are many other cases where such knowledge is less apparent and is not available. For example, given a rule that says "if b competes with w then opinions of b about w may not be reliable", it is not always straighforward to determine how the 'competitor' roles of b and w can be deferred. Since there are no mechanisms in FIRE to determine the relationships of agents (and their roles), the types of roles that can be used in such rules are thus limited (to the simple and apparent ones as mentioned above). Therefore, developing a model for agent relationship identification is also a potential extension to FIRE. Given the ability to determine high-level relationships, such as competition, collaboration, dependency between agents, the range of rules that could be used in FIRE would be greatly expanded, allowing a wider range of knowledge (than currently possible) to be encoded and used by FIRE. The model presented by Ashri et al. [2005] is an example of such relationship identification models and might provide a promising point of departure for this line of enquiry.

All the work above focuses on extending the capability of FIRE. In a broader view, it is not realistic for an agent a to select another agent b for an interaction merely based on the trustworthiness of b. Agent a also needs to consider other factors such as the cost requested by b in comparison with others. For example, if there is another agent c which is marginally less trustworthy than b, but requests a far lower price, b might not be the best choice. In another example, if the service that a needs is very critical to a and its failure would mean a catastrophy, b might be selected because a could not accept the risk of choosing a lesser provider. Hence, a decision model that takes into account the cost, the utility gain, the risk (which can be calculated from the potential chosen agent's trust values and the corresponding reliability values) involved in an agent's delegation action is clearly needed. Simply comparing the trustworthiness of agents may not always suffice.

Appendix A

Relevant Trust Models

Of the trust models reviewed in Chapter 2, SPORAS [Zacharia and Maes, 2000] was later used in our experiments (Chapters 5 and 6) as a benchmark and Regret's Direct Trust component [Sabater, 2003] was reused in FIRE. Thus, their operations need to be described in greater detail. However, since our review in Chapter 2 mainly focuses on the characteristics of trust models, these were not discussed. Therefore, this appendix provides a more detailed survey on the operations of SPORAS and Regret (Sections A.1 and A.2, respectively).

A.1 SPORAS

SPORAS is a centralised trust model similar to the online reputation models used on eBay and Amazon (which manage the reputation of all its users in a centralised manner). However, it extends those online reputation models by introducing a new method for rating aggregation. Specifically, instead of storing all the ratings, each time a rating is received it updates the reputation of the involved party using an algorithm that satisfies the following principles:

1. New users start with a minimum reputation value and they build up reputation during their activity on the system.
2. The reputation value of a user never falls below the reputation of a new user.
3. After each transaction, the reputation values of the involved users are updated according to the feedback provided by the other parties, which reflect their trustworthiness in the latest transaction.

4. Users with very high reputation values experience much smaller rating changes after each update.

5. Ratings must be discounted over time so that the most recent ratings have more weight in the evaluation of a users's reputation.

A new user a in SPORAS starts with a reputation value $R_0^a = 0$. R_i^a is used to denote user a's reputation at time i. The range of a reputation value in SPORAS is $D = [0, 3000]$. The reputation rating for a reported by user b at time i is denoted by $W_i^{a,b}$ and ranges from 0.1 to 1.0. The formula for updating reputation at R_i^a at time i upon receiving the rating $W_i^{a,b}$ is as follows:

$$R_i^a = R_{i-1}^a + \frac{1}{\theta} \cdot \Theta(R_{i-1}^a) \cdot R_i^b \cdot (W_i^{a,b} - E_i^a) \tag{A.1}$$

where $\theta > 1$ is the effective number of ratings considered, E_i^a is the expected value of $W_i^{a,b}$ (i.e. the expected performance of a) and $\Theta(R_{i-1}^a)$ is the damping function defined to slow down reputation changes of a user with a very high reputation (see the principles of SPORAS above). E_i^a and $\Theta(R_{i-1}^a)$ are given in the following formula:

$$E_i^a = \frac{R_{i-1}^a}{D} \tag{A.2}$$

$$\Theta(R_{i-1}^a) = 1 - \frac{1}{1 + e^{\frac{-\left(R_{i-1}^a - D\right)}{\sigma}}} \tag{A.3}$$

where σ is chosen so that function Θ remains above 0.9 for all users whose reputation is below $\frac{3}{4}$ of D.

In addition, SPORAS also introduces a reliability measure based on the reputation deviation (RD) of the estimated reputations. The reputation deviation of user a at time i is calculated as follows:

$$(RD_i^a)^2 = \frac{\left\lfloor \lambda \cdot (RD_{i-1}^a)^2 + \left(R_i^b \cdot (W_i - E_i)\right)^2 \right\rfloor}{\theta} \tag{A.4}$$

where λ is the forgetting factor computed from the effective number of ratings considered:

$$\lambda = \frac{\theta - 1}{\theta} \tag{A.5}$$

Hence, the recursively estimated RD of a user is an indication of the predictive power of SPORAS for that user's reputation. A high RD can mean either that the user has not been active enough to be able to make a more accurate prediction for his/her reputation, or that the user's behaviour has a lot of variation. The initial

value of *RD* is set to $\frac{D}{10}$.

A.2 Regret

Regret is a reputation model in which the trust evaluation process is decentralised. Employing Regret, each agent is able to evaluate the reputation of others by itself. In order to do so, each agent rates its partner's performance after every interaction and records its ratings in a local database. The relevant ratings will be queried from this database when trust evaluation is needed. The trust value derived from those ratings is termed *direct trust*.

In more detail, in order to calculate the direct trust of agent b, agent a retrieved its past ratings about b's performance. The set of those ratings is called R[1]. Then the direct trust of a to b, denoted by $DT_{a\to b}$, is calculated as follows:

$$DT_{a\to b} = \sum_{r_i \in R} \rho(t, t_i) \cdot r_i \tag{A.6}$$

where r_i is a rating value in the set $\mathcal{R}$, and $\rho(t, t_i)$ is a normalised weight value that gives higher values to ratings of more recent interactions:

$$\rho(t, t_i) = \frac{f(t_i, t)}{\sum_{r_i \in R} f(t_i, t)} \tag{A.7}$$

$$f(t_i, t) = \frac{t_i}{t} \tag{A.8}$$

where t is the current time and t_i is the time the rating r_i is recorded. However, this weight function has some shortcomings on time granularity control. Actually, after being factored, Equation A.7 is equivalent to:

$$\rho(t, t_i) = \frac{t_i}{\sum_{r_i \in \mathcal{R}} t_i} \tag{A.9}$$

Therefore, the weight function depends only on the time values of a particular set of ratings, rather than the recency of those ratings in comparison with the current time t. For example, assume that r_1 and r_2 are ratings about the interactions that took place at time $t_1 = 1$ and $t_2 = 2$ respectively, and that $R = \{r_1, r_2\}$. Equation A.9 will give $\rho(t, t_1) = \frac{1}{3}$ and $\rho(t, t_2) = \frac{2}{3}$. Hence the difference of the

[1]Due to the limited scope of this thesis, Regret and its notation are simplified here. However, the main ideas of the model are still retained.

weights for r_1 and r_2 is $\frac{1}{3}$ in the interaction trust calculation. Suppose that r'_1 and r'_2 are completely the same ratings to r_1 and r_2 except that $t'_1 = 100$ and $t'_2 = 101$. The same calculations will give a weight difference that is now $\frac{1}{10100}$. This vast change in weight differences is not desirable given that the rating time difference in both cases is the same (1 time unit, e.g. second). Moreover, given the same set of ratings, the weight function produces the same value regardless of the current time t[2] (i.e. the time of calculating the interaction trust).

Like SPORAS, every trust value in Regret comes with a reliability value that reflects the confidence of Regret in that trust value. This reliability value is calculated from a combination of two measures that are based on the number of ratings in the set R and the deviation of those ratings. Regret defines an intimate level of interactions, denoted by itm, that represents the minimum number of ratings needed for a close relationship. The reliability degree increases until $|R|$ reaches this number. After that, more interactions will not increase reliability. The measures (*No* for number of ratings and Dv for deviation of ratings) and the reliability for direct trust, denoted by $DTRL_{a\to b}$, are specified in the following formula:

$$No(R) = \begin{cases} \sin(\frac{|R|\cdot\pi}{2\cdot itm}) & |R| \leq itm \\ 1 & |R| > itm \end{cases} \tag{A.10}$$

$$Dv(R) = \sum_{r_i \in R} \rho(t, t_i) \cdot |r_i - DT_{a\to b}| \tag{A.11}$$

$$DTRL_{a\to b} = No(R) \cdot (1 - Dv(R)) \tag{A.12}$$

In addition, agents are further assumed to be willing to share their opinion about others. Based on this, Regret develops a witness reputation component along with a sophisticated method for aggregating witness reports taking into account the possibility of dishonest reports. The operation of this component depends on the social network built up by each agent. In particular, Regret uses a social network to find witnesses, to decide which witnesses will be consulted, and how to weight those witnesses' opinions. However, Regret does not specify how such social networks can be built, and, thus, this component is of limited use.

Besides direct trust and witness reputation, Regret also introduces the concepts of neighbourhood reputation and system reputation. The former is calculated from

[2]Suppose that r_1 and r_2 are experiences of some person and the time unit used is *year*. If the current time is year 2, r_2 should have much more influence on the trust decision of that person than r_1 since it is much more recent. If the current time is year 20, both r_1 and r_2 are too old, and they should have a similarly low levels of influence on his current trust decision.

the reputation of the target's neighbour agents based on fuzzy rules. However, this again requires a social network to work. The latter is a mechanism to assign default trust values to the target agent based on its social role in an interaction (e.g. buyer, seller). This is only useful if additional domain specific information is available.

Bibliography

Alfarez Abdul-Rahman and Stephen Hailes. Supporting trust in virtual communities. In *Proceedings of the 33rd Hawaii International Conference on System Sciences*, volume 6. IEEE Computer Society Press, Jan 2000.

Marshall D. Abrams and Michael V. Joyce. Trusted system concepts. *Computers & Security*, 14(1):45–56, 1995.

Ronald Ashri, Sarvapali D. Ramchurn, Jordi Sabater, Mike Luck, and Nicholas R. Jennings. Trust evaluation through relationship analysis. In *Proceedings of the 4nd International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, volume 3, pages 1005–1012, Utrecht, Netherlands, 2005.

K. Suzanne Barber and Joonoo Kim. Soft security: Isolating unreliable agents from society. In *5th Workshop on Deception, Fraud and Trust in Agent Societies, AAMAS 2002*, pages 8–17, 2002.

Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, May 2001.

Felix Brandt. A verifiable, bidder-resolved auction protocol. *Workshop on Deception, Fraud and Trust in Agent Societies, AAMAS 2002*, pages 18–25, 2002.

Javier Carbo, José M. Molina, and J. Davila. Trust management through fuzzy reputation. *International Journal of Cooperative Information Systems*, 12(1): 135–155, 2003.

Cristiano Castelfranchi and Rino Falcone. Principles of trust for MAS: cognitive anatomy, social importance, and quantification. In Yves Demazeau, editor, *Proceedings of the Third International Conference on Multi-Agent Systems*, pages 72–79, Paris, July 1998. IEEE Computer Society.

Cristiano Castelfranchi and Rino Falcone. Social trust: A cognitive approach. In Cristiano Castelfranchi and Yao-Hua Tan, editors, *Trust and Deception in Virtual Societies*, pages 55–90. Kluwer Academic Publishers, 2001.

Cristiano Castelfranchi and Yao-Hua Tan. The role of trust and deception in virtual societies. In *34th Annual Hawaii International Conference on System Sciences*, volume 7. IEEE Computer Society, 2001.

Adam Cheyer and David Martin. The Open Agent Architecture. *Journal of Autonomous Agents and Multi-Agent Systems*, 4(1):143–148, March 2001.

Paul R. Cohen. *Empirical Methods for Artificial Intelligence*. The MIT Press, 1995.

Partha Dasgupta. Trust as a commodity. In Diego Gambetta, editor, *Trust: Making and Breaking Cooperative Relations*, pages 49–72. Department of Sociology, University of Oxford, electronic edition, 2000.

Keith Decker, Katia Sycara, and Mike Williamson. Middle-agents for the internet. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence*, Nagoya, Japan, 1997.

Morris H. DeGroot and Mark J. Schervish. *Probability and Statistics*. Addison-Wesley, 2002.

Theo Dimitrakos and Juan Bicarregui. Towards modelling e-Trust. In *Proceedings of the 3rd Panhellenic Logic Symposium*, 2001.

Rino Falcone, Suzanne Barber, Larry Korba, and Munindar Singh, editors. *Trust, Reputation and Security: Theories and Practice*, volume 2631 of *Lecture Notes in Computer Science*, 2003. Springer-Verlag Berlin Heidelberg. AAMAS 2002 International Workshop - Bologna, Italy, July 15, 2002 - Selected and Invited Papers.

Rino Falcone, Munindar Singh, and Yao-Hua Tan, editors. *Trust in Cybersocieties: Integrating the Human and Artificial Perspectives*, volume 2246 of *Lecture Notes in Computer Science*, 2001. Springer-Verlag Berlin Heidelberg.

FIPA. FIPA Abstract Architecture Specification. `http://www.fipa.org/specs/fipa00001/`, 2002. Foundation for Intelligent Physical Agents.

Ian Foster, Carl Kesselman, and Steven Tuecke. The anatomy of the Grid: Enabling scalable virtual organizations. *The International Journal of High Performance Computing Applications*, 15(3):200–222, Fall 2001.

Diego Gambetta. Can we trust trust? In Diego Gambetta, editor, *Trust: Making and Breaking Cooperative Relations*, pages 213–237. Department of Sociology, University of Oxford, electronic edition, 2000a.

Diego Gambetta. *Trust: Making and Breaking Cooperative Relations*. Department of Sociology, University of Oxford, electronic edition, 2000b.

Tyrone Grandison and Morris Sloman. A survey of trust in internet applications. *IEEE Communications Surveys & Tutorials*, 3(4), 2000.

Robert H. Guttman, Alexandros G. Moukas, and Pattie Maes. Agent-mediated electronic commerce: A survey. *The Knowledge Engineering Review*, 13(2): 147–159, 1998.

Minghua He, Nicholas R. Jennings, and Ho-fung Leung. On agent-mediated electronic commerce. *IEEE Trans on Knowledge and Data Engineering*, 15(4): 985–1003, 2003.

Morten Hertzum, Hans H.K. Andersen, and Verner Andersen. Trust in information sources: Seeking information from people, documents, and virtual agents. *Interacting with Computers*, 14(5):575–599, 2002.

Ming-Chih Hsu and Von-Wun Soo. A secure multi-agent Vickrey auction scheme. *Workshop on Deception, Fraud and Trust in Agent Societies, AAMAS 2002*, pages 86–91, 2002.

Nicholas R. Jennings. An agent-based approach for building complex software systems. *Communications of the ACM*, 44(4):35–41, April 2001.

Nicholas R. Jennings, Katia Sycara, and Michael Wooldridge. A roadmap of agent research and development. *Autonomous Agents and Multi-Agent Systems Journal*, 1(1):7–38, 1998.

Christian Jensen, Stefan Poslad, and Theo Dimitrakos, editors. *Trust Management*, volume 2995 of *Lecture Notes in Computer Science*, 2004. Springer-Verlag Berlin Heidelberg. Proceedings of the Second International Conference, iTrust 2004 Oxford, UK, March 29 - April 1, 2004.

Audun Jøsang, Roslan Ismail, and Colin Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 2006. (in press).

Radu Jurca and Boi Faltings. An incentive compatible reputation mechanism. In *Proceedings of the IEEE Conference on E-Commerce CEC03*, Jun 24-27 2003.

Leslie Pack Kaelbling, Michael L. Littman, and Andrew W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

Martha L. Kahn and Cynthia Della Torre Cicalese. The CoABS Grid. *Goddard-/JPL Workshop on Radical Agent Concepts*, 2002.

Marvin Karlins and Herber I. Abelson. *Persuasion, how opinion and attitudes are changed*. Crosby Lockwood & Son, 1970.

Matthias Klusch and Katia Sycara. Brokering and matchmaking for coordination of agent societies: A survey. In Andrea Omicini, Franco Zambonelli, Matthias Klusch, and Robert Tolksdorf, editors, *Coordination of Internet Agents*. Springer, 2001.

Henry E. Kyburg, Jr. Bayesian and non-bayesian evidential updating. *Artificial Intelligence*, 31(3):271–293, Mar 1987.

Stephen Paul Marsh. *Formalising Trust as a Computational Concept*. PhD thesis, University of Stirling, 1994.

David L. Martin, Adam J. Cheyer, and Douglas B. Moran. The open agent architecture: A framework for building distributed software systems. *Applied Artificial Intelligence: An International Journal*, 13(1-2):91–128, 1999.

E. Michael Maximilien and Munindar P. Singh. Reputation and endorsement for web services. *ACM SIGEcom Exchanges*, 3(1):24–31, 2002.

Deborah L. McGuinness and Paulo Pinheiro da Silva. Inference web: Portable and shareable explanations for question answering. In *New Directions for Question Answering*, pages 67–71. The AAAI Press, Mar 2003.

Lik Mui, Mojdeh Mohtashemi, and Ari Halberstadt. A computational model of trust and reputation. In *Proceedings of the 35th Hawaii International Conference on System Science*, pages 280–287, 2002.

Timothy J. Norman, Alun Preece, Stuart Chalmers, Nicholas R. Jennings, Michael Luck, Viet D. Dang, Thuc D. Nguyen, Vikas Deora, Jianhua Shao, W. Alex Gray, and Nick J. Fiddian. Agent-based formation of virtual organisations. *Knowledge-Based Systems*, 14(2-4):103111, 2004.

Iyad Rahwan, Sarvapali D. Ramchurn, Nicholas R. Jennings, Peter McBurney, Simon Parsons, and Liz Sonenberg. Argumentation-based negotiation. *The Knowledge Engineering Review*, 18(4):343–375, 2004.

Sarvapali D. Ramchurn, T. Dong Huynh, and Nicholas R. Jennings. Trust in multi-agent systems. *The Knowledge Engineering Review*, 19(1):1–25, March 2004.

Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1st edition, 1995.

Jordi Sabater. *Trust and Reputation for Agent Societies*. Phd thesis, Universitat Autònoma de Barcelona, 2003.

Jordi Sabater and Carles Sierra. REGRET: A reputation model for gregarious societies. In *Fourth Workshop on Deception Fraud and Trust in Agent Societies*, pages 61–70, Montreal, Canada, 28 May - 19 June 2001.

Jordi Sabater and Carles Sierra. Social ReGret, a reputation model based on social relations. *SIGecom Exchanges, ACM*, 3.1:44–56, 2002.

Gerald Salton and Michael McGill. *An Introduction to Modern Information Retrieval*. McGraw-Hill, New York, 1983.

Michael Schillo, Michael Rovatsos, and Petra Funk. Using trust for detecting deceitful agents in artificial societies. *Special Issue of the Applied Artificial Intelligence Journal on "Deception, Fraud and Trust in Agent Societies"*, 14(8): 825–848, 2000.

Sandip Sen and Neelima Sajja. Robustness of reputation-based trust: Boolean case. In *Proceedings of the First Intenational Joint Conference on Autonomous Agents and Multiagent Systems*, volume 1, pages 288–293, 2002.

Halvard Skogsrud, Boualem Benatallah, and Fabio Casati. Model-driven trust negotiation for web services. *IEEE Internet Computing*, 7(6):45–52, 2003.

Barry Smith. Ontology. In Luciano Floridi, editor, *The Blackwell Guide to Philosophy of Computing and Information*, pages 155–166, Oxford, 2003. Blackwell.

W. T. Luke Teacy, Jigar Patel, Nicholas R. Jennings, and Michael Luck. TRAVOS: Trust and reputation in the context of inaccurate information sources. *Autonomous Agents and Multi-Agent Systems*, 12(2):183–198, 2006.

Andrew Whitby, Audun Jøsang, and Jadwiga Indulska. Filtering out unfair ratings in bayesian reputation systems. In *Proceedings of the Seventh International Workshop on Trust in Agent Societies*, 2004.

H. Chi Wong and Katia Sycara. Adding security and trust to multi-agent systems. *Applied Artificial Intelligence*, 14(9):927–941, 2000.

Michael Wooldridge. *An Introduction to Multiagent Systems*. John Wiley & Sons, 2002.

Pinar Yolum and Munindar P. Singh. Service graphs for building trust. In *International Conference on Cooperative Information Systems (CoopIS)*, volume 3290 of *Lecture Notes in Computer Science*, pages 509–525, 2004.

Bin Yu and Munindar P. Singh. An evidential model of distributed reputation management. In *Proceedings of First International Joint Conference on Autonomous Agents and Multi-Agent Systems*, volume 1, pages 294–301. ACM Press, 2002.

Bin Yu and Munindar P. Singh. Detecting deception in reputation management. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 73–80. ACM Press, July 2003a.

Bin Yu and Munindar P. Singh. Searching social networks. In *Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 65–72. ACM Press, 2003b.

Giorgos Zacharia and Pattie Maes. Trust management through reputation mechanisms. *Applied Artificial Intelligence*, 14(9):881–908, 2000.

Franco Zambonelli, Nicholas R. Jennings, Andrea Omicini, and Michael Wooldridge. Agent-oriented software engineering for internet applications. In Andrea Omicini, Franco Zambonelli, Matthias Klusch, and Robert Tolksdorf, editors, *Coordination of Internet Agents*, pages 326–346. Springer Verlag, 2001.

Philip R. Zimmermann. *The Official PGP User's Guide*. MIT Press, Cambridge, MA, 1995.