IIIA
Institut d'Investigació en
Intel·ligència Artificial

UAB
Universitat Autònoma
de Barcelona

Trust Alignment and Adaptation:
Two Approaches for Talking about Trust in Multi-Agent Systems

Andrew Koster

12th March 2012

Dissertation submitted to obtain the degree:
Doctor en Informàtica
(Ph.D. in Computer Science)

Advisors:
Dr. Jordi Sabater-Mir and
Dr. Marco Schorlemmer

Departament de Ciències de la Computació — Escola d'Enginyeria
Universitat Autònoma de Barcelona

Contents

List of Figures

Abstract

In open multi-agent systems, trust models are an important tool for agents to achieve effective interactions; however, trust is an inherently subjective concept, and thus for the agents to communicate about trust meaningfully, additional information is required. This thesis focuses on Trust Alignment and Trust Adaptation, two approaches for communicating about trust.

The first approach is to model the problem of communicating trust as a problem of alignment. We show that currently proposed solutions, such as common ontologies or ontology alignment methods, lead to additional problems, and propose trust alignment as an alternative. We propose to use the interactions that two agents share as a basis for learning an alignment. We model this using the mathematical framework of Channel Theory, which allows us to formalise how two agents' subjective trust evaluations are related through the interactions that support them. Because the agents do not have access to each other's trust evaluations, they must communicate; we specify relevance and consistency, two necessary properties for this communication. The receiver of the communicated trust evaluations can generalise the messages using θ-subsumption, leading to a predictive model that allows an agent to translate future communications from the same sender.

We demonstrate this alignment process in practice, using TILDE, a first-order regression algorithm, to learn an alignment and demonstrate its functioning in an example scenario. We find empirically that: (1) the difficulty of learning an alignment depends on the relative complexity of different trust models; (2) our method outperforms other methods for trust alignment; and (3) our alignment method deals well with deception.

The second approach to communicating about trust is to allow agents to reason about their trust model and personalise communications to better suit the other agent's needs. Contemporary models do not allow for enough introspection into — or adaptation of — the trust model, so we present AdapTrust, a method for incorporating a computational trust model into the cognitive architecture of the agent. In AdapTrust, the agent's beliefs and goals influence the priorities between factors that are important to the trust calculation. These, in turn, define the values for parameters of the trust model, and the agent can effect changes in its computational trust model, by reasoning about its beliefs and goals. This way it can proactively change its model to produce trust evaluations that are

better suited to its current needs. We give a declarative formalisation of this system by integrating it into a multi-context system representation of a beliefs-desires-intentions (BDI) agent architecture. We show that three contemporary trust models can be incorporated into an agent's reasoning system using our framework.

Subsequently, we use AdapTrust in an argumentation framework that allows agents to create a justification for their trust evaluations. Agents justify their evaluations in terms of priorities between factors, which in turn are justified by their beliefs and goals. These justifications can be communicated to other agents in a formal dialogue, and by arguing and reasoning about other agents' priorities, goals and beliefs, the agent may adapt its trust model to provide a personalised trust recommendation for another agent. We test this system empirically and see that it performs better than the current state-of-the-art system for arguing about trust evaluations.

Acknowledgements

I have been told that, despite the effort that goes into writing a doctoral thesis, nobody ever reads one. I would therefore like to start by thanking you, the reader, for proving that statement wrong! However, it is a fact that you would not be reading this thesis, if not for the help and support of a great many people.

The first and foremost of these are, of course, my Ph.D. supervisors, Jordi Sabater-Mir and Marco Schorlemmer. Despite only having met me in a Skype meeting, they gave me the opportunity to start the adventure that was doing my Ph.D. in Barcelona. Upon my arrival I was greeted with two topics I knew nothing about: trust on the one hand, and channel theory on the other. My task? To use the latter to enable communication about the former. How? Read this thesis and see. Marco and Jordi's enthusiasm for the research we set out to do, their inspiration and advice made this work possible.

Equally important has been the support of my lovely girlfriend, Safae Jabri, who put up with my endless, unintelligible, attempts to explain what I was doing and why it was, or was not working. She has been at my side through the highs and lows of this project, with her comfort when I was frustrated and her enthusiasm when I was excited. For that (and for doing the dishes) I am eternally grateful.

I would also like to thank my parents and brother in Holland. My parents have always encouraged and stimulated my interest in science, and supported my decision to pick up and move to Spain when the opportunity arose. I am thankful to my Dutch friends, who are always up for a beer or a board game when I'm back for a visit.

I do not know how I would have survived my thesis without the opportunity to blow off some steam from time to time. Sailing almost every weekend has been fantastic and I thank Francesc "Kiko" Pares, his son Marc, and everybody else with whom we crewed the Tete in the many regattas we won, or lost, but always had fun in. Aside from great crew mates they are also great drinking buddies and the late nights with gin-tonics, caipirinhas and laughs are equally as memorable as the sailing race before. Of course, for winding down after a day of work, my friends and colleagues at the IIIA have been amazing. My thanks to Tomáš Treščák, Marc Pujol, Norman Salazar, Toni Penya, Juan-Antonio Rodríguez, Jesús Cerquides, Pablo Almajano, Pere Pardo, Marco Cerami, José-Luis Fernández, Mertitxell Vinyals, Isaac Pinyol, Mari-Carmen Delgado, Angela Fabregues

and Jesús Giráldez for being great colleagues and friends, and whether at the Vila or elsewhere, always good for a party. A special shout-out to Marc Esteva, who left us too early. May he rest in peace.

My 3-month stay at the KU Leuven in Belgium was a great success thanks to the hospitality of all the people there. Obviously a huge thanks to Hendrik Blockeel for accepting my stay, supervising my work there, and helping me get started. Thanks to Jan Ramon, Leander Schietgat and Kurt de Grave for sharing their office with me, Daan Fierens for his help and troubleshooting with TILDE and Celine Vens for assisting remotely with the same. Finally thanks to Bogdan Moldovan, the many bars in Leuven and the hundreds of Oranje supporters for making my stay fun, as well as productive!

This thesis would not have been possible without the Spanish government's generous funding through the Agreement Technologies project (CONSOLIDER CSD2007-0022, INGENIO 2010) and I am grateful to Carles Sierra for his leadership of this project, giving me the opportunity to write this thesis and present parts of it at various conferences and workshops. I am also grateful for the funding provided by the Generalitat de Catalunya grant 2009-SGR-1434, the European Union's COST Action on Agreement Technologies (ICO801) and the Spanish project CBIT (TIN2010-16306). My visit to Belgium was funded by the Spanish government's grant for mobility (EDU/2933/2009).

Finally, I would like to thank everybody who I have forgotten to thank. If I have forgotten to acknowledge you, I am in remiss. I want you to know that I am terribly sorry and will be sure to rectify this oversight in any future thesis I write.

–Andrew Koster

Part I

Introduction

Chapter 1

Introduction

Trussst in me, jussst in me
Shhhut your eyes and trussst in me
You can sssleep, sssafe and sssound
Knowing I am around

–Kaa, in Disney's The Jungle Book

The philosopher and statesman Francis Bacon said that "the greatest trust between man and man is the trust of giving counsel" [1625]. When we rely on someone's advice, we trust them to know not only what is best for them, but also what is best for us. We expect them to counsel us, taking into account our wishes and goals. This is equally true for counsel about trust. We trust an adviser to recommend people who are trustworthy for accomplishing our purposes.

Trust is rapidly being recognised as an essential mechanism for computational agents in social multi-agent systems [Luck et al., 2005], but computational trust is still a young field of research. The main focus has so far been to discover, and improve upon, mechanisms for the modelling of trust, whereas the communication of that trust has received less attention; however, as more, and better, computational trust models have been designed, an important observation must be made: trust for computational agents is equally subjective as trust is for humans. Agents, human or computational, must trust other agents in a certain environment, and for a specific purpose. It is, therefore, important that, even if an agent is a trusted adviser, his advice cannot simply be taken at face value. We must, additionally, have an assurance that the advice is applicable to our specific environment and purpose.

In this thesis we describe two different ways of providing a computational agent with this assurance. The first is Trust Alignment, which gives the receiver of a trust evaluation a way of translating from the recommender's frame of reference into the receiver's own. The second is Trust Adaptation, which allows the receiver and recommender to argue about both the environment, and purpose for which the trust evaluation is necessary. This allows the recommender to personalise its trust evaluation to the seeker's frame of reference.

1.1 Motivation

Trust is, without doubt, an indispensable attribute of a healthy society. Recent history is an important reminder of this fact: while the financial crisis of the late-2000s was the culmination of various different factors, many economists blame it, at least partly, on a lack of trust among the major financial entities involved [Guiso et al., 2008]. This is an impactful example of how trust, or the lack of it, can influence our lives, but we do not need to look at such global examples alone to see the necessity for trust in our society. We trust that cars will stop for a pedestrian crossing, that the plumber will fix our kitchen sink (and will not steal our television) and that, if my girlfriend comes home late, I will have done the cooking. Trust permeates our everyday life, and there is strong indication that trust is a fundamental concept, not just in human society, but in many animal societies as well [Bekoff and Pierce, 2009].

In this discussion we have so far left out one very important aspect, and that is *why* we trust, or do not trust, someone. This question will return in many different forms throughout this thesis, but in this case we are simply interested in its most direct interpretation: we trust someone based on some amount of information that we have about his (or her) behaviour. In the simplest case this is information obtained from *direct experiences*, or past interactions with that person; however, in our modern society, social groups are far too large for direct experiences alone to provide enough information about all the people we must choose to trust, or not. In all of the animal kingdom, it seems to be a unique characteristic of humans, that we form complex relationships with so many people, and there is considerable evidence that this is due to our use of language [Dunbar, 1993] and the ability to spread *reputation* [Fehr and Fischbacher, 2003; Conte and Paolucci, 2002]. Especially interesting is the simulation by Conte and Paolucci. This simulation shows that a society's overall utility is improved far more if the agents have the ability to spread reputation, than if they merely have the ability to evaluate trust based on direct experiences. It thus demonstrates that the ability to communicate about trust is an essential part of forming effective trust evaluations.

Why we trust, however, is not just based on information about someone's behaviour. Another influence is the beliefs, goals and disposition that we have, or, simply put, our frame of reference. A central assumption in this thesis is that *everybody trusts differently*, with which we mean that, given the exact same information about someone's behaviour, it is entirely possible that one person decides to trust that person, and another person decides not to. Trust is thus inherently *subjective*, which greatly complicates the communication we discussed above. In communication between humans we can easily solve this. We have no problems asking *why* someone is trustworthy, and subsequently, interpreting this information within our own frame of reference. The question we set out to answer in this thesis is:

> *How can computational agents communicate their subjective trust evaluations in a meaningful manner?*

An obvious prior to the central question of this thesis is, obviously, why we should care about computational agents trusting in the first place, and secondary to that, why we should care about them communicating their evaluations. First off, we need to state that we take the concept of a computational agent in the sense that it is an *intelligent* agent. It is a piece of software "whose simplest consistent description requires the intentional stance" [Wooldridge and Jennings, 1995]. The intentional stance is a term in folk psychology, popularised and properly defined by Dennett [1987, page 17]:

> Here is how it works: first you decide to treat the object whose behaviour is to be predicted as a rational agent; then you figure out what beliefs that agent ought to have, given its place in the world and its purpose. Then you figure out what desires it ought to have, on the same considerations, and finally you predict that this rational agent will act to further its goals in the light of its beliefs. A little practical reasoning from the chosen set of beliefs and desires will in most instances yield a decision about what the agent ought to do; that is what you predict the agent will do.

In other words, an intelligent agent is a piece of software, whose simplest description requires it to be described in terms of a rational entity that reasons based upon its personal beliefs and desires. This, in contrast to other pieces of software, whose simplest description is given in terms of states and reactions, an algorithmic description of its function or any other way to describe the program without endowing it with intentions.

We are specifically interested in those computational agents that form part of a multi-agent system. Agents in such a system must interact with one another; however, trust is still unnecessary if all agents are programmed with the same beliefs and desires, as is often the case in swarm computing [Fernandez-Marquez, 2011]. Things start to get interesting if there are multiple agents with conflicting goals and the agents are required to cooperate, coordinate and negotiate with each other. Multi-agent systems with these characteristics are being designed for application in, among other domains, e-commerce, autonomous vehicles, semantic web services, sensor networks and the smart electricity grid. In these environments, the agents are required to make choices about *whom* to cooperate, coordinate or negotiate with, and as in human (and animal) societies, trust is an indispensable tool for aiding in this choice [Luck et al., 2005].

Communication about trust is important for a similar reason. When it is necessary for an agent to interact with agents that it has had no prior experience with, communication about trust allows it to gather valuable information, allowing it to choose whom to interact with. Furthermore, because we are talking about intelligent agents, each with its own beliefs and desires, these trust evaluations may be equally subjective as they are to humans, and therefore agents must also be able to communicate *why* they evaluate another as trustworthy. Little work has been done in allowing agents to, on the one hand, communicate such descriptions, and on the other hand, interpret them. With the methods we

provide, we aim to extend both the theoretical and experimental work in this interesting and important field of research.

1.2 Main Contributions

This thesis contributes to the field of computational trust for multi-agent systems by answering the question posed in the previous section in two different ways, and uncovering interesting concepts along the way.

1.2.1 Trust Alignment

The first approach to answering how agents can communicate subjective trust evaluations is given through what we call Trust Alignment. As the name implies, this approach is heavily influenced by approaches to semantic alignment and allows us to crisply define the problem of communicating subjective trust evaluations, as well as a solution. We show that our computational solution, FORTAM, improves upon the state of the art. The contributions of Trust Alignment are thus:

- We give a mathematical model of the problem of aligning sets of subjective trust evaluations. Specifically, we assume computational trust models are algorithms for calculating a trust evaluation, based upon a set of interactions. The subjectivity of trust can then be identified by understanding that, given a set of interactions that are shared between two different agents, the trust evaluations each agent's algorithm calculates are different. By modelling this in Channel Theory [Barwise and Seligman, 1997], a mathematical framework of information, we gain a deeper understanding of the problem, its relation to other alignment problems, and most importantly, possible solutions.

- By placing some restrictions on the language used for describing interactions, we can move from the mathematical model of the problem to a theoretical description of a solution, using machine learning techniques to allow an agent to automatically learn an alignment between another agent's subjective trust evaluations, and its own. By using the description of the interactions, this alignment is sensitive to the context in which a trust evaluation is made.

- We analyse the requirements for communication in this framework and identify the *relevance* and *consistency* as necessary, if not sufficient, properties that the sender must adhere to for the receiver to be able to learn an alignment. We show that, if the sender's translation, between its internal description of the shared interactions and the shared language for describing these, forms a Galois connection, then this translation is guaranteed to maintain relevance and consistency.

- We introduce the First-Order Regression Trust Alignment Method, or FORTAM. This method uses TILDE [Blockeel and De Raedt, 1998], a machine-learning algorithm, to perform Trust Alignment. We analyse its properties empirically with regards to the number of shared interactions, and especially, the complexity of the trust models. We introduce a novel method for measuring the difference between two computational trust models and postulate that this corresponds to the difficulty of learning an alignment between the two agents' evaluations. FORTAM, in addition to providing an alignment of subjective trust evaluations, is also capable of dealing with a significant amount of deception on the part of the sender.

1.2.2 Trust Adaptation

The second approach we take in this thesis is to consider the problem from the perspective of argumentation, rather than alignment. Whereas an alignment is the process of finding correspondences, and differences, between concepts, an argumentation dialogue allows agents to attempt to reach a consensus through logical reasoning. We use argumentation in Trust Adaptation to discover where two agents' trust models differ, and more importantly, why they differ. Such argumentation may result in one, or both agents, adapting its trust model, possibly temporarily, so that the sender can better personalise its trust evaluation to the beliefs and desires of the receiver. We identified the lack of integration between computational trust models and intelligent agent models, and consequently rectify this with the introduction of AdapTrust, a cognitive framework for integrating a computational trust model into an intelligent agent. The main characteristics of AdapTrust are:

- AdapTrust introduces a mechanism for reasoning about the agent's trust model, thereby adapting this trust model to new goals or changes in the environment. This is done by providing the tools to integrate a computational trust model into the beliefs-desires-intentions (BDI) architecture [Rao and Georgeff, 1991]. Our framework is unique in that it places very few constraints on the computational trust model used, as opposed to most cognitive trust models, which are designed with a specific method for computing trust in mind.

- We use a multi-context system [Giunchiglia and Serafini, 1994] to formalise an extension of the classical BDI logic for intelligent agents. Multi-context systems have the distinct advantage of supporting modular architectures and encapsulation, making them easy to implement. This also holds for AdapTrust, in which we incorporate the logics for reasoning about the trust model in separate contexts. An additional advantage of the formalisation in a multi-context system is that the logic is easily extensible, if required.

- We analyse the applicability of AdapTrust by demonstrating how to incorporate three, very different, trust models into AdapTrust and we show

how reasoning about, and adaptation of, the model adds additional functionality.

We use AdapTrust as the mechanism for adapting the trust model in the argumentation dialogue for Trust Adaptation. The main features of Trust Adaptation are:

- We improve upon Pinyol's argumentation framework [2011], an existing argumentation dialogue for trust, by extending the argumentation language down to the beliefs and desires of an agent. This allows an agent to build a coherent justification for a trust evaluation all the way up from its basic cognitive concepts.

- We provide a dialogue protocol for a structured dialogue, in which a recommendation-seeker can ask a recommendation-supplier exactly why it has sent a specific trust evaluation. Furthermore, we show how the recommendation-seeker can compare the choices the supplier has made in its trust model to the choices in its own trust model and decide on a course of action. Such an action may be to adapt its own model, to ask, or persuade, the supplier to adapt its trust model, or even to enter into a separate persuasion dialogue about the beliefs underlying the trust models.

1.3 Related Publications

The work presented in this thesis has generated the following publications, in reverse chronological order:

- Andrew Koster, Marco Schorlemmer, and Jordi Sabater-Mir. Opening the Black Box of Trust: Reasoning about Trust Models in a BDI Agent. *Journal of Logic and Computation*, In Press, doi:10.1093/logcom/EXS003.

- Andrew Koster, Marco Schorlemmer, and Jordi Sabater-Mir. Engineering Trust Alignment: Theory, Method and Experimentation. *Journal of Human-Computer Studies*(2012), Forthcoming 2012, doi:10.1016/j.ijhcs.2012.02.007.

- Andrew Koster, Jordi Sabater-Mir, and Marco Schorlemmer. Personalizing Communication about Trust. In *Proceedings of the Eleventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS'12)*, Valencia, Spain, Forthcoming 2012. IFAAMAS.

- Andrew Koster, Jordi Sabater-Mir, and Marco Schorlemmer. Trust Alignment: a Sine Qua Non of Open Multi-Agent Systems. In Robert Meersman, Tharam Dillon, and Pilar Herrero, editors, *Proceedings of Cooperative Information Systems (CoopIS 2011)*, volume 7044 of *LNCS*, pages 182–199, Hersonissos, Greece, 2011. Springer.

- Andrew Koster, Jordi Sabater-Mir, and Marco Schorlemmer. Talking about Trust in Heterogeneous Multi-Agent System. In *Proceedings of the Doctoral Consortium of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI'11)*, pages 2820–2821, Barcelona, Spain, 2011. AAAI Press.

- Andrew Koster, Jordi Sabater-Mir, and Marco Schorlemmer. Inductively Generated Trust Alignments Based on Shared Interactions (Extended Abstract). In *Proceedings of the Ninth International Conference on Autonomous Agents and Multiagent Systems (AAMAS'10)*, pages 1571–1572, Toronto, Canada, 2010. IFAAMAS.

- Andrew Koster. Why does trust need aligning? In *Proceedings of the Thirteenth Workshop "Trust in Agent Societies" at AAMAS'10*, pages 125–136, Toronto, Canada, 2010. IFAAMAS.

- Andrew Koster, Jordi Sabater-Mir, and Marco Schorlemmer. Engineering Trust Alignment: a First Approach. In *Proceedings of the Thirteenth Workshop "Trust in Agent Societies" at AAMAS'10*, pages 111–122, Toronto, Canada, 2010. IFAAMAS.

- Andrew Koster, Jordi Sabater-Mir, and Marco Schorlemmer. A Formalization of Trust Alignment. In Sandra Sandri, Miquel Sánchez-Marré, and Ulises Cortes, editors, *AI Research and Development*, volume 202 of *Frontiers in Artificial Intelligence and Applications*, pages 169–178, 2009. IOS Press.

- Andrew Koster, Jordi Sabater-Mir, and Marco Schorlemmer. An Interaction-oriented Model of Trust Alignment. In *Proc. of the 13th Conference of the Spanish Association for Artificial Intelligence (CAEPIA 2009)*, pages 655–664, Sevilla, Spain, 2009.

- Andrew Koster, Jordi Sabater-Mir, and Marco Schorlemmer. Towards an Inductive Algorithm for Learning Trust Alignment. In Tina Balke, Serena Villata, and Daniel Villatoro, editors, *Proceedings of the 11th European Agent Systems Summer School Student Session (SS-EASSS'09)*, volume 47 of *Bayreuth Reports on Information Systems Management*, pages 5–11, Turin, Italy, 2009. Universität Bayreuth.

1.4 Overview and Structure of the Thesis

This thesis is structured into four parts.

Part I: This part contains this introductory chapter and motivates our research. We have summarised our main contributions and provided an overview of the publications that the work has generated.

Part II: The second part of this thesis deals with Trust Alignment. In Chapter 2 we lay the groundwork, giving a complete description of what a computational trust model is, and surveying the state-of-the-art work in communicating about trust. Next, in Chapter 3 we describe Trust Alignment formally, using the mathematical framework of Channel Theory [Barwise and Seligman, 1997] to describe the problem, and inductive logic programming (ILP) [De Raedt, 2008] as a basis for providing a solution. Also in Chapter 3, we discuss the concepts of relevance and consistency and their necessity in order to learn an alignment. We then move on to the practical method, FORTAM, that is an implementation of the method described in the theory. We demonstrate its viability and compare it to other state-of-the-art methods for communicating about trust in Chapter 4.

Part III: The chapters in this part describe the second approach to communicating about trust: Trust Adaptation. In Chapter 5 we discuss the state of the art in cognitive trust models and discuss the budding field of research that combines argumentation and trust. We then present AdapTrust, our own integration of computational trust with a cognitive agent framework, in Chapter 6. AdapTrust uses a multi-context system [Giunchiglia and Serafini, 1994] to specify an extension of the beliefs-desires-intentions (BDI) logic for intelligent agents [Rao and Georgeff, 1991]. We demonstrate AdapTrust's applicability by showing how three contemporary trust models can be incorporated into it and how this allows an agent to proactively adapt its trust model to the changing environment, and the goals it is trying to achieve. Subsequently, in Chapter 7, we use this ability to adapt the trust model in an argumentation framework. This allows agents to justify their trust evaluations and communicate about these justifications. We describe a dialogue protocol for performing this communication and demonstrate empirically that agents that can adapt and provide personalised trust recommendations perform better than agents that do not have this capability.

Part IV: We conclude with a comparison between Trust Alignment and Adaptation. We discuss the strengths and weaknesses of both approaches and discuss the types of situations in which either, or both, can be applied. Finally we present our conclusions and discuss some future lines of research.

Part II

Trust Alignment

Chapter 2

Trust Alignment: State of the Art

You can know the name of a bird in all the languages of the world, but when you're finished, you'll know absolutely nothing whatever about the bird... So let's look at the bird and see what it's doing – that's what counts.

–Richard Feynman

2.1 Introduction

This part of the thesis bears upon what we call Trust Alignment. Trust Alignment deals with the problem of communicating trust by finding correspondences between two agents' conceptualisations of trust. These correspondences can then be used to translate the other's communications into the own frame of reference. We give a formal description of this in Chapter 3; and in Chapter 4 we describe FORTAM, a practical implementation of trust alignment, and we evaluate it empirically. Before diving into the details, however, we review state-of-the-art methods for dealing with the problem of communicating trust. We treat computational trust models in a very abstract manner, because we are more interested in communication than in the computation of trust itself. Consequently, we do not give a complete overview of contemporary computational trust models. Nevertheless, because of the large number of definitions of what trust is, and what a computational trust model should do, it is necessary to describe our own take on the matter and give a brief review of computational trust models. We do this in the next section and in Section 2.3 we discuss the various approaches to communicating trust.

2.2 Trust Models

Trust has many different definitions, and its use is studied in, among others, philosophy, economy, psychology and sociology, both in classical works [Plato, 370BC; Locke, 1689; Hume, 1737] and in more recent works [Gambetta, 1988; Dasgupta, 1988/2000; Bromley, 1993; Celentani et al., 1996]. One thing all these studies agree on is that trust is subjective. Whether a target is trustworthy is a subjective, personal, consideration on the part of the evaluator. In this thesis we rely heavily on the theory of trust as described by Castelfranchi and Falcone [2010], who state that "trust is (conscious and free, deliberated) reliance based on a judgement, on an evaluation of Y's virtues, and some explicit or entailed prediction/expectation". In other words, trust is a deliberated decision to rely on another agent to perform some action, after evaluating the probability of the result being satisfactory. In contrast to this socio-cognitive model, *computational* trust models do not often deal with the decision to trust, but are rather methods for equipping an agent with the tools for performing the evaluation [Pinyol and Sabater-Mir, In Press].

Seen in the light of Castelfranchi and Falcone's definition of trust, it is unsurprising that communication about trust is problematic. Even if we ignore the cognitive processes involved in making a conscious decision based upon an evaluation, Castelfranchi and Falcone say that the evaluation itself is a subjective judgement that takes into account the criteria upon which an agent will base its decision. In Section 2.3 we shall discuss why the communication of such personal, subjective evaluations is problematic, but first it remains to be shown that computational trust models fit the role we have assigned them: methods for computing such subjective and personal judgements. In the continuation of this section we give our own, abstract definition of what a computational trust model is and what its principal components are, and describe a number of state-of-the-art trust models in the light of this definition.

There are already a number of surveys of trust models in the literature [Sabater and Sierra, 2005; Jøsang et al., 2007; Ramchurn et al., 2004; Pinyol and Sabater-Mir, In Press] that give a good overview of computational trust models. These surveys do a good job of describing the differences, both conceptual and computational, among the various models and are helpful for choosing among the models for a specific purpose; however, we are more interested in considering, from an algorithmic perspective, what all these trust models do. This perspective also allows us to explain how and where an agent's subjective viewpoint plays a role in the computation of a trust evaluation.

2.2.1 Computational trust models

When talking about trust in a multi-agent system, the first thing to note is that we are talking about trust in *computational* entities and not humans. As a result, any model of trust we take under consideration should be a computational model. With this we mean that the description of the trust model must define an *algorithm* for obtaining a trust evaluation from a set of inputs. An algorithm is a

computational method for computing the output of a partial function: given an input in the function's domain for which the function is defined, the algorithm computes a unique output in the function's range. In the most abstract sense, a trust model is thus a *Turing computable function*, or, put in another way, a computational trust model is a computational process defined by a Turing machine, and as such, is a method for calculating the output of a *function*.

This function has as input the different pieces of evidence that an agent considers pertinent for evaluating trust, such as direct experiences, communicated evaluations or reputation. As output it produces an evaluation of a target, based on the evidence provided. This is an agent's evaluation of the target's virtues, and based upon it, the evaluator can decide whether or not to delegate some task to the target, in order to achieve some goal. For the moment we disregard the context-sensitive nature of trust. In the general case an agent's evaluation of a target is dependent upon the task the target should perform and the goal an evaluator wants to achieve; furthermore, the beliefs an agent has about the environment can also influence how an agent's trust model computes a trust evaluation. We return to these aspects of trust in Chapter 5, when we discuss cognitive trust models, but for the purpose of Alignment it is sufficient to regard trust as independent of these issues.

The majority of computational trust models can be considered as monolithic "black boxes" by the agent using them: the input is provided and the output is generated, ready for the agent to use, without the agent needing to know anything about how its model actually works. This makes it necessary for the designer to customise the trust model, in order to incorporate an agent's preferences. We view trust as a personal, subjective, evaluation, and for it to be personalised it must take the agent's own criteria for evaluating the trustworthiness of a target into account.

Most computational trust models can be split into two separate parts. The first processes single pieces of evidence. This evidence can be an agent's direct experience, another agent's communicated evaluation, reputation, or any other evidence an agent might consider pertinent for evaluating trust. Each piece of evidence is evaluated in such a way that it can be used as input in the second part of the algorithm, that aggregates the various different types of evidence in order to best predict whether the target will behave in a trustworthy manner in a future interaction. This second part is the focus of most descriptions of trust models in the literature. The different trust models give a different importance to the various sources of evidence, make different assumptions about their representation, and use different theoretical foundations for aggregating the evidence; nevertheless, in essence, they all perform the same function — they try to provide the most accurate evaluation of a target, given the evidence available.

Sabater and Sierra [2005] survey existing trust models and classify them along a number of dimensions. Among other dimensions, they classify trust models according to the sources of information that a model uses as evidence, and they identify the following sources:

- Direct experiences. These are subdivided into interactions, in which the evaluator itself was an active participant, and observations, in which other agents in the environment participated, but which the evaluator observed in some way. Either way, these are normally the most reliable source of information, although it may not always be easy to obtain direct experiences.

- Witness information. These are communicated evaluations from other agents in the system. As Sabater and Sierra point out, such communicated information suffers from the obvious problem that witnesses, possibly deliberately, manipulate or hide pieces of information; however, in this thesis we argue that it also suffers from the inherent subjectivity of trust, and we focus on methods for solving this.

- Sociological information. This is information that can be learnt from analysing the network of social relations that the agents have with one another.

We would like to add that some trust models also take ontological information into account, which provides information on whether two agents are similar to one another, regardless of the network structure. We would also consider reputation as separate from witness information, because the problem of subjectivity plays a far smaller role in using reputation. Reputation is an aggregation of what many different agents say about a target, and the problem of trust evaluations being subjective must, therefore, be dealt with when determining the reputation. It is often defined as "what a group of agents say about a target", or, in other words, what a group of heterogeneous agents have agreed on to such an extent that they all say the same thing, although they might not actually believe what they say. Reputation, therefore, does not suffer from the same problem as direct communication. Instead, it has the problem that it is a very general estimate of trustworthiness, while agents require an evaluation for a specific situation.

The four sources of information that Sabater and Sierra mention, plus ontological information and reputation, are not exclusive, but we identify these as the most important types in the current literature. Using this information, a trust model computes a single output: the expected trustworthiness of a target. Trust models can, very generally, be captured by the schema of Algorithm 1.

The function **calctrust** is the main focus of most of the literature on computational trust models. Additionally, there are works that have a secondary focus on one, or more, of the functions in the first part. For instance, ReGreT [Sabater and Sierra, 2001] and the model proposed by Rehák and Pěchouček [2007] describe different ways of dealing with the ontological aspects of trust. In this thesis we focus on the **process_witness** function. Other models that give a detailed account of this function are, for instance, TRAVOS [Teacy et al., 2006], BLADE [Regan et al., 2006] and Abdul-Rahman and Hailes' model [2000], which we will describe in more detail in Section 2.3. First, we show how the subjectivity of trust is expressed in computational trust models, using the schema in Algorithm 1 as our guide.

Algorithm 1: Trust Model Schema

Input: T, the target to be evaluated
Input: DI, a set of direct interactions
Input: DO, a set of direct observations
Input: WI, a set of communicated evaluations
Input: SI, sociological information
Input: OI, ontological information
Input: Rep, the reputation of target T
$Outcomes :=$ **process_experiences**(DI, DO)
$Recommends :=$ **process_witness**(WI)
$Network\_vals :=$ **process_network**(SI)
$Role\_vals :=$ **process_roles**(OI)

$trust :=$ **calctrust**$(T, Outcomes, Recommends, Network\_vals, Role\_vals, Rep)$

Output: $trust$, the trust evaluation of target T

2.2.2 A brief survey of trust models

Subjectivity of trust evaluations is caused by two different factors. The first is that all agents have different evidence for their trust evaluations. In other words, if we look at Algorithm 1, the sets DI, DO and WI will differ from agent to agent, and in some models, the same holds for SI, OI and Rep; although there are also models in which these are treated as shared information. We do not consider this factor as problematic, in fact, we argue that this is a desired property: agents can provide each other with new information precisely because they have different evidence; however, the second factor that causes subjectivity does create problems when communicating about trust. The second factor is that each agent uses a different way of evaluating the evidence. In other words, the functions **process_experiences**, **process_witness**, **process_network**, **process_roles** and **calctrust** can be different from agent to agent. Let us start with subjectivity in the easiest place to identify it: the processing of an agent's own direct experiences.

Direct experiences

Most trust models ignore how the **process_experiences** function works and simply assume that it provides evaluations of direct experiences, using a representation that the **calctrust** function can use. There are some notable exceptions. Sierra and Debenham [2005] describe an information-based trust model, in which they describe precisely how an agent can evaluate a target based on a single interaction. In their model any interaction is accompanied by a contract (although such a contract can simply be considered as the expected outcome of an interaction). They use a probability distribution over the possible outcomes of an interaction, and use, as evaluation of an interaction, one minus the normalised

negative entropy of this probability distribution. The evaluation of a single direct experience is thus a measure of how trustworthy the target is based on that experience, because, as Sierra and Debenham state, "the more trust the less dispersion of the expected observations and therefore the closer to 1 [the measure gets]".

It is immediately obvious that the outcome of this model depends very heavily on the contract, or expectations, of the evaluating agent. If the agent considers certain aspects of an interaction unimportant, it will not include these in the contract. Two agents who consider different aspects of an interaction important will thus have different contracts, and resultingly, different evaluations of the interaction, even if the interaction partner acts in exactly the same manner.

A very similar model is used by FORTrust [Hübner et al., 2009]. This is a cognitive model of trust, in which the link is made explicit between, on the one hand, an interaction, and on the other hand, the goal that the delegating agent wishes to achieve with this interaction. The outcome of an interaction results in the delegating agent updating its beliefs about the state of the environment, and the evaluation of the interaction is a comparison between the goal state and the actual resulting state. Thus the evaluation of an interaction is very dependent on the specific goal an agent was trying to accomplish by interacting, similar to Sierra and Debenham's contract. ReGreT [Sabater and Sierra, 2001] introduces the term *outcome*, which is used to indicate the evaluation of a pairing of an initial contract and the result of an interaction. The same approach is used in Repage [Sabater et al., 2006]. Similar to FORTrust, the specifics of this evaluation are left to the designer.

In general, the descriptions of trust models in the literature assume that some **process_experiences** function exists that is similar to the ones described above. Yu and Singh state, in the description of their highly influential reputation model, that the outcome of an interaction can be considered in terms of Quality of Service, measured in a real value between 0 and 1 [Yu and Singh, 2002]. This also implies an expected service, even if they do not mention it outright, and some measurable difference between the expected and provided service. This way of dealing with direct experiences seems to be a widely adopted approach. Abdul-Rahman and Hailes [2000] use a discrete representation, in which an agent is evaluated as being "very untrustworthy", "untrustworthy", "trustworthy" or "very trustworthy", based on a single interaction, and while they do not explain how such an evaluation is reached, the description implies a similar method to the one above. The same can be said for BRS [Jøsang and Ismail, 2002], TRAVOS [Teacy et al., 2006] and Vogiatzis et al.'s model [2010], which use an even more rudimentary representation: a binary value to indicate whether the interaction was satisfactory or not. While such a coarse representation of an outcome leaves little room for subjectivity, the process of evaluating will still follow the same principle, and with enough differences between two agents' expectations the problem will still arise. Because evaluations of interactions form the foundations of all trust models, they all result in subjective evaluations.

Note that we are not claiming this is a bad thing. In fact, we think it is an

absolute necessity that agents evaluate others according to their own, personal, criteria. This notwithstanding, if agents try to use witness information, this is something that needs to be taken into account.

Witness information and reputation

Most trust models use direct experiences if they are available, and fall back on the other sources of information if they are not. Using witness information and reputation is the most common method of obtaining information that allows the evaluator to estimate the trustworthiness of the target. Reputation, if available at all, is considered in two different manners. The first is to consider reputation using a centralised environment, where all agents can access a service that provides reputation information, such as BRS [Jøsang and Ismail, 2002]. Considered like this, reputation is a generally available evaluation of any target in the system. This is a practical implementation of the concept of reputation described by Conte and Paolucci [2002], where information about reputation spreads very rapidly throughout the system, and the agents, while having their own evaluation (or in Conte and Paolucci's words, *image*) of a target, share a global reputation value for each target.

The second, and possibly more prominent, use of reputation is as an aggregated evaluation of witness information: each agent calculates its own "reputation" of a target, based on the communicated evaluations it receives. In this way, reputation is more an implementation of the **process_witness** function than a use of reputation, as it is understood in sociological sources. The computational model that best distinguishes between the two different concepts of reputation and witness information is Repage [Sabater et al., 2006], in which reputation is clearly defined as the aggregation of what agents *say* about a target, whereas a "communicated image" is what another agent claims it actually *believes*. The difference between the two is that in the latter case the agent is taking responsibility for the evaluation, while in the former it is communicating it, possibly in the hope that it is useful to the receiver. In BDI+Repage [Pinyol et al., 2012] this difference is modelled with a separate modality S for "say" in the logic for an agent's belief base, allowing Pinyol et al. to formalise the difference between a communicated reputation and communicated image.

We maintain this distinction and consider the second use of the word reputation as a method for incorporating witness information, rather than a proper use of reputation: the trust models clearly assume that the communicating agent takes responsibility for the communicated evaluation. Moreover, some models, such as LIAR [Vercouter and Muller, 2010], rely on this in order to "punish" the communicator in case this information is erroneous. Regardless of whether the computational trust model has mechanisms for detecting misleading agents, it is generally acknowledged that there may be such deceptive agents in the environment. Consequently witness information is considered to be less reliable than direct experiences. In addition to deception, the subjectivity of trust is problematic: even information from a completely honest witness is not as reliable as direct observations, simply because the witness has different preferences and

priorities for evaluating trust. This subjective bias is necessarily reflected in the witness' communication. As a result, there are a number of ways for processing witness information in order to deal with deception and subjectivity, which we discuss in Section 2.3 and we consider how our work, as described in this thesis, improves on the state of the art in this area. Even though these methods may also differ from trust model to trust model, and thus cause witness information to have different effects in different models, we do not really consider the processing of witness information as a source of subjectivity. Many of the models for processing witness information do so in order to deal with the subjectivity of trust evaluations, but because there are different ways of processing witness information, once again taking preferences of the agent into account, these methods themselves add subjectivity. In other words, the fact that trust is subjective in the first place leads to different ways for incorporating witness information being used, which in its turn adds further subjectivity to agents' trust evaluations. Despite this, we do not think this problem is very serious: the subjectivity added by evaluating direct experiences differently, and using different aggregation methods, is far greater.

Network and role information

Information about the underlying social network can be another valuable source of information, although it is dependent on the environment whether such information is available and usable. With this type of information we mean, explicitly, that information about the social network is used as a separate source of information in the trust model. Another, different, use of the social network is for aggregating witness information. Yu and Singh [2002] take this approach by simply filtering out any witnesses who are more than a certain, configurable, number of hops away. Advogato [Levien and Aiken, 1998] and TidalTrust [Golbeck, 2006] use the social network in a similar, but more sophisticated manner: the structure and paths to the source are taken into account in order to process witness information. They do not take agents' location in the network, number of friends, or other such dimensions into account as a separate source of information, unlike ReGreT [Sabater and Sierra, 2001]. ReGreT explicitly models what Sabater and Sierra call "neighbourhood reputation" and this is calculated by using a rule-based system that evaluates targets based on their position in the network. For instance, if the evaluator trusts an agent A, and agent B has a cooperative relationship with A, a rule in the neighbourhood reputation model can give a positive evaluation to B, despite the agent having no direct interactions or witness information about B. This can, once again, lead to different evaluations between agents. Assuming the social network is fixed and shared, there is still no reason to assume different agents have similar rules for this neighbourhood reputation. If they don't, their **process_network** functions will result in different outputs given the same inputs.

Ontological information is used in a variety of different ways in trust models. ReGreT, for instance, uses the ontology to define roles that agents can perform, and if nothing at all is known about an agent, then ReGreT evaluates that agent

based on the role it performs. Osman et al. [2010] and Schläfli [2011] use the ontology in a slightly different manner: by defining a distance metric over an ontology, they propagate evaluations among different nodes (or roles) in the ontology. While Osman et al. and Schläfli use this propagated trust in very different ways, the basic idea is the same. From a communication perspective Osman et al.'s model is quite straightforward: they define it as a reputation model, and its inherent function is to aggregate diverse opinions from many different agents. Schläfli assumes the ontology is shared among agents and they can use this ontology to avoid privacy issues in discussing agents' trustworthiness in sensitive contexts. This seems like a promising approach, although they still must deal with the inherent subjectivity from evaluating direct experiences differently.

Another way of using ontological information is to define roles dynamically. Specifically, Rehák and Pěchouček [2007] use an approach that deals with the context-dependency of trust by clustering similar interactions together based on an ontology for describing interactions. This, however, creates a separate problem for the communication. The trust model uses a clustering algorithm to dynamically assign different roles, but due to agents having different experiences, they will cluster their experiences differently, and thus have different reference contexts. This creates a problem when communicating, because two agents may cluster a single experience into different contexts. Insofar as we know, nobody deals with this problem: context-dependent models that use witness information, refer to a shared role taxonomy, such as in Schläfli's work [2011]. The second problem is that even if agents agree on the role, they may use different criteria to evaluate trust. It might be argued that if this is the case, the clusters are too large, and they conceptually encompass multiple roles. We, in contrast, argue that, due to agents using their own personal criteria for evaluating direct experiences, attempting to fit evaluations into a fixed role taxonomy is counterproductive. Instead we need to find ways for translating trust from one agent's subjective frame of reference into another.

Aggregation methods

Finally, after having processed the different types of input a trust model might use, the function **calctrust** in Algorithm 1 computes a trust evaluation of a target, using the available input. While most models describe this algorithm as being virtually independent of the evaluator agent (except for the inputs), this is not necessarily the case. Many trust models use parameters in this computation process, and deciding on the optimal parameters depends partially on the environment and partially on individual preferences. We return to this in more detail in Chapter 6, but now we just wish to mention this additional source of subjectivity in trust models. Examples of such parameters are factors for discounting older evidence in many models, including BRS [Jøsang and Ismail, 2002] and FIRE [Huynh et al., 2006], biases for (or against) the different types of input, as in ReGreT [Sabater and Sierra, 2001] and Castelfranchi and Falcone's socio-cognitive model of trust [2010] and thresholds for discarding information, such as ignoring witness information if the distance to the witness in a social

network is above the threshold, as described in Yu and Singh's model [2002].

These parameters of the aggregation model, combined with the subjectivity introduced in the processing of the various inputs, can cause individual agents' trust evaluations to be very different, even if the actual inputs are the same and the agents use the same aggregation method. If agents, therefore, attempt to use witness information without processing it correctly, they will, in general, not obtain useful information, and it may even be counterproductive. In the next section we discuss the methods that are used for processing communicated information from witnesses.

2.3 Processing Witness Information

In the previous section, we explained that computational trust models receive objective evidence about the world as input. Each agent's trust model processes and aggregates this input differently, resulting in a subjective evaluation of a target, for a specific task. Moreover we showed how, even if two agents use what seems to be the same trust model, their inherent preferences can still cause similar evidence to result in different trust evaluations. We now focus on the communication of these subjective evaluations, and specifically on the methods proposed for processing witness information.

Witness information is an important source of evidence for most trust models. While direct experiences are almost always a more reliable source, they are also harder to obtain, and if they are unsatisfactory they generally imply a cost to the evaluator. In most environments, communication with other agents has a very low cost, and because these agents have had their own direct experiences in the environment, they can prove to be cheap sources of evidence about a large number of targets. Because of this, almost all computational trust models use witness information to some extent. Nevertheless, the reliability of this evidence is problematic; both because of the subjectivity of trust evaluations, and because witnesses might have reason to provide false information. There are a variety of approaches to deal with these problems and we discuss them below. We divide these approaches into two different categories: those whose focus is mainly on detecting deceptive agents (and incidentally deal with subjectivity of trust), and those that try to deal with subjectivity directly (and incidentally deal with deceptive agents).

2.3.1 Dealing with deception

Problems with deceptive agents have played a prominent role in the literature. Bromley [1993], in one of the first studies to explore the sociological aspects of reputation, devotes a chapter to "impression management", with which he means the ways an individual can change how others perceive him. Moreover, he deals with other people attempting to influence how an agent is perceived. Specifically, in the chapter on corporate reputation [Bromley, 1993, page 162], he says:

> "The performance of a person or firm is not the only source of information that contributes to reputation. The behaviour of other people or other firms also provides information. Deliberately spreading false information to enhance or diminish a reputation is a direct intervention."

Since Bromley's book, most of the literature on trust and reputation recognises the role that deception plays in manipulating agents' evaluations. Additionally, collusion and deception are large problems in the most prominent computational reputation system to date, eBay [Dellarocas, 2000; Hoffman et al., 2009]. It is, therefore, unsurprising that many methods have been developed to detect and deal with individuals and groups providing deliberately deceptive trust evaluations. Most approaches deal with this in the computational trust or reputation model itself, and we discuss the most common approaches below. Another approach is to cleverly design the environment so that deception is not profitable. If, through mechanism design, the incentives to deceive are removed, or a larger incentive is provided to tell the truth, then agents are more likely to provide truthful information [Witkowski, 2010; Jurca and Faltings, 2007]. This type of mechanism requires a centralised approach. Witkowski, for instance, focuses his incentive mechanism on online auction houses, such as eBay, where a central authority can provide the incentives. Note that this only aims to deal with deceptive agents, and not with the problem of communicating subjective trust evaluations. The agent-centric approaches do allow an agent to deal with subjectivity to a certain extent. The advantage of a mechanism-based approach, however, is that it can be used in combination with any of the agent-centric approaches.

The agent-centric approaches that we focus on attempt to distinguish between lying and truthful witnesses, and use this distinction to filter out information from the liars. Sometimes they also actively spread their findings, warning other agents of potential liars. There are a number of mechanisms for distinguishing between liars and honest agents, and we discuss the ones most relevant to our work here.

Using the trust model

The very first computational trust model described by Marsh [1994] did not mention deception explicitly, but did provide a very simple mechanism for processing witness information. In this model, the value of a trust evaluation is considered as a quantification of the trust that the evaluator agent has in the target. Marsh states that, to an agent A, the trustworthiness of a target T, based on witness W's communication is: $Trust_A(T) = Trust_A(W) \times Trust_W(T)$, or the product of A's trust in the witness W and the witness' trust in the target T. This has a number of very strange properties. Intuitively, if the trustworthiness in the witness is low, the reliability of its communication should decrease, not the trust in the target. This is especially obvious if trust evaluations from various witnesses are combined. An additional oddity is that if the two values for

$Trust_A(W)$ and $Trust_W(T)$ are negative (which is possible in Marsh's model), $Trust_A(T)$ becomes positive. This can be explained in the sense that "the enemy of my enemy is my friend", but we feel it is strange to treat trust in such a black and white manner — especially as, even for enmity, the validity of the maxim is, at best, situational. Despite these computational discrepancies, the idea of using the trustworthiness of the witness to process its communications is a very intuitive one.

Vercouter and Muller [2010] follow the same intuition, and LIAR uses the trust model to process recommendations. In LIAR, trust is considered in a multi-faceted manner. As such, trust in an agent as, for example, a seller is completely separate from the trust in that same agent as a recommender. The latter role is the one used to evaluate witness information. Witness information is aggregated using a weighted average, in which the trustworthiness of each witness as a recommender is used as the weight. This means that recommendations from untrustworthy recommenders are given less importance than recommendations from trustworthy ones; however, this approach still suffers from some irregularities.

The first is that, while they use the trustworthiness of each witness as a weight in the aggregation, this is not reflected in the final result: this is a single numerical evaluation of the target. The information that this numerical evaluation is based on the information provided by, for example, a few untrustworthy witnesses is lost and thus cannot be taken into account in the remaining computation of a trust evaluation.

More important to us is the fact that Vercouter and Muller do not make it clear *how* the trustworthiness of an agent in its role as a recommender is evaluated. They state that LIAR is used recursively to decide whether a recommendation is trustworthy or not, but the model proposed for evaluating direct experiences is not immediately applicable to trust recommendations. Whereas a direct experience has immediate feedback because the difference between the expected outcome and the actual outcome of an interaction can be evaluated, a recommendation is not as easily evaluated. At best, an agent can evaluate an agent's recommendation of a target, if it chooses to interact with it. In this case it measures the difference between the expected outcome, based on the recommendation, and the actual outcome. If, however, the agent decides not to interact with the target of the recommendation, then it is entirely unclear how to evaluate a witness' recommendation using LIAR.

If LIAR were to have an effective method for assessing the trustworthiness of recommenders, then this method would, in addition to filtering out information from liars, also filter out recommendations from witnesses whose subjective criteria for evaluating trust are substantially different from the agent's own criteria.

Statistical filtering

Schillo et al. [1999] provide the first computational method for evaluating witness information in a statistical manner. They assume witnesses do not lie outright because the chance of getting caught is too high. Instead, they deceive the

receiver by withholding information with a certain, unknown, probability. The receiver thus only receives the information that was not withheld. By considering this as a Bernoulli process in which only one of the two outcomes can be observed they can use statistical methods to estimate how much information was left out. They then use this estimate to obtain an approximation of what the witness' recommendation would have been if it had not withheld information. In other words, they attempt to reconstruct the witness' actual evaluation, given some assumptions about a witness' behaviour. While the assumptions are somewhat unrealistic, the approach is of special interest to us because it attempts to find a way for *using* a witness' information under the assumption that the witness is deliberately deceptive. Our work focuses on how to use witness' information, despite differences in the subjective frames of reference of the evaluator and witness. In this sense, Schillo et al.'s work is historically relevant to us, although their approach is completely different to ours and is unable to deal with the subjectivity of trust evaluations. Later examples of statistical approaches to detecting liars do make more realistic assumptions than Schillo et al. do, but they do not attempt to use deceiving witnesses' evaluations. Instead they focus on filtering out the influence of deceptive agents on the computation of a target's trustworthiness.

TRAVOS [Teacy et al., 2006] provides one such a method for filtering out deceiving witnesses. In contrast to LIAR, they specify exactly how the accuracy of a witness' recommendation is estimated. Teacy et al. take the Bayesian view on statistics, and they model trust as the probability a target will fulfil its contract, given the evidence available. If the agent has low confidence in its evaluation based on direct experiences (for instance, it has interacted too little with the target), then TRAVOS uses witness information and uses the same Bayesian principles for processing this. The accuracy of a witness' recommendation is the probability that it is accurate, given a probabilistic estimate of that witness' accuracy in the past. Because trust evaluations are probabilities, they are easily comparable and the past accuracy is given by comparing past recommendations from the witness to the agent's own evaluations of the target. If these are similar the accuracy is considered high and otherwise it is low. Based on this comparison, Teacy et al. give a formula for calculating the accuracy of a new recommendation. This accuracy is used to modify the importance in the aggregation of different witnesses' opinions. Because they do not take the average, but calculate this in terms of conditional probabilities, this formula is quite complex, but has the desired effect of filtering out inaccurate witness' opinions. A similar approach is taken by Vogiatzis et al. [2010], who take into account that agent behaviours can change over time.

POYRAZ [Şensoy et al., 2009] also filters out witness information, but Şensoy et al. realise that agents may not deceive consistently, but rather the decision whether or not to deceive is context-dependent. They therefore take the context into account when comparing the accuracy of witness' recommendations to the actual evaluation.

These methods can all filter out information from witnesses who are lying,

and just as LIAR, also filter out recommendations from agents with very different subjective criteria.

Other approaches

There are a multitude of other methods for attempting to deal with deception. Fullam and Barber [2007] use Q-learning to learn when to use direct experiences and when to use witness information. They do not distinguish among witnesses, but treat all witnesses the same. Obviously if witnesses are generally unreliable the algorithm will learn not to use witness information. This approach does not distinguish between deceiving agents and agents whose subjective criteria are different, but it should learn to disregard witness information if agents of either kind are prevalent.

Another approach is to use the underlying social network. As mentioned in Section 2.2.2, Yu and Singh [2002] simply consider the path length to a witness as a measure of reliability. This does not explicitly take deception into account, but assumes any agent too far away is unreliable. ReGreT [Sabater and Sierra, 2001], in addition to using the social network for assessing "neighbourhood reputation", uses the network to assess the credibility of a witness. The same rule-based approach they use for neighbourhood reputation applies, but this time the witness is evaluated, rather than the target. ReGreT uses this not as the primary method for evaluating the credibility of a witness but rather as part of a hybrid system. It primarily uses the trust model recursively to evaluate the trustworthiness of the witness, in a manner similar to LIAR. If this cannot be reliably calculated it uses the "social credibility" based on the social network. These methods allow ReGreT to decide whether or not to believe the witness' communication, or filter it out.

BDI+Repage [Pinyol et al., 2012] also uses a hybrid approach, but combines the trustworthiness of a witness with Q-learning. The Q-learning is used in a similar manner to the way that Fullam and Barber use it, namely to learn the optimal mix of using direct experiences and witness information. However, they also use the trustworthiness of witnesses to decide whether to just outright discard information from certain witnesses. Any communication is thus preprocessed into a "valued communication" which attaches a weight to it, before it is aggregated.

We do not suppose this is a complete overview of the methods for dealing with deceptive agents, but neither do we mean it to be. We aimed to distinguish the most prominent approaches and show that deception is dealt with in a variety of manners; however, none of the approaches mentioned can distinguish between witnesses that intentionally deceive another agent and those that simply use different criteria for evaluating trust. All of the approaches mentioned use some measure of the reliability of a witness in order to decide whether or not to discard that witness' information. The only method that attempts to translate deceptive information is Schillo et al.'s model [1999]; however, it makes some rather strong assumptions about the environment in order to function. Distinguishing between intentionally deceiving agents and innocent agents with different methods for

evaluating trust can prevent a number of problems, both for the agent and for the wellbeing of the society.

Disadvantages of filtering approaches

The first and foremost problem with filtering out any witness information based on some criteria for detecting deceptive agents is that the methods described also filter out information from agents with different subjective criteria. In an open multi-agent system, such a method will filter out recommendations from so many witnesses that the agent searching for a recommendation will be hard-pressed to find any recommendations that *do* pass its criteria. The experiments in Sections 4.3 and 7.5 corroborate this, although these are simulated environments and may not model real situations. A bit of introspection also seems to indicate that we, as humans, deal with the two situations in very different manners. We do indeed attempt to detect, and avoid, witnesses who are likely to deceive us, but if a friend's recommendation does not coincide with our own experience we are far more likely to attempt to find out why we disagree and learn at least something from his recommendation than just dismiss it as an attempt to deceive. Thus it seems that finding some way of using witness' subjective evaluations is preferable to filtering them all out. Deception and subjectivity in witness information are two separate problems, that both require treatment in a computational trust model.

The problem of dealing with both together is exacerbated if consequences are attached to the detection of a deceptive agent, whether through the mechanism, as in [Jurca and Faltings, 2007], or by individual agents, as in LIAR [Vercouter and Muller, 2010]. If a witness runs the risk of being punished for giving recommendations that are different from the receiving agent's expectations, merely because they use different criteria for evaluating trust, the mechanism runs the risk of failing. The mechanism of sanctioning deception goes beyond its purpose, and in the worst case causes agents to no longer offer recommendations for fear of being punished. Even in the best case it is likely that some agents will be unjustly marked as a liar, with all the consequences that entails. Thus if agents want to sanction deceptive witnesses, it is essential that they distinguish between genuine deception and other causes of different, subjective, trust evaluations.

2.3.2 Dealing with subjectivity directly

Having explained why it is necessary to deal with deception and subjectivity separately, it is only logical that we now discuss the state of the art in methods for dealing with subjectivity in trust. Staab and Engel [2008] give a theoretical treatment of the problem of integrating subjective witness information into the trust computation. They consider this as a layered problem: an agent has objective observations of the environment, which it evaluates as subjective experiences. These are aggregated into trust evaluations. This is quite similar to our abstract treatment of a trust model in Algorithm 1 (page 17). The main difference is that Staab and Engel consider the initial evidence that serves as

input for the trust model as being objective, while we do not make this assumption. What they describe as "experiences" corresponds with what we call outcomes, the output of **process_experiences**, and they do not discuss other types of evidence, although their model is abstract enough to consider all intermediate evaluations (*Outcomes*, *Recommends*, *Network_vals* and *Role_vals* in Algorithm 1) as experiences and not just those based on direct interactions. The output of the trust model in Staab and Engel's abstraction is the same as in ours: an aggregation of subjective evaluations of evidence.

The main principle of Staab and Engel's work is that at each subsequent level it becomes harder to exchange information. They state that, by communicating at the lowest level (that of objective descriptions of the environment), the problem of subjectivity is removed; the receiving agent can use the objective information to simulate the situation as if it were its own experience. In our abstract view of a trust model, this corresponds to the sender communicating the sets DI and DO (although Staab and Engel do not discuss the other sources of information, the agent could communicate the other inputs, WI, SI, OI and Rep, as well), rather than communicating the output of the trust model. The receiving agent then uses its own trust model with the received input and computes its own subjective trust evaluation, using its own algorithm, thereby obviating the need for communication of the higher level elements.

This approach is used by Şensoy and Yolum [2007], but has some important drawbacks. The first is that it restricts agents' autonomy: agents must "store" all information that might possibly be related to any other agent's trust evaluation. For instance if, in an e-commerce environment, an agent finds the colour of an item irrelevant, then it can function perfectly well if it does not remember the colour of items it has bought; however, if we expect other agents to be able to recreate their own evaluations from such low level information, it must be available. The agent must be able to communicate the colour of items to other agents. Colour of the item for sale is merely one of many properties of interactions that might be considered evidence for the trustworthiness of a target. Any of these properties, however trivial it may seem to the agent in question, might be relevant to other agents, and it must thus be stored — just in case some agent somewhere requires that information. Our approach, discussed in Chapter 3, does not require this: agents can choose what properties of an interaction they want to remember and communicate about. This ties into the second disadvantage of communicating the evidence directly, which is that agents are limited to the use of a shared ontology — Şensoy and Yolum [2007] make the implicit assumption that all relevant aspects of an interaction for all possible agents are communicable in a shared ontology. It is not clear that this is always the case; real interactions might include details that are not included in the ontology. On top of this, there may be privacy issues because agents may be willing to tell each other that an interaction was successful, or not, without wanting to discuss the exact details of that interaction; especially, if these details contain sensitive information (such as in financial transactions).

Our approach does not make these assumptions. While we do require a shared

language in which to communicate about the interactions, agents may choose themselves what properties of an interaction to communicate. This means that their internal representation of such an interaction can be much richer than the one in the common ontology. In our approach we learn an alignment that is based on a set of shared interactions: an agent can learn relationships using a limited amount of communication about these interactions by additionally taking the information at the second level, that of the subjective evaluations of these interactions, into account. How this works exactly is described in Chapter 3.

A straightforward approach to tackling subjectivity of trust might be to propose to communicate the trust model itself; however, this approach has a number of practical problems. The first is that it requires agents to be introspective regarding their trust model: they must not only be able to use it, but know how it works, in order to tell other agents how it works. In practice very few trust models are designed with this property in mind, and most are simply treated as black boxes: information from a variety of sources is given as input and the calculated trust evaluations are returned as output. In Chapter 6 we propose a model that allows more introspection, and in fact allows agents a limited amount of communication about the trust model itself, which we discuss in Chapter 7. Regardless, if agents can explain how their trust model works and they share a language in which to express this explanation, there remains a problem with the interpretation. It is unclear how the receiving agent can compare its own trust model to the received trust model. Automatically finding similarities and dissimilarities between two complex programs is itself a complex problem.

Dealing with subjectivity, at least in practice, thus requires something more than the straightforward approaches offer. A number of different methods have been proposed, using a range of different technologies. We discuss all of them separately.

Ontologies for trust and reputation

One natural approach is to consider the problem of communicating trust evaluations as a problem of ontology alignment. As such, it seems logical to define a comprehensive ontology for trust. LRep [Pinyol and Sabater-Mir, 2007] and FORe [Casare and Sichman, 2005] are two prominent examples of such ontologies; however, in practice these ontologies are unsupported by most of the different computational trust models in development. Furthermore, even if these ontologies were generally accepted, the subjective nature of trust would still not be captured by them. The problem is not one of defining what concept is captured by the terminology, but how an evaluation comes to be, and how it is used. Euzenat and Shvaiko [2007] call this type of heterogeneity in use "pragmatic heterogeneity". In their book they describe three levels of heterogeneity. The first, the level of syntactic heterogeneity, is quite straightforward: two agents use a different syntax to describe the same domain. An example in the real world would be a Chinese and an English speaking agent trying to communicate. At the semantic level the problem of syntax is presumed solved, but two agents use a different semantic for the same words. For instance, two agents who are

discussing a minivan. One categorises a minivan as a small bus, while the other categorises it as a large car, so the meaning they assign to the word minivan is slightly different. This is the level at which most research into ontology alignment falls. Problems of syntactic and semantic heterogeneity can be solved by having a well defined and shared ontology, such as the ones defined by Pinyol and Sabater-Mir, or Casare and Sichman.

There are, however, problems of heterogeneity that fall into neither of these two categories, but must be described at the third level of heterogeneity, that of pragmatics. At this level two agents agree on the syntax and the conceptual meaning of the word, but there is heterogeneity in how the word is used; this is almost always the problem when two agents try to talk about subjective concepts, such as "taste", but also trust. Problems of pragmatic heterogeneity have received little attention, although the problem has long been recognised in the field of semiotics [Chandler, 2002]. Semiotics studies the meaning of signs, and Morris [1938] proposed that the pragmatic aspect is an integral part of the meaning of signs. He specifically distinguishes pragmatics from syntax and semantics as follows:

> "Since most, if not all, signs have as their interpreters living organisms, it is a sufficiently accurate characterisation of pragmatics to say that it deals with the biotic aspects of semiosis, that is, with all the psychological, biological, and sociological phenomena which occur in the functioning of signs. ... The unique elements within pragmatics would be found in those terms which, while not strictly semiotical, cannot be defined in syntactics or semantics; in the clarification of the pragmatical aspect of various semiotical terms; and in the statement of what psychologically, biologically and sociologically is involved in the occurrence of signs." [Morris, 1938, pages 108, 111]

Trust, as we understand it, is *defined* as a psychological and sociological phenomenon, and thus any communication about trust must take the pragmatic aspect into account. Any miscommunication about trust is almost invariably due to pragmatic heterogeneity, regardless of whether this is due to psychological underpinnings in humans, or because we represent such human preferences in a computational trust model. A more modern description of pragmatics in semiotics is given by Bianchi [2004, page 1]:

> "Pragmatics, however, studies how speakers use context and shared information to convey information that is supplementary to the semantic content of what they say, and how hearers make inferences on the basis of this information."

Accordingly, any attempt to bridge the gap of pragmatic heterogeneity should attempt to take the "context and shared information" between the agents into account; however, in the study of alignment, the context has played a marginal role. Atencia and Schorlemmer [2012] perform semantic alignment in a manner that also deals with pragmatic heterogeneity in multi-agent communication.

They learn the conceptual meaning of messages within the context of the way they are used. Insofar as we know this is the only approach that deals with pragmatic heterogeneity. In the next chapter we discuss the problem of Trust Alignment in a theoretical manner, building upon the same concepts that Atencia and Schorlemmer build upon for aligning ontologies.

We would be remiss if we did not also mention the work done by Nardin et al. [2008]. They recognise the problem of heterogeneity of trust, but treat it as a problem of semantic heterogeneity. They thus propose an ontology mapping service from an agent's own ontology to a shared ontology, for which they use FORe [Casare and Sichman, 2005]. By establishing such a mapping for each trust model, when communicating about trust, an agent can use the service to translate its own evaluation into the FORe ontology, and the receiving agent can translate it from FORe into its own personal framework. Nevertheless, the model is problematic for a number of reasons. The first is that the mapping cannot be established automatically: Nardin et al. established the mapping manually for the Repage [Sabater et al., 2006] and LIAR [Vercouter and Muller, 2010] trust models. They showed that it works in a simple environment, but to establish a mapping for each model manually is tedious. The second problem with this ontology mapping is that it still treats trust alignment as a problem of semantic heterogeneity, and as a consequence the agents' subjective criteria for evaluating trust are disregarded. This subjectivity cannot be mapped into FORe and thus a lot of the meaning of an agent's trust evaluation is lost in translation.

We resolve the latter problem by treating the agent's subjective evaluations in the context of the evidence that supports them, thus offering a solution to the problem of the pragmatic heterogeity of trust.

Learning a translation

Another approach to dealing with the subjectivity of trust is by attempting to translate the subjective witness information to the own model. The first to take this approach are, to our knowledge, Abdul-Rahman and Hailes [2000]. Their work describes a trust model that evaluates a target with an integer between 1 and 4, where 1 stands for *very untrustworthy* and 4 for *very trustworthy*. The alignment process uses the recommendations from another agent about known targets to calculate four separate biases: one for each possible trust value. First the alignment method calculates the own trust evaluations of the corresponding target for each incoming recommendation. The *semantic distance* between the own and other's trust is simply the numerical difference between the values of the trust evaluations. The semantic distances are then grouped by the value of the corresponding received trust value, resulting in four separate groups. Finally, the bias for each group is calculated by taking the mode of the semantic distances in the group. This results in four integers between -3 and 3, which can be used when the agent receives recommendations about unknown targets. An agent can simply subtract the corresponding bias from the incoming trust evaluation to translate the message. While this is a very simple approach it seems to work surprisingly well and we will discuss its functioning in more detail in Section 4.3.

Insofar as we know the only other method that learns a translation between the witness information and the own trust model is BLADE [Regan et al., 2006]. This work builds upon a statistical trust model that, similar to TRAVOS [Teacy et al., 2006], uses a Bayesian approach to modelling trust; however, unlike TRAVOS, BLADE attempts to translate subjective (and, incidentally, deceptive) witness information by modelling trust in a Bayesian network. The priors of a trust evaluation are a number of variables that represent aspects of the environment, such as, in the example they give, the delivery time of items in an e-commerce interaction. BLADE then uses a Bayesian inference learner to learn the probability that the evaluator will consider the target trustworthy, given a prior set of interactions in which the target was labelled as trustworthy, or untrustworthy. The same Bayesian inference learner is employed to learn a translation of a witness' communicated evaluation. It receives a similar set of labelled interactions from the witness and simply uses what the witness says as the priors for its learner, rather than its own knowledge of the interactions.

This method shows many similarities to the approach we take; however, it suffers from a number of limitations. Firstly Regan et al. limit their experimentation to an environment with interactions that have a single variable property. While Bayesian networks are known to scale fairly well, it is unclear how tractable the approach is in situations where interactions have multiple properties. A more serious issue is that their choice of Bayesian inference learner limits their description of interactions to a propositional language. Such a language cannot adequately represent domains involving multiple entities and the relationships among them [De Raedt, 2008]. In Section 4.3 we discuss the differences between BLADE and our approach in more detail. One major difference is in the theoretical foundations of the work. BLADE is, first and foremost, a statistical learning method for computing trust evaluations with a specific focus on dealing with witness information. Our work gives a theoretical framework for dealing with subjective witness information, which can also be used to describe BLADE; based on this framework, we give a practical implementation that uses a similar learning method, although we improve upon BLADE in practice as well.

Finally, Serrano et al. [Forthcoming 2012] present a method for the sender of a trust evaluation to perform the learning. Rather than the receiver attempting to translate received messages, the sender learns in what situations the receiver would evaluate an interaction as having been successful. When communicating new trust evaluations, the sender uses this learnt model, rather than its own model. This means that the recommendation-seeking agent, rather than the witness, must communicate about their prior interactions. This aleviates some of the privacy issues because it is the agent who is requesting help (in the form of an evaluation of a target) who has to disclose information about previous interactions to the witness. The witness, however, has a different obligation: it must learn the translation from its own evaluations to the recommendation-seeker's. This can be computationally intensive. More problematic is if the witness is deceptive. Serrano et al. must therefore assume the witness is benevolent. In this, the approach is more similar to Trust Adaptation (see Part III) than Alignment.

Nevertheless, its basic approach is to learn a translation, using a propositional tree learner, so the method it uses is similar to the one we describe in this part of the thesis. As the method uses a propositional learner, it has similar limitations to BLADE.

Arguing about trust

The final approach for dealing with subjective witness information is to use argumentation. We discuss this in detail in Chapter 5, but for the sake of completeness, we give a brief description of Pinyol and Sabater-Mir's method [2010] for deciding whether or not to accept a witness' communication.

A downside of most of the methods described so far, including our own approach, is that statistical, or learning, methods require a large number of prior interactions to obtain an adequate sample. TRAVOS, for instance, needs approximately 50 interactions between the evaluator and the witness in order to estimate the witness' trustworthiness, and Vogiatzis et al. [2010] runs simulations with each witness providing 100 evaluations in order to learn whether the witness is a liar or not. Learning a translation that takes the context into account, as in BLADE or our approach, requires more shared interactions still, although how many depends on the complexity of the problem.

Pinyol and Sabater-Mir [2010] take a completely different approach and allow agents to enter a dialogue in order to argue about the witness' reasons for having the evaluation it communicated. By arguing about the trust evaluation, the receiver can decide whether or not the witness' subjective frame of reference is similar enough to its own, and thus whether or not to accept the communicated evaluation. In this way its aim is similar to the other filtering techniques and therefore suffers from similar drawbacks. With any filtering approach the agent must make a binary decision to either accept, or reject, the witness' communication, while a better approach may be to learn how to use the communication even if it is not an ideal match with the own subjective frame of reference. By filtering out all subjective communications, an agent runs the risk of being too restrictive and filtering out communications from too many witnesses, resulting in a lack of information. Unlike the other filtering techniques, however, Pinyol and Sabater-Mir assume the differences between a witness' recommendation and the own trust evaluation are because of subjective differences between agents' trust models, and the argumentation method they propose tests, for each received piece of information, whether it is acceptable or not. This may mitigate the problem of filtering out too much information at the cost of having to build an argument to support a trust evaluation: this requires some form of introspection in the trust model and is generally unavailable. Even so, argumentation about trust seems like a promising approach and we explore it in great detail in the second part of this thesis.

2.4 Summary

In this chapter we have provided our own definition of a computational trust model, and we have shown how this definition fits in with the existing literature. This abstract definition of computational trust models serves to show how agents' subjective, personal criteria come into play in the computation. We then showed how the subjectivity of trust is problematic in the communication of witness information, one of the primary sources of evidence for trust models. We are not the first to make this observation, and in Section 2.3, we discuss the large variety of methods for processing witness' recommendations, in order to deal with its subjectivity. Despite this variety, none of the methods described deal with the problem in an entirely satisfactory manner. In the following chapters we detail our approach, which differs from previous approaches in a number of ways. We provide a theoretical description of the problem that uses a framework similar to those used in other problems of alignment. This framework provides a theoretically sound basis for dealing with communication of trust evaluations as a problem of pragmatic heterogeneity, and based upon this framework, we describe a solution to the problem of trust alignment. We show how this approach works in practice and compare it to some of the methods described in this chapter.

Our view of trust follows the socio-cognitive theory of Castelfranchi and Falcone and the computational process described by Algorithm 1 does not cover all that the concept of trust, according to this theory, encompasses. However, it does provide a concise explanation of how a trust evaluation is calculated in a subjective manner. In Chapter 6 we propose a method for integrating trust models into a cognitive agent, which deals with other aspects of trust, such as how the output of a trust model is used. The details of such a model are unnecessary for learning a translation of a witness' communication, which is what this part of the thesis discusses. In the next chapter we provide a theoretical framework in which to view the problem of communicating subjective witness information and search for solutions.

Chapter 3

Theoretical Framework

'When I use a word,' Humpty Dumpty said in rather a scornful tone,
'it means just what I choose it to mean – neither more nor less.'
'The question is,' said Alice, 'whether you can *make words mean so many different things.'*

–Lewis Carroll (in Through the Looking Glass)

3.1 Introduction

In the previous chapter we saw how communicating trust is a problem of pragmatic heterogeneity. We discussed the state-of-the-art mechanisms for dealing with this heterogeneity in trust, the most advanced of which is BLADE [Regan et al., 2006]. It uses machine-learning to translate a trust evaluation, using the evidence that the trust evaluations are based on; however, while they base their understanding of trust in a statistical foundation, their treatment of subjectivity in communication is quite ad-hoc. In this chapter we provide a theoretically sound approach to the problem of communicating subjective trust evaluations, which provides the foundations for understanding it as a problem of pragmatic heterogeneity. We propose a novel solution to the problem and show how this ties in to the theoretical model.

We call the process of finding an interpretation of a witness' communicated trust evaluations the process of *Trust Alignment*. Ontology alignment is the process of determining correspondences between ontologies, and in accordance, trust alignment is the process of finding correspondences between agents' conceptualisations of trust. The entire process is summarised in Figure 3.1, and in this chapter we provide a formalisation of the entire process using Channel Theory [Barwise and Seligman, 1997] and Relational Learning [De Raedt, 2008].

In Figure 3.1, agent Alice is the agent performing trust alignment with Bob, who sends the required messages. Ellipses represent objects or sets of objects in the alignment process, and rectangles represent action or computation by the

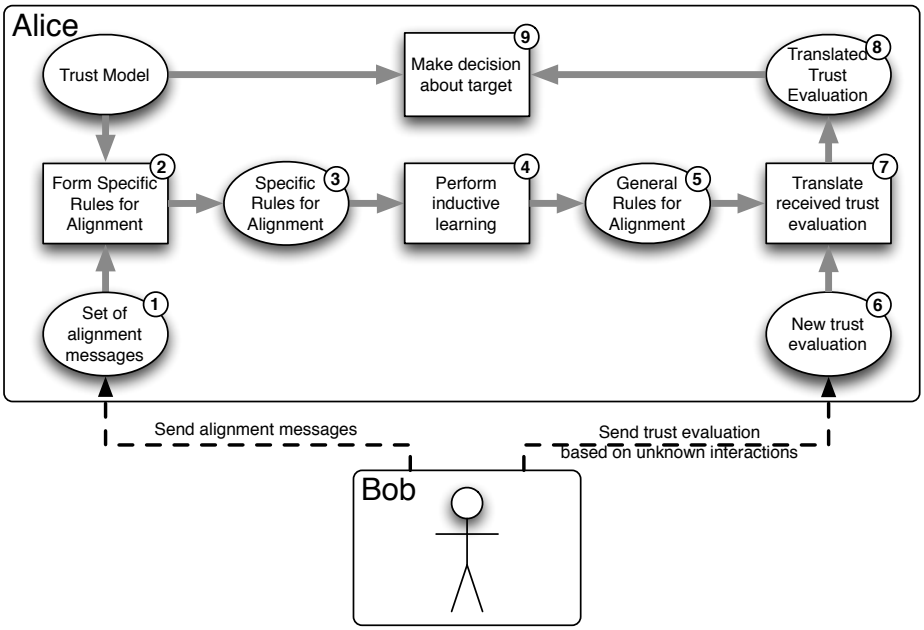


Figure 3.1: Schematic overview of the Trust Alignment process

agent. The alignment process starts with item (1), the reception of a set of alignment messages that are based on a set of shared interactions between Alice and Bob. In step (2) of Figure 3.1 the receiving agent forms a set of Specific Rules for Alignment (SRAs), or item (3). These SRAs represent the relation between the sender's and the receiver's trust evaluations in specific cases. To have predictive value these SRAs need to be generalised, which is done through induction in step (4) of Figure 3.1. This results in the General Rules for Alignment (GRAs), or item (5). These GRAs form the alignment between two agents' trust evaluations and can be seen as the output of the trust alignment mechanism.

We also describe how to apply the alignment. When the same agent, Bob, sends a new trust evaluation based on interactions that are not shared, item (6), Alice can use the alignment to find a translation of this communicated evaluation. This is represented in step (7) of Figure 3.1 and results in a trust evaluation that is translated into the agent's own frame of reference, which is represented in item (8) of the figure. Finally, this translated evaluation can serve as input in the trust model and be used, possibly together with other information about the target, to make a decision regarding the trustworthiness of the target, in step (9).

Section 3.2 describes what we understand as shared interactions. We start in Section 3.2.1 by specifying our abstraction of a trust model. Sections 3.2.2 and 3.2.3 describe the formation of alignment messages and how these relate to the agent's own trust evaluations. In Section 3.3 we discuss how we can learn

an alignment based on the communication. The communication language for discussing interactions is detailed in Section 3.3.1, and we describe the SRAs in Section 3.3.2. In Section 3.3.3 we describe how a set of GRAs are learnt from these SRAs, and at the end of that section, we demonstrate how the alignment can be applied to translate a trust evaluation. To aid in each part of this explanation, we illustrate the main definitions with an example.

Scenario: Alice is writing a book and wants a guest author to write an introduction for it. She is unsure which author would be good at this and asks Bob for his advice. Bob has read some of Zack's articles and was impressed by their clarity and technical quality, so he recommends Zack.

In the above situation it seems like Alice should accept this information; however, it is not that straightforward. Alice only evaluates the first author of any article, thinking that the other authors are not truly involved with writing the article. Bob considers an article as the collective work of all the authors and evaluates all the authors equally. On the other hand, he considers all articles older than two years to be outdated, while Alice takes older articles into account when evaluating an author. If Bob's evaluation was based on articles in which Zack was a second author maybe Alice should reject his opinion. Rather than search for an author herself, Alice instructs her personal computational agent to do this. Alice's agent asks Bob's agent which author it should entrust with this task. Because these agents have not exchanged trust information before, they need to align first.

3.2 Interaction-based Model of Trust Alignment

In Section 2.2 we detailed our perspective on the functionality of computational trust models: they are computational processes that take evidence from a variety of sources as input and perform some computation in order to generate a trust evaluation of a specific target as output. In this chapter we formally describe *Trust Alignment*, or how to communicate these trust evaluations. We are not interested in the actual computation performed and will use a notation that abstracts away from the algorithmic representation entirely. To simplify matters, we also only consider trust evaluations based on direct experiences, which are the most interesting evaluations to communicate about. In other words, we consider a trust model that only takes direct experiences into account, processes these and aggregates them into a trust evaluation of a target. Even so, the processing of direct experiences, as well as the aggregation of these, leads to each agent having its own, subjective trust evaluations based on the same interactions. Agents have different viewpoints and different interests in observing an interaction and thus, even before the trust model adds subjectivity, the agents already differ in their beliefs about an interaction.

In order to start any alignment, agents must be able to communicate. For this purpose, we assume there is a shared language in which agents can communicate about the interactions. This language may not allow agents to express all their observations of an interaction. For instance, if their observations are

internally represented in a language that is richer than the shared language; nevertheless, we assume that, at the very least, the shared language allows agents to communicate some of their observations. What is considered an interaction and what can be expressed about them in the shared language is domain dependent, and we do not want to restrict this. We will also not distinguish between participants in an interaction and the observers of the interaction: these are all considered to be observers, although they all observe different information. In Section 3.3.1 we will go into details about interactions and the shared language for representing them, but for now we simply use a unique identifier as a representation of an interaction. We assume the environment is populated by a set *Agents* of agents and the set $\mathfrak{I}$ is the set of all interactions among these agents in the environment.

Definition 3.1 (Observing interactions)**.** The *observations* of an agent $A \in Agents$ are given by the partial function $observe_A : \mathfrak{I} \to \mathcal{P}(\mathfrak{O}_A)$, which associates an interaction with a set of observations. The set $\mathfrak{O}_A$ is the entire set of observations of agent A and is a set of sentences in $\mathcal{L}_{\mathfrak{O}_A}$, agent A's internal representation language. We use $\mathcal{P}(X)$ as notation for the power set of X.

Because trust evaluations are generally supported by observations of multiple interactions, we define another function, $Observe_A$, in the following manner. Let $I \subseteq \mathfrak{I}$ be a set of interactions, then $Observe_A(I) = \bigcup_{i \in I} observe_A(i)$. We denote the subset of interactions an agent A observes with $\mathfrak{I}_{|A} \subseteq \mathfrak{I}$. The $|$ denotes that this is the set of interactions limited to those observed by agent A. The partial functions $observe_A$ and $Observe_A$ are total on the set $\mathfrak{I}_{|A}$.

Example 3.1. In our example scenario we can consider an article as the representation of an interaction. Strictly speaking, an article is only an outcome of a more complex interaction of several researchers collaborating on a research topic, but in this example it is the only aspect of the interaction that can be observed. There are different observers: the authors perceive the interaction in a very different manner from the readers. For our example we focus on the readers. The act of reading an article can be considered as the act of observing the interaction. The reader observes all sorts of properties of an interaction: whether it is well written, whether it is relevant for his, or her, own line of work, how original the work is, etc. If we assume this type of knowledge is available to a computerised agent, this agent can also be said to have observed the interaction: it has access to the observations.

An agent A's observations *support* some trust evaluation. This is the essence of the trust model. As we stated in Section 2.2 there are many different computational trust models, but all incorporate the concept of direct experiences. We model these experiences as observations of interactions in the environment, which serve as input for the trust model. We say these observations, and by extension the interactions, support the trust evaluation and assume these trust evaluations are statements in a language $\mathcal{L}_{Trust}$. The agents share the syntax of this language, but the way each agent computes its trust evaluations from the different observations can vary and is defined by the agent's trust model. We

do not strictly need the agents to share the syntax of $\mathcal{L}_{Trust}$, but it makes the framework more comprehensible. Eventually each agent's trust evaluations will be considered entirely separately in any case, and the framework would work equally well if agents A and B had two entirely different trust languages. Nevertheless, for the sake of system design, and for understanding the framework, it is useful to assume a single, shared syntax in $\mathcal{L}_{Trust}$.

Example 3.2. In our scenario, we use a simple $\mathcal{L}_{Trust}$, with only one predicate: $trust(target, value)$, with $value \in [-5, 5] \cap \mathbb{Z}$ and $target$ an agent in the system.

We require sentences in $\mathcal{L}_{Trust}$ to be ground atoms of a first-order logic (FOL). This is necessary in Section 3.2.2, where we define the constraints on the alignment using $\mathcal{L}_{Trust}$. We also need sentences in $\mathcal{L}_{Trust}$ to be ground atoms to learn a predictive model, as we will see in Section 3.3. Furthermore, because trust is always about some target agent (all trust predicates have an object), we only consider those predicates which give an evaluation of exactly one such target $T \in Agents$:

$$\mathcal{L}_{Trust}[T] \equiv \{\varphi \in \mathcal{L}_{Trust} \mid \text{the target of } \varphi \text{ is } T\}$$

Thus the communicated trust evaluations must always be about a specific target agent. This split of targets allows us an easy way to partition $\mathcal{L}_{Trust}$:

- $\bigcup_{T \in Agents} \mathcal{L}_{Trust}[T] = \mathcal{L}_{Trust}$
- For all $T, T' \in Agents$ such that $T \neq T'$, we have that: $\mathcal{L}_{Trust}[T] \cap \mathcal{L}_{Trust}[T'] = \emptyset$

We can now give a very abstract model of trust, which we can use as a basis for communication. We ground this framework in a mathematical model of information flow, called Channel Theory [Barwise and Seligman, 1997]. This theory gives a way to model the flow of information within a distributed system and has proven to be useful for modelling a variety of different ontology alignment and semantic integration scenarios [Kalfoglou and Schorlemmer, 2003; Schorlemmer et al., 2007; Schorlemmer and Kalfoglou, 2008; Atencia and Schorlemmer, 2012]. Modelling the problem in this manner allows us to formulate a representation-independent definition and theory of what trust alignment is — against which we can define and check concrete realisations of alignments (see, e.g., Theorem 1 on page 55).

Channel Theory, rather than being a quantitative theory of information flow as with Shannon's classical theory of communication [1948], is a qualitative theory, better suited for semantic or trust alignment scenarios. In the following we will introduce the main concepts of Channel Theory as we proceed in formalising our trust alignment framework; however, we refer to Barwise and Seligman [1997] for a comprehensive understanding of Channel Theory, which is outside the scope of this thesis.

3.2.1 Trust models in Channel Theory

In Channel Theory, systems that hold information are modelled as *classifications*, while the exchange of information among systems is modelled through the use of a *channel*. In our situation we consider trust models as systems that hold information. This information is considered in a very abstract manner: in Channel Theory it is the act of classifying an object, or token, as being of a specific type that causes the system to carry information. Such a classification is defined as follows:

Definition 3.2 (Classification [Barwise and Seligman, 1997, page 69])**.** A *classification* $\mathbf{A} = \langle \mathbf{tok}(\mathbf{A}), \mathbf{typ}(\mathbf{A}), \models_{\mathbf{A}} \rangle$ consists of:

- A set, $\mathbf{tok}(\mathbf{A})$, of objects to be classified, called *tokens*
- A set, $\mathbf{typ}(\mathbf{A})$, of objects used to classify the tokens, called *types*
- A binary relation $\models_{\mathbf{A}} \subseteq \mathbf{tok}(\mathbf{A}) \times \mathbf{typ}(\mathbf{A})$, giving the classification relation.

If $a \models_{\mathbf{A}} \alpha$, then a is said to be of *type* α *in* $\mathbf{A}$.

We define a trust model as a classification to make explicit our intuition that trust in a target is supported by some set of observations of interactions in the environment. The abstract representation, provided by Channel Theory, allows the definition to be independent of the inner workings of the trust model. An important feature of this representation is that, despite abstracting away from the computational trust model that an agent uses, it captures the idea that a trust evaluation is supported by a set of evidence.

Definition 3.3 (Targeted trust model)**.** A trust model $\mathcal{M}_A^{\mathfrak{I}}[T]$ of agent A with regards to target T, given interactions $\mathfrak{I}$, is the classification with:

- $\mathbf{tok}(\mathcal{M}_A^{\mathfrak{I}}) = \mathcal{P}(\mathfrak{O}_A)$
- $\mathbf{typ}(\mathcal{M}_A^{\mathfrak{I}}) = \mathcal{L}_{Trust}[T]$
- $\models_A \subseteq \mathcal{P}(\mathfrak{O}_A) \times \mathcal{L}_{Trust}[T]$, such that if $O \subseteq \mathfrak{O}_A$ (in other words, $O \in \mathcal{P}(\mathfrak{O}_A)$) and $\varphi \in \mathcal{L}_{Trust}[T]$, then $O \models_A \varphi$ represents that, for agent A, the set of observations O *supports* the trust evaluation φ

Example 3.3. In the example we could see this as follows. Let us assume there are two articles, with identifiers $article_1$ and $article_2$, which support Alice's trust in an author Dave. Alice has programmed a trust model into her agent based on her own opinions and thus her agent can compute how to evaluate Dave based on these two articles. Formally:

$$Observe_{Alice}(\{article_1, article_2\}) \models_{Alice} trust(Dave, 4)$$

Where $trust(Dave, 4)$ is a targeted predicate in $\mathcal{L}_{Trust}$. Bob has observed the same articles, but observes different properties and evaluates these differently. His trust evaluation is, formally:

$$Observe_{Bob}(\{article_1, article_2\}) \models_{Bob} trust(Dave, 3)$$

In addition to having a target, trust is often considered to be multifaceted [Castelfranchi and Falcone, 2010; Sabater and Sierra, 2002]: a target is evaluated with regards to a specific role or action that he or she is expected to perform. For instance, an agent may trust another to write an introduction to his book, but not to fix his car. We do not consider this aspect of trust in detail until Chapter 6, but a possible method for dealing with the multifaceted aspect of trust is to consider the target in our framework as not simply a target agent, but rather an agent performing a specific role. For actually calculating the trust, this multifaceted aspect of trust may be more important, but this is not directly a concern when performing trust alignment.

A targeted trust model computes trust evaluations for one specific target, but there are multiple agents in the system. We say an agent's entire trust model is an indexed family of such targeted models $\mathcal{M}_A^{\mathfrak{I}} = \{\mathcal{M}_A^{\mathfrak{I}}[T]\}_{T \in Agents}$.

There is a further remark to make: while Channel Theory does not put any restrictions on the binary relation $\models_A$, we are discussing a specific type of classification; it is a trust model of an agent, and its output should, therefore, make sense for that agent. We assume trust models are deterministic and always give the most accurate trust evaluation that an agent is able to compute. Consequently, any observation O is of at most one type and we can consider the trust model as a partial function between sets of observations and single trust evaluations. This ensures that the Channel Theoretic view of a trust model agrees with our explanation of computational trust models in Section 2.2; additionally, it simplifies the theory we take into consideration in the next section.

3.2.2 The trust channel

Now that we have described trust models algebraically, we can focus on the formal model of how to assess each others' communications about trust. To do so, the agents should form an alignment, and to do this, the agents A and B establish some set of shared interactions $\mathfrak{I}_{|AB} = \mathfrak{I}_{|A} \cap \mathfrak{I}_{|B}$, the set of interactions they have both observed. In this theoretical section we assume that $\mathfrak{I}_{|AB}$ can be established as the intersection of the sets of interactions that each individual agent observed; however, in practice establishing this set may not be so straightforward. Luckily, any subset of $\mathfrak{I}_{|AB}$ may serve the same purpose: a set of shared interactions that both agents have observed and forms the basis of an alignment. Nevertheless, we will see that the larger this set, the more accurate the alignment. Each agent's observations of the shared interactions are a subset of all its observations: $\mathfrak{O}_{A|B} \subseteq \mathfrak{O}_A$ are A's observations of the interactions shared with B (and similarly $\mathfrak{O}_{B|A}$ for agent B's observations). These are observations of interactions that both agents know they have shared.

In the introduction of this chapter, we stated that trust alignment provides a way to align different agents' subjective trust evaluations based on shared interactions. These shared interactions are the set $\mathfrak{I}_{|AB}$, and by considering only the observations of these interactions we restrict what trust evaluations can be supported. In each agent's model there may be trust evaluations that are supported by interactions which are not in this shared set of interactions. From

now on we will no longer refer to an agent's entire trust model, but we will only discuss the trust models $\mathcal{M}_i^{\mathfrak{I}_{|AB}}$ that is limited to the shared set of interactions for agents $i \in \{A, B\}$. We will generally omit the superscript and just write $\mathcal{M}_i$.

Example 3.4. Because both Alice and Bob have observed the interactions $article_1$ and $article_2$, these are in the set of shared interactions of Alice and Bob: $\{article_1, article_2\} \subseteq \mathcal{I}_{|Alice,Bob}$. One thing we should note is that it is not obvious in our scenario why interactions are shared among observers at all. It means that everybody who has read an article knows who else has read an article. This is, however, easily sidestepped by having a prior communication in which the agents exchange information about which articles the agents' owners have read, thus forming the set of shared interactions.

In addition to classifications representing each of the agents' trust models, we follow Channel Theory and represent the system itself with a classification. This allows us to define how the information flows between the different entities. The classification representing the distributed system itself is the *core* of a so called information channel in Channel Theory. In contrast to the classifications that represent each individual trust model, the *core* of a channel should be seen as the connection between the agents' trust models. While an information channel can be defined among any number of entities of a network, we are interested in a binary channel between $\mathcal{M}_A[T]$ and $\mathcal{M}_B[T]$. We call this a trust channel and define its core as follows:

Definition 3.4 (Core of the trust channel)**.** Given two agents A and B, the core of their trust channel with regards to target T is a classification $\mathbf{C}[T]$ given as follows:

- The tokens are subsets of shared interactions:
  $\mathbf{tok}(\mathbf{C}[T]) = \mathcal{P}(\mathfrak{I}_{|AB})$
- The types are trust evaluations represented in $\mathcal{L}_{Trust}$, but distinguishing between those from agent A and agent B by taking the disjoint set union:

$$\begin{aligned}\mathbf{typ}(\mathbf{C}[T]) &= \mathbf{typ}(\mathcal{M}_A[T]) \uplus \mathbf{typ}(\mathcal{M}_B[T]) \\ &= (\{A\} \times \mathcal{L}_{Trust}[T]) \cup (\{B\} \times \mathcal{L}_{Trust}[T])\end{aligned}$$

- The classification relation $\models_{\mathbf{C}[T]}$ such that a subset of interactions I support a trust evaluation of agent A if this trust evaluation is supported in A's trust model by A's observations of I or analogically for B.
  $I \models \langle i, \varphi \rangle$ iff
  - either $i = A$ and $Observe_A(I) \models_A \varphi$
  - or $i = B$ and $Observe_B(I) \models_B \varphi$

We are now ready to define how the information flows between the different entities of the distributed system. For this we need a way to model the relationship between the system as a whole (or the core of the channel) and the

individual entities (trust models). This relationship is given by an *infomorphism*: a contravariant pair of functions that define how information may move between two different systems, or in other words, how two classifications are connected.

Definition 3.5 (Infomorphism [Barwise and Seligman, 1997, page 72])**.** An *infomorphism* $f : \mathbf{A} \rightleftarrows \mathbf{B}$ from classification $\mathbf{A}$ to classification $\mathbf{B}$ is a contravariant pair of functions $f = \langle \hat{f}, \check{f} \rangle$, with $\hat{f} : \mathbf{typ}(\mathbf{A}) \to \mathbf{typ}(\mathbf{B})$ and $\check{f} : \mathbf{tok}(\mathbf{B}) \to \mathbf{tok}(\mathbf{A})$ and satisfying the *fundamental property of infomorphisms*, namely that classification is preserved under mapping of types and tokens:

$$\check{f}(b) \models_{\mathbf{A}} \alpha \Leftrightarrow b \models_{\mathbf{B}} \hat{f}(\alpha)$$

for each token $b \in \mathbf{tok}(\mathbf{B})$ and type $\alpha \in \mathbf{typ}(\mathbf{A})$.

In the trust channel, these infomorphisms are defined as contravariant functions between $\mathcal{M}_i[T]$ and $\mathbf{C}[T]$ for $i \in \{A, B\}$: $\check{f}_i(I) = Observe_i(I)$ and $\hat{f}_i(\varphi) = \langle i, \varphi \rangle$. It is easy to check that these functions satisfy the fundamental property of infomorphisms. With the trust models, the core and the infomorphisms defined, we have the components needed to describe the flow of information within the system. This flow is defined by an information channel. The connections in such an information channel capture Barwise and Seligman's basic insight that the flow of information, expressed at the level of types, occurs because of particular connections at the level of tokens.

Definition 3.6 (An information channel [Barwise and Seligman, 1997, page 76])**.** A *channel* $\mathcal{C}$ is an indexed family [with indices I] $\{f_i : \mathcal{A}_i \rightleftarrows \mathbf{C}\}_{i \in I}$ of infomorphisms with a common codomain $\mathbf{C}$, called the *core* of $\mathcal{C}$. The tokens of $\mathbf{C}$ are called *connections*; a connection c is said to *connect* the tokens $f_i(c)$ for $i \in I$. A channel with index set $\{0, \ldots, n-1\}$ is called an *n-ary* channel.

This definition can be applied to trust alignment: the agents' trust evaluations, despite being subjective, are connected through the interactions which support them. We are thus interested in the binary channel, describing the flow of information between $\mathcal{M}_A[T]$ and $\mathcal{M}_B[T]$:

Definition 3.7 (Trust channel)**.** We define an information channel $\mathcal{C}[T]$ between trust models with target T as follows. Let A and B be agents and $\mathfrak{I}$ a set of interactions, then:

- The two agents' trust models $\mathcal{M}_A[T]$ and $\mathcal{M}_B[T]$ as in Definition 3.3.
- The core of the trust channel $\mathbf{C}[T]$ as in Definition 3.4.
- The trust infomorphisms $f_A : \mathcal{M}_A \leftrightarrows \mathbf{C}[T]$ and $f_B : \mathcal{M}_B \leftrightarrows \mathbf{C}[T]$ as defined above.

We call this information channel the trust channel between agents A and B with regards to T and give a graphical representation of this channel in Figure 3.2. In the centre is the core of the channel: the classification with as tokens

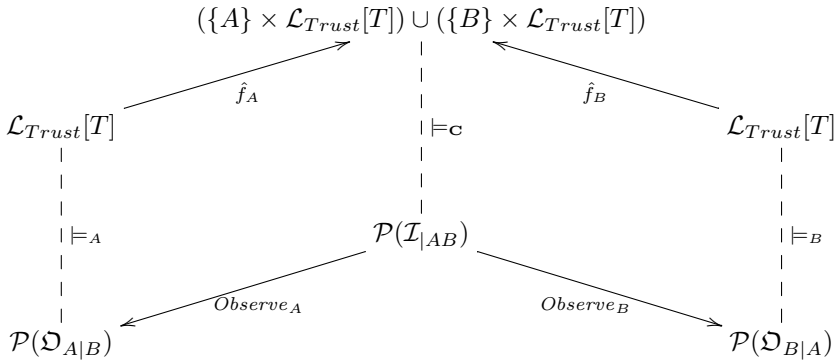


Figure 3.2: A trust channel

the sets of interactions and the disjunct union of both agents' possible trust evaluations as its types. To either side, the individual agents' trust models and the infomorphisms connecting the whole. The channel $\mathcal{C}[T]$ describes the information flow between agents A and B by relating a set of interactions $I \subseteq \mathfrak{I}_{|AB}$ to each agent's trust evaluation of some target T, that is supported by the agent's observations of I. The relationship between A's and B's trust evaluations, at the level of the types of the classifications, is supported by tokens, the sets of shared interactions. This relation, however, is implicit in the trust channel. To make the regularities in the distributed system explicit we need to consider the theory generated by the core classification.

Definition 3.8 (Theory [Barwise and Seligman, 1997, pages 117, 119])**.** Given a set Σ, a *sequent* is a pair $\langle \Gamma, \Delta \rangle$ of subsets of Σ. A binary relation $\vdash$ between subsets of Σ is called a *(Gentzen consequence relation on* Σ. A *theory* is a pair $T = \langle \Sigma, \vdash \rangle$. A sequent $\langle \Gamma, \Delta \rangle$ such that $\Gamma \vdash \Delta$ is called a *constraint* on the theory T. T is *regular*[1] if, for all $\alpha \in \Sigma$ and $\Gamma, \Delta, \Gamma', \Delta', \Pi_0, \Pi_1, \Sigma' \subseteq \Sigma$, it satisfies the following conditions:

Identity: $\alpha \vdash \alpha$

Weakening: if $\Gamma \vdash \Delta$ then $\Gamma, \Gamma' \vdash \Delta, \Delta'$

Global cut: if $\Gamma, \Pi_0 \vdash \Delta, \Pi_1$ for each partition $\langle \Pi_0, \Pi_1 \rangle$ of any subset Σ' of Σ then $\Gamma \vdash \Delta$

Definition 3.9 (Theory generated by a classification [Barwise and Seligman, 1997, page 118])**.** Let $\mathbf{A}$ be a classification. A token $a \in \mathbf{tok}(\mathbf{A})$ *satisfies* a sequent $\langle \Gamma, \Delta \rangle$ of $\mathbf{typ}(\mathbf{A})$ if and only if $(\forall \gamma \in \Gamma : a \models_{\mathbf{A}} \gamma) \rightarrow \exists \delta \in \Delta : a \models_{\mathbf{A}} \delta$. The theory *generated by* $\mathbf{A}$, written $Th(\mathbf{A})$, is the theory $\langle \mathbf{typ}(\mathbf{A}), \vdash_{\mathbf{A}} \rangle$ where $\Gamma \vdash_{\mathbf{A}} \Delta$ if all tokens of $\mathbf{A}$ satisfy $\langle \Gamma, \Delta \rangle$.

[1] The property of regularity is well-known in algebraic logic when dealing with a multiconclusional consequence relation as in our case [see Dunn and Hardegree, 2001]. Notice that any theory generated by a classification (Definition 3.9) is always regular.

A theory in Channel Theory is a generalisation of the notion of consequence in logic: if a classification has models as its tokens and logical sentences as its types, the consequence relation of Definition 3.8 is equivalent to logical consequence. The definition, however, captures a more general notion of consequence, where tokens are not restricted only to models as understood in mathematical logic, such as the ones we describe in this thesis. Such a broader notion makes explicit the regularities of a system of classifications if we consider the theory generated by the *core* of a channel. In our trust channel this is $Th(\mathbf{C}[T])$ and it is this theory that ultimately captures, in the form of sequents, the logical relation between trust evaluations of separate agents.

Theoretically, this would be the theory we are aiming for when aligning trust evaluations, but it may be very large; moreover, in practice, the agents do not have the means to instantly construct the channel and the corresponding theory, but must do so by communicating each constraint of the theory separately. We are therefore interested in finding a smaller set of constraints which serves our overall purpose of alignment. We are looking for those constraints that relate single trust evaluations of two different agents. The theory which satisfies our needs is the subtheory $\mathfrak{X}[T] = \langle \mathbf{typ}(\mathbf{C}[T]), \vdash_{\mathfrak{X}[T]} \rangle$, with $\vdash_{\mathfrak{X}[T]} \subseteq \vdash_{Th(\mathbf{C}[T])}$ such that $\langle i, \gamma \rangle \vdash_{\mathfrak{X}[T]} \langle j, \delta \rangle$ if and only if $i \neq j$ and $\langle i, \gamma \rangle \vdash_{Th(\mathbf{C}[T])} \langle j, \delta \rangle$. We see that this theory contains precisely those constraints we require, because its consequence relation has trust evaluations from different agents on the left and right hand side. Furthermore, each set of interactions supports at most one trust evaluation for each agent, and thus we have single elements — the theory fulfils the purpose of alignment.

Example 3.5. We continue the same scenario and recall Alice and Bob support their own trust evaluations of Dave upon observing interactions $article_1$ and $article_2$. Given just these interactions and the target Dave, we see we have the channel whose core is:

- $\mathbf{tok}(\mathbf{C}) = \{\{\}, \{article_1\}, \{article_2\}, \{article_1, article_2\}\}$
- $\mathbf{typ}(\mathbf{C}) \supset \{\langle Alice, trust(Dave, 4)\rangle, \langle Bob, trust(Dave, 3)\rangle\}$
- $\models_{\mathbf{C}} = \{(\{article_1, article_2\}, \langle Alice, trust(Dave, 4)\rangle),$
  $(\{article_1, article_2\}, \langle Bob, trust(Dave, 3)\rangle)\}$

This generates a theory and we are interested in the subtheory $\mathfrak{X}[Dave]$ of this theory. $\mathfrak{X}[Dave]$ has the following constraints:

- $\langle Bob, trust(Dave, 3)\rangle \vdash_{\mathfrak{X}[Dave]} \langle Alice, trust(Dave, 4)\rangle$
- $\langle Alice, trust(Dave, 4)\rangle \vdash_{\mathfrak{X}[Dave]} \langle Bob, trust(Dave, 3)\rangle$

Both of these are satisfied by the token $\{article_1, article_2\}$.

We have so far considered the channel with regards to a single target. Alignment, however, should happen with regards to all targets in the environment. As with the trust models, the partitioning of $\mathcal{L}_{Trust}$ gives us an in-

dexed family of trust channels $\{\mathcal{C}[T]\}_{T \in Agents}$. We are interested in the theory $\mathfrak{X} = \bigcup_{T \in Agents} \mathfrak{X}[T]$.

3.2.3 Communicating trust constraints

Because the agents do not have access to the channel, they need to communicate their trust evaluations to construct it. This communication is a set of alignment messages. Each alignment message contains an agent's trust evaluation and identifies the set of interactions that supports the evaluation. This is the communication we refer to in Figure 3.1, resulting in the set of received alignment messages in item (1). The actual communication requires a language in which sets of interactions can be specified, and the formal definition of alignment messages is given in Definition 3.11 (page 50), but we can informally define an alignment message from agent B to agent A as a tuple $\langle \beta, I \rangle$ with $\beta \in \mathcal{L}_{Trust}[T]$ and $I \subseteq \mathfrak{I}_{|AB}$, such that $Observe_B(I) \models_B \beta$. In other words, agent B explains the intended meaning of a trust evaluation β, by pinpointing the set of shared interactions I that it observed to support β. Using I, agent A can then use its own trust model to find its own corresponding trust evaluation α, such that $Observe_A(I) \models_A \alpha$. The set I, therefore, satisfies the constraint $\langle B, \beta \rangle \vdash_{\mathfrak{X}[T]} \langle A, \alpha \rangle$: this constraint specifies the semantic relationship of α and β, coordinated through the shared interactions I.

This way of specifying the communication shows large similarities with the work described by Schorlemmer et al. [2007], which also builds on Channel Theory, but with the purpose of defining a framework of meaning coordination for ontology alignment. The intuitions behind both models are very similar as they draw from Channel Theory, although the technical details of the actual channel, the generated theory and the use of the messages is very different.

Repeated exchange of alignment messages, about all targets, allows the agents to approximate the theories $\{\mathfrak{X}[T]\}_{T \in Agents}$ from the family of trust channels between the two agents' trust models. Communication, however, is asymmetric. When agent B sends a message, it allows agent A to find the constraint $\langle B, \beta \rangle \vdash_{\mathfrak{X}[T]} \langle A, \alpha \rangle$, while agent B does not have this information. Communication, therefore, leads to a partition of $\mathfrak{X}$, with $\mathfrak{X}_A$ consisting of the constraints generated through alignment messages from agent B and $\mathfrak{X}_B$ vice versa.

Example 3.6. Dave is not the only author who has written an article. Edward and Frank are other target agents of whom Alice and Bob both have trust evaluations, supported by interactions in $\mathfrak{I}_{|Alice,Bob}$. The resulting theory $\mathfrak{X}_{Alice}$ is:

1. $\langle Bob, trust(Dave, 3) \rangle \vdash_{\mathfrak{X}_{Alice}[Dave]} \langle Alice, trust(Dave, 4) \rangle$
2. $\langle Bob, trust(Edward, -1) \rangle \vdash_{\mathfrak{X}_{Alice}[Edward]} \langle Alice, trust(Edward, -4) \rangle$
3. $\langle Bob, trust(Frank, 5) \rangle \vdash_{\mathfrak{X}_{Alice}[Frank]} \langle Alice, trust(Frank, 5) \rangle$

4. $\langle Bob, trust(Frank, -2)\rangle \vdash_{\mathfrak{X}_{Alice}[Frank]} \langle Alice, trust(Frank, 3)\rangle$

With each of these constraints satisfied by a different token, or subset of $\mathfrak{I}_{|Alice,Bob}$. Note that the last two constraints both concern target Frank, but because different subsets of $\mathfrak{I}_{|Alice,Bob}$ support them, these two constraints are very different. In constraint (3), both Alice and Bob's observations of the interactions lead to a very positive evaluation of Frank. In constraint (4), Bob's evaluation of Frank, based on this different set of interactions is negative, while Alice's is still positive. We reiterate that when either agent actually needs to evaluate Frank for its own use, it will use all the information available to it, but in alignment it can be very useful to generate multiple constraints based on different subsets of the full set of interactions, thus giving a greater understanding of the semantic relationship between the agents' trust evaluations.

We will see that the partition $\mathfrak{X}_A$ allows an agent A to find a predictive model, such that, given a trust evaluation β' from agent B, based on interactions I', that might not be shared, the agent A can find a corresponding α'. Because the interactions may not have been observed by agent A, the alignment needs some way of deriving α' from β', using $\mathfrak{X}_A$. There are various ways of doing this and our work focuses on using a machine learning approach. In Chapter 4 we compare our approach to some other methods. In the next section we detail the theory underlying this approach and show how it resolves the problem of trust alignment.

3.3 Trust Alignment Through θ-subsumption

In general, the agents may have observed different properties of the interactions, and the trust models may use these different observations in different manners. To learn a generalisation which allows an agent to translate the other's trust evaluations based on interactions it has not observed, the agents will have to communicate more than just their trust evaluations. Specifically, they must communicate some information about the interactions which support their trust evaluation. In the previous section we specified alignment messages as a tuple $\langle \varphi, I\rangle$, with φ a trust evaluation and I a set of interactions; however, we did not specify the manner in which these interactions are communicated. We now clarify that this is done using a language for discussing the properties of the agent's environment, which we call $\mathcal{L}_{Domain}$.

The minimal requirement that $\mathcal{L}_{Domain}$ must fulfil is to uniquely identify interactions and thus communicate between agents the precise set I that supports a trust evaluation; however, merely pinpointing this set of interactions leads to two problems. First, an interaction may contain a lot of information, and thus for an agent to obtain a predictive model, it may be very hard to discern what information is *relevant* to the other agent. Secondly, if the interactions are not shared, then merely an identifier of an interaction is meaningless: the receiving agent has not observed the interaction and the identifier is useless to it. Thus we need something more from $\mathcal{L}_{Domain}$: it must not just pinpoint the interactions,

but also allow agents to communicate the properties of the interactions that are relevant to the computation of the trust evaluation. Using $\mathcal{L}_{Domain}$, the agents can communicate what properties of the specified interactions led to the communicated trust evaluation. That information can be seen as a justification for the trust evaluation and can be used to find a predictive model for translating future trust evaluations.

In this section, the examples follow a more Prolog-like syntax than we have used before: any term that starts with a capital letter is a variable, while constants either start with a lower case letter or are quoted. We do this because the logical notation used in this section is reminiscent of Prolog, and we thus follow suit with the specification of variables and constants.

3.3.1 Syntax and semantics of $\mathcal{L}_{Domain}$

$\mathcal{L}_{Domain}$ is a shared language, and most MAS environments have such a language to describe the environment. If there is no shared language to describe the domain readily available, we will assume the agents have used other techniques, such as ontology matching methods [Euzenat and Shvaiko, 2007], to settle on a joint language to describe the environment, allowing them to communicate about the interactions using some such $\mathcal{L}_{Domain}$.

We will use $\mathcal{L}_{Domain}$ in two different ways: the first is as mentioned, for agents to communicate about the interactions; however, it also serves to describe the background information about the environment. For both uses we assume $\mathcal{L}_{Domain}$ is an FOL language. For the definition of background information this is sufficient, and we will discuss its use as such later; however, the communication should provide descriptions of the interactions to be generalised over. In order to learn such a generalisation, we require a specific format, namely Horn clauses. See further below for their construction and use, but for now it is sufficient to know that we require the communication in $\mathcal{L}_{Domain}$ to be restricted in order to achieve this, by taking the existential-conjunctive (EC) fragment of FOL that underlies most assertional languages in Knowledge Representation. We emphasise that we do not wish to restrict all common knowledge among agents in this manner. The background information is also described in $\mathcal{L}_{Domain}$, but we do not restrict the syntax to this fragment, and it may contain other full first-order sentences.

Example 3.7. The agents need a language in which to describe articles, complying with the restrictions above. This language uses the ontology defined in Figure 3.3, using a UML-like notation. It is kept deliberately small and describes only objective properties of the articles. An agent can describe an article with a unique identifier, which, in this example we assume to be the title (in reality this assumption does not hold and we need, for instance, the DOI to uniquely identify articles). For instance, we can identify and refer to $article_1$ using the atomic sentence $title(\text{`}ArticleOne\text{'})$ and describe that article's attributes, such as its authors, with predicates for the relations, such as $author(\text{`}ArticleOne\text{'}, \text{`}Dave\text{'})$. This is the EC fragment of FOL used for descriptions of single interactions. The

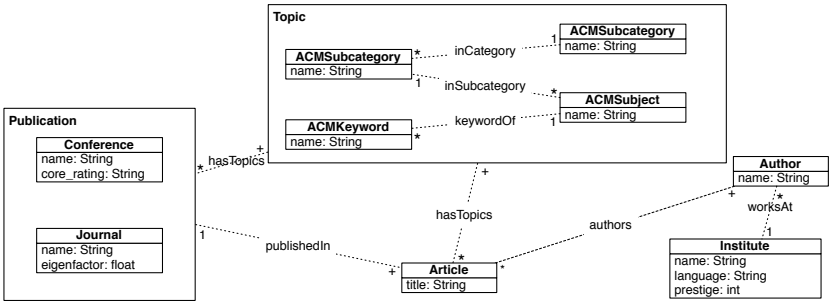


Figure 3.3: A UML-like representation of $\mathcal{L}_{Domain}$

background information uses this same ontology, but is not limited to the EC fragment: it may contain such formulae as $\forall X : title(X) \rightarrow \exists Y : author(X, Y)$, stating that all articles have at least one author. Strictly that sentence says that all items that have a title have at least one author; however, because we use the title as the identifier, we already know that all articles have titles.

$\mathcal{L}_{Domain}$ is a shared language of all agents in the system, and we define the satisfiability of sentences in $\mathcal{L}_{Domain}$ based on Herbrand logic, by using the interactions as Herbrand models. We deliberately left the definition of the interactions vague at the start of Section 3.2, and now consider the interactions as the Herbrand universe for $\mathcal{L}_{Domain}$. Let $U_{\mathcal{L}_{Domain}}$ be the set of all ground atoms in $\mathcal{L}_{Domain}$, also known as its Herbrand base. Each interaction ι should be considered as a set $\{p_1, p_2, \ldots, p_n\}$, where each $p_i \in U_{\mathcal{L}_{Domain}}$. In this manner we can see each interaction ι as a Herbrand structure and can consider all atoms $p_i \in \iota$ as *true* for the interaction ι and all other elements in $U_{\mathcal{L}_{Domain}}$ as being false. This leads to standard Herbrand satisfiability for formulae in which we consider interactions as Herbrand models; however, if a formula ψ describes not a single interaction, but rather identifies the properties of a set of interactions that support a trust evaluation, then such a semantics is not sufficient. For instance, if $I = \{i_1, i_2\}$ and we use the formula $\psi = p_1 \wedge p_2$ to describe I, then we want I to model ψ, but not necessarily i_1 or i_2 to model ψ, because the individual atoms p_1 and p_2 may refer to different interactions. We therefore define satisfiability as follows:

Definition 3.10 (Interaction-based satisfiability of $\mathcal{L}_{Domain}$)**.** Let I be a set of interactions, φ, ψ be formulae in $\mathcal{L}_{Domain}$and p an atom of $\mathcal{L}_{Domain}$, then:

$$
\begin{aligned}
&I \models\!\models p \text{ iff there is a } \iota \in I \text{ such that } p \in \iota \\
&I \models\!\models \neg\psi \text{ iff } I \not\models\!\models \psi \\
&I \models\!\models \varphi \wedge \psi \text{ iff } I \models\!\models \varphi \text{ and } I \models\!\models \psi \\
&I \models\!\models \exists x : \psi(x) \text{ iff } I \models\!\models \psi(a) \text{ for some ground term } a
\end{aligned}
$$

This is equivalent to normal Herbrand semantics after flattening I. Let $\models^{\mathsf{H}}$ stand for standard Herbrand semantics, then $I \models\!\models \psi \Leftrightarrow flatten(I) \models^{\mathsf{H}} \psi$.

Example 3.8. An agent can use $\mathcal{L}_{Domain}$ to describe the articles on which its trust is based. For instance, it could describe the article $article_1$ using the formula $title(\text{`}ArticleOne\text{'})$. We then consider each interaction, or article, as a Herbrand model of such formulae. Each article is a set of all the predicates related to it, e.g. $article_1 = \{title(\text{`}ArticleOne\text{'}),\ author(\text{`}ArticleOne\text{'}, \text{`}Dave\text{'}),\ publication(\text{`}ArticleOne\text{'}, \text{`}ConferenceA\text{'})\}$. As such we have:
$\{article_1\} \models\!\!\models title(\text{`}ArticleOne\text{'})$, because $title(\text{`}ArticleOne\text{'}) \in article_1$.

3.3.2 Specific Rules for Alignment

In Section 3.2.3, we informally defined alignment messages as a tuple $\langle \alpha, I \rangle$, with α a trust evaluation and I a set of interactions. In the previous section we specified $\mathcal{L}_{Domain}$, a language for communicating properties of the interactions. This allows us to give a formal definition of alignment messages.

Definition 3.11 (Alignment Message)**.** An *alignment message* from agent B is a tuple $\langle \beta, \psi \rangle$, with $\beta \in \mathcal{L}_{Trust}[T]$ for some target T and $\psi \in \mathcal{L}_{Domain}$. Additionally, there is a set of interactions $I \subset \mathfrak{I}_{|AB}$ such that:

- $I \models\!\!\models \psi$ and $Observe_B(I) \models_B \beta$
- I is the unique set for which this holds: for all $I' \subset \mathfrak{I}_{|AB}$, if $I' \models\!\!\models \psi$ and $Observe_B(I') \models_B \beta$, then $I = I'$.

An alignment message thus contains a trust evaluation and a description in $\mathcal{L}_{Domain}$ of a unique set of interactions, whose observation supports the communicated trust evaluation. Agents are free to choose *what* they communicate in the $\mathcal{L}_{Domain}$ part of the message, as long as it uniquely identifies the interactions supporting the trust evaluation.

Because $\mathcal{L}_{Domain}$ is an FOL language, any $\psi \in \mathcal{L}_{Domain}$ can be Skolemised by replacing the existentially quantified variables with a new constant. We will call this formula $skolem(\psi)$. We now define a Specific Rule for Alignment (SRA) in a straightforward manner from the alignment message.

Definition 3.12 (Specific Rule for Alignment (SRA))**.** Let $\langle \beta, \psi \rangle$ be an alignment message from agent B to agent A, and let I be the unique set of interactions that supports β and is also a Herbrand model of ψ. Furthermore, let $\alpha \in \mathcal{L}_{Trust}$ such that $Observe_A(I) \models_A \alpha$, and α and β have the same target. We then define an *SRA* as the rule $\alpha \leftarrow \beta, skolem(\psi)$.

The SRA as defined above is nothing more than the constraint $\langle B, \beta \rangle \vdash_{\mathfrak{X}_A} \langle A, \alpha \rangle$, of the theory $\mathfrak{X}_A$, rewritten and annotated with the description ψ in $\mathcal{L}_{Domain}$ of the token that supports this constraint in the core of the channel. Given the restrictions on $\mathcal{L}_{Trust}$ and $\mathcal{L}_{Domain}$, we see that an SRA is a *definite Horn clause*. We write $\mathcal{E}$ for the set of all SRAs constructed by the agent and we see that $\mathcal{E}$ is a representation of a subset of $\mathfrak{X}_A$.

In the last section we have discussed the semantics of $\mathcal{L}_{Domain}$ in terms of Herbrand models; however, the SRAs in $\mathcal{E}$ are a combination of sentences in

$\mathcal{L}_{Domain}$ and $\mathcal{L}_{Trust}$. The semantics of $\mathcal{L}_{Trust}$ is given by the agents' trust models, not by Herbrand semantics. We therefore need to extend the $\models\!\!\models$ relationship between interactions and our languages as follows:

Definition 3.13 (A model for SRAs)**.** Let $\epsilon = \alpha \leftarrow \beta, \psi \in \mathcal{E}$ and $I \subseteq \mathfrak{I}$, then $I \models\!\!\models \epsilon$ iff: if $Observe_B(I) \models_B \beta$ and $I \models\!\!\models \psi$ then $Observe_A(I) \models_A \alpha$.

In other words: if I is a Herbrand model of ϵ, then either the observations of I support A's trust evaluation α, I is not a Herbrand model of ψ, or B's observations of I do not support trust evaluation β.

Example 3.9. We recall from Example 3.6 that we had the constraint of a theory $\mathfrak{X}[Dave]$: $\langle Bob, trust(Dave, 3)\rangle \vdash_{\mathfrak{X}_{Alice}[Dave]} \langle Alice, trust(Dave, 4)\rangle$. Now we know how an agent can obtain the knowledge of such a constraint through communication. For instance, Bob could send a message with evaluation $trust(Dave, 3)$ and the justification in $\mathcal{L}_{Domain}$:
$author(\text{`}ArticleOne\text{'}, Dave) \wedge publication(\text{`}ArticleOne\text{'}, \text{`}ConferenceA\text{'}) \wedge author(\text{`}ArticleTwo\text{'}, \text{`}Dave\text{'}) \wedge topic(\text{`}ArticleTwo\text{'}, \text{`}TopicX\text{'})$

This would allow Alice to construct the SRA:

$$\epsilon_1 = trust(\text{`}Dave\text{'}, 4) \leftarrow trust(\text{`}Dave\text{'}, 3),\ author(\text{`}ArticleOne\text{'}, \text{`}Dave\text{'}),\ publication(\text{`}ArticleOne\text{'}, \text{`}ConferenceA\text{'}),\ author(\text{`}ArticleTwo\text{'}, \text{`}Dave\text{'}),\ topic(\text{`}ArticleTwo\text{'}, \text{`}TopicX\text{'})$$

Because $\{article_1, article_2\}$ is the token that supports both agents' trust evaluations, and it is a Herbrand model for the $\mathcal{L}_{Domain}$ part, we have $\{article_1, article_2\} \models\!\!\models \epsilon_1$.

By constructing SRAs from each alignment message, an agent forms a set of SRAs as step (2) of the process in Figure 3.1 (page 36).

3.3.3 Learning a prediction

Viewing the alignment messages in this way leads to a natural way of considering *predictive* models: the representation of SRAs as Horn clauses, and the fact that we have a set of examples from which we need to induce a more general model, leads naturally to considering this as a problem of inductive learning. Inductive Logic Programming (ILP) is a Machine Learning technique that is concerned with learning in expressive logical or relational representations. We follow De Raedt [2008] and formalise an ILP problem as follows:

Definition 3.14 (An ILP Problem [De Raedt, 2008, page 71])**.** An *ILP problem* consists of:

- A language of examples $\mathcal{L}_e$, whose elements are descriptions of *examples, observations or data*
- A language of hypotheses $\mathcal{L}_h$, whose elements describe *hypotheses* about (or regularities within) the examples, observations or data

- A *covers* relation $\mathbf{c} \subseteq \mathcal{L}_h \times \mathcal{L}_e$ to determine whether a hypothesis *covers* an example.

Most notably this can be applied to problems of classification, in which the examples are divided into labelled classes, and the task is to discover generally applicable rules in the language of hypotheses. The rules should cover as many examples as possible belonging to one class, and as few as possible belonging to any other class. Regression is another form of this, in which the classes are not predefined, but all examples are labelled numerically. The task is then to find generally applicable rules that group the examples together, such that the average internal deviation of each group is minimal. How a rule divides the example set is defined by coverage. By viewing hypotheses as a general set of clauses that entail the examples they cover, we represent trust alignment as a problem of "learning from entailment" and use the corresponding coverage relation:

Definition 3.15 (Coverage for entailment [De Raedt, 2008, page 72])**.** For a hypothesis $H \in \mathcal{L}_H$ and an example $e \in \mathcal{L}_e$, $\mathbf{c}(H, e)$ iff $H \models e$.

In other words, a hypothesis covers an example if and only if it entails it.

To properly consider our problem as an ILP problem we need to define an example language $\mathcal{L}_e$ and a hypothesis language $\mathcal{L}_h$, such that the coverage operator works. We use the set $\mathcal{E}$ of SRAs as $\mathcal{L}_e$, and we see each SRA as an example. For our hypothesis language we use Horn logic with the non-logical parameters from $\mathcal{L}_{Trust}$ and $\mathcal{L}_{Domain}$. A hypothesis $\mathcal{H}$ is a conjunction of Horn clauses in this language. The task at hand is to find a hypothesis $\mathcal{H}$, which covers all of our examples, or more formally, the hypothesis $\mathcal{H}$, such that for all SRAs $\epsilon \in \mathcal{E}$ we have $\mathcal{H} \models e$.

$\mathcal{E}$, however, is not the only information available to the agent for finding $\mathcal{H}$: it can also use background information. $\mathcal{L}_{Domain}$is a language in which to represent facts about the interactions, but some of these facts may be related in a known manner. Such extra information can be considered background information, which can be used when learning an alignment. We assume such background information is available to all agents in the system, or that it can be disseminated in prior rounds of communication.

This background information could, for example, contain taxonomy information, such as that all cars are motor vehicles. If we find a hypothesis which covers all examples about motor vehicles, then with the background information, this hypothesis covers all examples about cars. The definition of coverage using background information is as follows:

Definition 3.16 (Coverage for entailment with background information [De Raedt, 2008, page 92])**.** Given background information $\mathcal{B}$, hypothesis H *covers* an example e iff $H \cup \mathcal{B} \models e$. In other words, if e is entailed by the conjunction of the hypothesis and the background information, we say the hypothesis covers it.

Using this definition of coverage, we define when one hypothesis is more general than another.

Definition 3.17 (Generality [De Raedt, 2008, page 48])**.** Let H_1 and H_2 be hypotheses. We say H_1 is *more general* than H_2, notated $H_1 \succeq H_2$, iff all the examples covered by H_2 are also covered by H_1, that is $\{e \mid \mathbf{c}(H_1, e)\} \supseteq \{e \mid \mathbf{c}(H_2, e)\}$. Conversely, we say that H_2 is *more specific* than H_1 in this situation.

Because we limit our language of hypotheses to conjunctions of Horn clauses, we can simplify the definition of generality and use instead:

Definition 3.18 (Generality of Horn clauses)**.** If the hypotheses H_1 and H_2 are Horn clauses, we can view these as sets of literals. We say $H_1 \succeq H_2$ iff $H_1 \subseteq H_2$. This is equivalent to the above definition.

Our task is to find the most specific hypothesis $\mathcal{H}^*$ that entails all examples in $\mathcal{E}$ given the background information $\mathcal{B} \subseteq \mathcal{L}_{Domain}$. This hypothesis should therefore satisfy the criteria: $\mathcal{H}^* \in \mathcal{L}_H$ such that $\mathcal{H}^* \cup \mathcal{B} \models \mathcal{E}$, and for all $\mathcal{H}' \in \mathcal{L}_H$, if $\mathcal{H}' \cup \mathcal{B} \models \mathcal{E}$, then $\mathcal{H}' \cup \mathcal{B} \models \mathcal{H}^*$. In other words: $\mathcal{H}^*$ covers all examples in $\mathcal{E}$, and if any other hypothesis covers $\mathcal{E}$, then it is more general than $\mathcal{H}^*$.

Another way of defining this is by saying $\mathcal{H}^*$ is the least general generalisation of $\mathcal{E}$. Defined in this way, we can start looking for this $\mathcal{H}^*$ by considering different generalisation operators. As our examples are ground definite Horn clauses, we must use a generalisation operator which can deal with this expressivity. θ-subsumption [Plotkin, 1970] is the most important framework for this, and virtually all ILP algorithms use it. Briefly summarised, we say a clause C θ-subsumes another clause C if there is a substitution θ such that $C\theta \subseteq D$ (i.e., every literal in $C\theta$ also appears in D)[2]. This definition does not take background information into account, and we give the full method below.

The hypothesis $\mathcal{H}^*$ we are searching for, is a formula such that for all $e \in \mathcal{E}$ there is an $H \in \mathcal{H}^*$ such that $H \vdash_{\mathcal{B},\theta} e$, with $\vdash_{\mathcal{B},\theta}$ the consequence relation given by θ-subsumption using background information. This hypothesis is found by iteratively finding the least general generalisation of two clauses, starting from the SRAs themselves. The least general generalisation of clauses, $lgg(c_1, c_2)$, is defined as:

Definition 3.19 (Least general generalisation [De Raedt, 2008, page 134])**.** $lgg(c_1, c_2) = \{lgg(a_1, a_2) \mid a_1 \in c_1 \wedge a_2 \in c_2 \wedge lgg(a_1, a_2) \neq \top\}$, where $lgg(a_1, a_2)$ is the *least general generalisation* of two atoms and $\top$ is the *most general clause*: it is satisfied by all models.

The least general generalisation of two atoms is defined as:

Definition 3.20 (Least general generalisation for atoms [De Raedt, 2008, page 120])**.** The least general generalisation of two atoms a_1, a_2 is a generalisation a of a_1 and a_2 such that for any other generalisation a' of a_1, a_2 there is a non-trivial substitution θ (in other words, it does not only rename variables) such

[2] We use juxtaposition to denote substitution.

that $a'\theta = a$. A generalisation of two atoms can be defined more precisely than in Definition 3.17, namely: if the atoms a_1 and a_2 start with different predicate symbols, their generalisation is $\top$, otherwise it is an atom a, such that there are substitutions θ_1, θ_2 such that $a\theta_1 = a_1$ and $a\theta_2 = a_2$

Because θ-subsumption makes no distinction between $\mathcal{L}_{Trust}$ and $\mathcal{L}_{Domain}$ we need to extend our definition of interactions as Herbrand models: we must define $\models$ for the $\mathcal{L}_{Trust}$ part of the message with substitutions. Let $\boldsymbol{\epsilon} = \boldsymbol{\alpha} \leftarrow \boldsymbol{\beta}, \boldsymbol{\psi}$ be a generalisation through θ-subsumption of n SRAs $\alpha_i \leftarrow \beta_i, \psi_i$ with $i \in [1, n]$ and $I \subseteq \mathfrak{I}$, then $I \models \boldsymbol{\epsilon}$ if and only if the following condition holds: Whenever

- there is a substitution θ_B, such that $Observe_B(I) \models_B \boldsymbol{\beta}\theta_B$
- there is a substitution θ, such that $I \models \boldsymbol{\psi}\theta$

then there is a substitution θ_A, such that $Observe_A(I) \models_A \boldsymbol{\alpha}\theta_A$. This is a straightforward extension of Definition 3.13.

Our definition of $lgg(c_1, c_2)$ does not take background information into account. We therefore need to define $lgg_{\mathcal{B}}(c_1, c_2)$, the least general generalisation relative to background information $\mathcal{B}$. For this we use the bottom clause $\bot(c)$ of a clause c.

Definition 3.21 (Bottom clause [De Raedt, 2008, page 139–141])**.** For a ground Horn clause c and background information $\mathcal{B}$, the *bottom clause* $\bot(c)$ is defined as the most specific ground Horn clause, such that $\mathcal{B} \cup \{\bot(c)\} \models c$, in other words there is no clause $c' \supset \bot(c)$ such that $\mathcal{B} \cup \{c'\} \models c$. This is also called the saturated clause, because we can see this as the process of adding ground literals to c until it is saturated; there are no more literals we can add without invalidating the premise $\mathcal{B} \cup \{\bot(c)\} \models c$.

Example 3.10. The background information, presumed to be known by all agents in the system, is the set of sentences $\{rating(\text{`}ConferenceA\text{'}, \text{`}TierA\text{'})$[3], $rating(\text{`}ConferenceB\text{'}, \text{`}TierA\text{'})$, $subtopic(\text{`}TopicX\text{'}, \text{`}TopicY\text{'})$[4]$\}$. This allows us to construct the bottom clause of the SRA ϵ_1 (from Example 3.9):

$$\begin{aligned}\bot(\epsilon_1) = trust(\text{`}Dave\text{'}, 4) \leftarrow\ & trust(\text{`}Dave\text{'}, 3),\ author(\text{`}ArticleOne\text{'}, \text{`}Dave\text{'}),\\ & publication(\text{`}ArticleOne\text{'}, \text{`}ConferenceA\text{'}),\\ & author(\text{`}ArticleTwo\text{'}, \text{`}Dave\text{'}),\\ & topic(\text{`}ArticleTwo\text{'}, \text{`}TopicX\text{'}),\\ & rating(\text{`}ConferenceA\text{'}, \text{`}TierA\text{'}),\\ & subtopic(\text{`}TopicX\text{'}, \text{`}TopicY\text{'})\end{aligned}$$

Using the bottom clause we define $lgg_{\mathcal{B}}(c_1, c_2) = lgg(\bot(c_1), \bot(c_2))$. All of these operations have workable algorithms to achieve them, described by De Raedt [2008] and Nienhuys-Cheng and de Wolf [1997].

[3] according to, for instance, the CORE ranking system for conferences
[4] where the topics might be from the ACM taxonomy

To summarise: starting from a set of SRAs, communicated through the channel, each agent can compute a generalisation $\mathcal{H}^*$, which covers all the SRAs. $lgg_{\mathcal{B}}(\mathcal{E})$ computes the bottom clauses of all the SRAs and then keeps computing pairwise least general generalisations until there are none left. The resulting conjunction of clauses is our hypothesis $\mathcal{H}^*$. This process is the induction process of step (4) in Figure 3.1 (page 36). Its result, the clauses in $\mathcal{H}^*$, are General Rules for Alignment (GRAs), and the outcome of Trust Alignment is a set of GRAs, depicted in item (5) of Figure 3.1.

Example 3.11. After more communication, Alice constructs a second SRA:

$$\epsilon_2 = trust(\text{`}Greg\text{'}, 4) \leftarrow trust(\text{`}Greg\text{'}, 3),\ author(\text{`}ArticleThree\text{'}, \text{`}Greg\text{'}),\ publication(\text{`}ArticleThree\text{'}, \text{`}ConferenceB\text{'})$$

She can construct $\bot(\epsilon_2)$ and as such can calculate the $lgg_{\mathcal{B}}(\epsilon_1, \epsilon_2)$:

$$lgg(\bot(\epsilon_1), \bot(\epsilon_2)) = trust(X, 4) \leftarrow trust(X, 3),\ author(A, X), publication(A, B),\ rating(B, \text{`}TierA\text{'})$$

This is a GRA. It says that for any agent X that Bob evaluates with value 3, based on an article published at a venue rated $TierA$, Alice should interpret that as an evaluation of agent X with value 4.

As we see in this example, GRAs are predictive rules that allow an agent to translate a trust evaluation, even if the agent has not observed the interactions that the evaluation is based on. The trust model can then use this information, in the way it would normally integrate witness information, to decide whether or not to interact with the target, as in items (7) – (9) of Figure 3.1.

Example 3.12. Using the GRA in Example 3.11, Alice can translate a new message from Bob, which states:

$$\langle trust(\text{`}Zack\text{'}, 3),\ author(\text{`}NewArticle\text{'}, \text{`}Zack\text{'}) \wedge publication(\text{`}NewArticle\text{'}, \text{`}ConferenceZ\text{'})\rangle$$

Alice has the additional background information $rating(\text{`}ConferenceZ\text{'}, \text{`}TierA\text{'})$. With this information she can translate the message from Bob into her own frame of reference, obtaining $trust(\text{`}Zack\text{'}, 4)$; however, we do not assume this is her own trust evaluation. Alice's trust evaluation now uses this translated message, together with any other information she can obtain about Zack, and calculates her actual evaluation of Zack. This she can use to decide whether or not Zack is a trustworthy author for the introduction of her book.

We end this section with a theorem relating the method of generalisation back to our formation of the trust channel of Definition 3.7:

Theorem 1:
For a set of agents $Agents$, aligning agents A and B and the family of trust channels $\{\mathcal{C}[T]\}_{T \in Agents}$, the pairwise union of the SRAs generated by the theories

$\mathfrak{X}[T]$ of these channels forms a complete lattice under θ-subsumption. Furthermore, if we consider the indexed family of partitions $\mathfrak{X}_A[T]$ and $\mathfrak{X}_B[T]$ for all targets T then the union of these also forms a complete lattice under θ-subsumption.

Proof. Let T and T' be target agents, and X and Y be the sets of SRAs generated from $\mathfrak{X}[T]$ and $\mathfrak{X}[T']$, respectively. To show that $X \cup Y$ forms a lattice under θ-subsumption, we need to show that for any two elements $x, y \in X \cup Y$, there is an element that θ-subsumes both x and y. We can show this, by demonstrating that the top element of such a lattice exists: in other words, there is an element $z \in \mathcal{L}_H$, such that z θ-subsumes any element in $X \cup Y$.

Let $z = lgg(lgg(X), lgg(Y))$. We must now show that z exists and that z θ-subsumes all elements in $X \cup Y$. According to Definition 3.12, the sets of SRAs X and Y, are sets of definite clauses. We distinguish two cases:

1. All the clauses in X have the same predicate in the head and similar for Y. In this case the $lgg(X)$ and $lgg(Y)$ are themselves definite clauses. If $lgg(X)$ and $lgg(Y)$ have the same predicate in the head, then $z = lgg(lgg(X), lgg(Y))$ is also a definite clause. If they do not have the same predicate in the head, then $z = \top$, the most general clause. In either case z exists, and $z \succeq lgg(X)$ and $z \succeq lgg(Y)$. Therefore, for any two elements $x, y \in X \cup Y$, we know that $z \succeq x$ and $z \succeq y$.

2. The clauses in X or Y have different predicates in the head. We show this case for X, but the proof for Y is analog. In this case the $lgg(X) = \top$. Thus the $lgg(X)$ is already the top element in the lattice, because there is no clause more general than $\top$. Furthermore, if $lgg(X) = \top$ then $z = \top$ and therefore, for any two elements $x, y \in X \cup Y$, we have $z \succeq x$ and $z \succeq y$.

Because we have made no assumptions about $\mathfrak{X}[T]$ or $\mathfrak{X}[T']$, we can see that this applies to any combination of theories in the family $\mathfrak{X}$ and we have a complete lattice.

The proof of the second part of the theorem, referring to the partition $\mathfrak{X}_A$ and $\mathfrak{X}_B$ is analog.

This proof also shows that if $\mathcal{L}_{Trust}$ contains more than one predicate, it is interesting to first partition the SRAs by predicate in the head and learn these generalisations separately: while the final generalisation will still be $\top$, along the way useful predictive models will be found. □

This theorem has some important implications. Firstly, it means that we do not lose predictive power by considering each targeted channel separately: the generalisation through θ-subsumption of the union of the theories is the same as the generalisation of the union of the generalisations. Secondly it means there is also no predictive power lost when each agent learns its own generalisation separately. They could together find the generalisation of their combined predictive models. Note, however, that this would not be very interesting: it would give only those rules for which both their trust models coincide exactly, whereas the

individual generalisations have more predictive power, giving the *translation* of the other's trust evaluation in a certain situation, described in $\mathcal{L}_{Domain}$.

3.3.4 Computation

The method described in the previous sections is a formal framework. To allow for actual computation based on it, quite some restrictions need to be made. First off, there are privacy issues in constructing the channel. The next section will somewhat alleviate these issues, by giving a way in order to choose what to communicate, although we still assume that the agents must disclose information about the interactions. Additionally, there are restrictions on the computation itself. The computation of the least general generalisation of two clauses is exponential in the number of clauses [Nienhuys-Cheng and de Wolf, 1997]. If that weren't enough, proving a clause θ-subsumes another clause is also proven to be NP-complete [Garey and Johnson, 1979]. Finally any generalisation operator that actually computes the lgg of two clauses may run into the problem that there may be infinite reductions [De Raedt, 2008].

These problems are solved by placing large restrictions on the space in which a computational algorithm searches for a hypothesis. This is done in two ways: the first is by imposing a language bias on the practical implementation of the learning algorithm. This limits which types of hypotheses can be considered. The second is by ensuring that the communicated information is truly *relevant* in the learning process. The sending agent must therefore choose the description of the interactions in $\mathcal{L}_{Domain}$ carefully. In the next section we formalise the concept of relevant observations and how such relevance is transferred to a communication in $\mathcal{L}_{Domain}$.

3.4 Describing Interactions

In Section 3.2 we formalised how trust evaluations depend on the observations of specific interactions, and in Section 3.3 we specified how the combination of these trust evaluations, and some information about the interactions, can be used to learn an alignment. There is, however, an important issue which has so far been left undiscussed: how an agent can choose *what* properties of a set of interactions it should communicate when sending alignment messages. Specifically, there are two reasons why this is an important choice.

First, information that is not actually used in the trust model can be considered noise. While learning methods, such as ILP, distinguish between informative and non-informative items, the more there are of the latter, the higher the chance that the algorithm accidentally overfits, based on them. Even if this does not happen, it makes the learning process take longer than necessary. The problem is hard enough without adding extraneous information.

Secondly, and possibly more importantly, the aim is for the receiving agent to use the alignment to translate messages based on interactions that the agents have not shared. So far we have not really discussed what is important to

communicate in the $\mathcal{L}_{Domain}$ language because alignment is performed using a set of shared interactions. Both agents have their own observations of the set of interactions, and if need be, can draw upon this knowledge to learn a good alignment. In contrast, if the interaction is not shared, the receiving agent has to make do with only the information that is communicated by the witness. Our best chance of learning a useful alignment is thus to use only the information in the alignment messages, because if a GRA is learnt using information that is never communicated, this GRA can never be used to translate evaluations based on interactions that are not shared.

In the alignment process, it is unlikely that the receiver learns anything new about the interactions, but rather it should be able to learn something about the sender, by analysing what properties the sender chooses to communicate. For this to be possible, and thus for the receiver to learn as accurate as possible an alignment, it depends in a large part on the sender communicating only *relevant* aspects of the interactions and doing this *consistently*. In this section, we investigate what these concepts mean in the context of trust alignment.

3.4.1 Relevance

While Section 3.3 focused on the receiver of alignment messages and we analysed how these could be used to learn an alignment, this section focuses on what the sender can do to help the receiver. We are thus assuming that this agent is *benevolent*: it will act in the best interest of the receiver. However, our method specifically deals with some of the privacy issues a sender might have. Within the bounds of what information is relevant, the sender is free to choose to omit some of this, as long as it does this consistently. We now formalise the intuitive meaning of relevance and consistency. Before we define relevance in the communications, however, we need to define what it means within an agent's trust model.

Let $\models_A$ be agent A's trust model with regards to target T, as in Definition 3.3, and let $\alpha \in \mathcal{L}_{Trust}[T]$ be a trust evaluation. We recall that a set of observations $O \subseteq \mathfrak{O}_A$ is said to support α, if $O \models_A \alpha$. This however, is not quite specific enough because all this states is that there are elements in O that support α. What we really want to find is a smallest set that still supports α. More formally, we say that a set O is *relevant* to α iff $O \models_A \alpha$ and for all sets $O' \subset O$, we have $O' \not\models_A \alpha$.

This condition captures our intuition in a fairly straightforward manner: we want to work with a smallest set of observations that supports a trust evaluation. Because $\mathfrak{O}_A$ is a set of logical sentences in a logical language $\mathcal{L}_{\mathfrak{O}_A}$ (as we mentioned in Definition 3.1 on page 38), we can generalise the definition of a relevant set to use the entailment relation rather than the subset relation. We interpret $O \subseteq \mathfrak{O}_A$ as a conjunction of the observations. The intuition behind this is that we want a set O to hold if and only if *all* sentences $o \in O$ hold. With $\mathcal{L}_{\mathfrak{O}_A}$ a logical language, we use entailment to give us a broader interpretation of relevance. We say that a set of observations O is relevant to a trust evaluation α, if $O \models_A \alpha$, and for any set O' such that $O \models O'$, we have $O' \not\models_A \alpha$. We can

give the same definition using a generalisation operator (see Definition 3.24 below), which will prove useful throughout this section. A generalisation operator is defined as follows:

Definition 3.22 (Generalisation operator [De Raedt, 2008, page 56])**.** For a language $\mathcal{L}_{\mathfrak{O}_A}$, a *generalisation operator* $\rho_g : \mathcal{L}_{\mathfrak{O}_A} \to \mathcal{P}(\mathcal{L}_{\mathfrak{O}_A})$ is defined as a function such that for all $o \in \mathcal{L}_{\mathfrak{O}_A}$ we have $\rho_g(o) = \{o' \in \mathcal{L}_{\mathfrak{O}_A} \mid o \models o'\}$, with $\models$ the entailment relation of $\mathcal{L}_{\mathfrak{O}_A}$. We say that for any $o' \in \rho(o)$, o' is more general than o, with notation: $o' \succeq o$. Using this definition we will define the *meet* of two observations: $o_1 \sqcap o_2 = \rho_g(o_1) \cap \rho_g(o_2)$.

Note that this operator works in essentially the reverse manner from the generalisation of SRAs to GRAs. Whereas a GRA entails the SRAs it generalises, in this case it is the other way round: the specific observation that is generalised now entails the more general one. In an ILP setting this corresponds with learning from satisfiability rather than learning from entailment. The intuition behind this switch is that in the former case, a GRA is a descriptor for a class of objects: it specifies the sufficient properties to distinguish among classes. In the latter case we do not want to distinguish among sets of observations, but rather find necessary properties for a set of observations to support a trust evaluations. Thus for generalising SRAs we find universally quantified disjunctions of sentences, whereas generalisation of observations results in existentially quantified conjunctions of sentences. For a detailed account of the relation between the two logical settings, we refer to De Raedt et al. [1997].

θ-subsumption provides us with a way of generating a generalisation operator as follows: let φ be a sentence in a Horn language then $\rho(\varphi) = \{\varphi' \mid \varphi' \ \theta\text{-subsumes } \varphi\}$ is a generalisation operator. As an example of this operator in practice, assume $o = title(a) \wedge author(a, b)$, then we obtain:

$$\begin{aligned}\rho(o) = \{\ & \top,\ author(X,Y),\ title(X), author(a,X),\ author(X,b),\\ & title(a),\ author(a,b),\ title(X) \wedge author(Y,Z),\\ & title(X) \wedge author(X,Y),\ title(a) \wedge author(X,Y),\\ & title(X) \wedge author(a,Y),\ title(X) \wedge author(Y,b),\\ & title(X) \wedge author(X,b),\ title(X) \wedge author(a,b),\\ & title(a) \wedge author(X,b),\ title(a) \wedge author(a,X),\\ & title(a) \wedge author(a,b)\ \}\end{aligned}$$

with all variables existentially quantified. This set is not exhaustive, and all sentences that are logically equivalent (such as $author(a, b) \wedge title(a)$) to those in the set are also in the generalisation. As we see, the number of generalisations of a single conjunction is large. Luckily, we are not interested in all of the generalisations, but only the *relevant* generalisations. Relevance is defined for sets of observations, so we similarly need to define the generalisation operator for sets of observations. We extend the generalisation operator ρ to generalise sets of sentences as follows:

Definition 3.23 (Generalisation operator for sets)**.** Let $O \subseteq \mathfrak{O}_A$, then if we enumerate all elements in O with $o_1, \ldots, o_n$, we define the generalisation operator $\rho(O) = \{\{p_1, \ldots, p_n\} \mid p_i \in \rho(o_i) \text{ for } i \in [1, n]\}$, with variable substitution to ensure each p_i uses a unique set of variables. Intuitively, it means that every possible generalisation of O contains one possible generalisation for each $o \in O$, and the set $\rho(O)$ is all possible combinations of the generalisations of the elements of O.

This operator allows us to also extend the definition of the meet: $O_1 \sqcap O_2 = \{o \mid \exists o_1 \in \rho(O_1), o_2 \in \rho(O_2) : o \equiv o_1 \wedge o \equiv o_2\}$. We need to define it in this manner, rather than simply taking the intersection of $\rho(O_1)$ and $\rho(O_2)$, because if O_1 and O_2 have different cardinalities, there might still be logical generalisations; we need to interpret each set O as an existentially quantified conjunction of sentences in $\mathcal{L}_{\mathfrak{O}_A}$ and consider the equivalence classes in the meet.

We use this generalisation operator to define a relevant set of observations.

Definition 3.24 (Relevant observations)**.** Let $O \subseteq \mathfrak{O}_A$ be a set of observations, and $\models_A$ be the trust model of A. Furthermore let $\alpha \in \mathcal{L}_{Trust}[T]$ be a trust evaluation with target T. We say the set of observations O is *relevant* to trust evaluation α iff:

- $O \models_A \alpha$.
- For any set $O' \in \rho(O)$, we have that $O' \not\models_A \alpha$ or $O = O'$.

Note that this new definition of a relevant set is a generalisation of the original description. The subset relation in $\mathfrak{O}_A$ can be used to generate a generalisation operator, and by using this generalisation operator in Definition 3.24, we have the original description back. The difference is that we can now also use other generalisation operators, such as the generalisation operator generated by θ-subsumption that we discussed earlier. We leave the choice of which generalisation operator to use to the designer because we do not want to limit agents with regards to the internal representation language they use. We only make the following assumption: for any agent A, $\mathcal{L}_{\mathfrak{O}_A}$ is a logic in which a generalisation operator can be defined. We continue our running example of Alice searching for a trustworthy author, and show how relevance can be defined in Alice's observation language.

Example 3.13. We recall from Example 3.3 (page 40) that the following holds for Alice:

$$Observe_{Alice}(\{article_1, article_2\}) \models_{Alice} trust(Dave, 4)$$

So far we have not talked about the internal representation language $\mathcal{L}_{\mathfrak{O}_A}$ that Alice's agent has for observations. Let us assume this is a first-order language in the agent's belief base. Alice can have the following representation of $\{article_1, article_2\}$:

$$\begin{aligned}Observe_{Alice}(\{article_1, article_2\}) &= observe_{Alice}(article_1) \cup observe_{Alice}(article_2)\\ =\{&author(\text{`}ArticleOne\text{'}, \text{`}Dave\text{'}) \ \wedge \ publication(\text{`}ArticleOne\text{'}, \text{`}ConferenceA\text{'})\\ &\wedge \ readable(\text{`}ArticleOne\text{'}) \ \wedge \ original(\text{`}ArticleOne\text{'}),\\ &author(\text{`}ArticleTwo\text{'}, \text{`}Dave\text{'}) \ \wedge \ topic(\text{`}ArticleTwo\text{'}, \text{`}TopicX\text{'}) \ \wedge\\ &\neg readable(\text{`}ArticleTwo\text{'}) \ \wedge \ original(\text{`}ArticleTwo\text{'})\}\end{aligned}$$

While this set supports $trust(Dave, 4)$, there is a lot of information that is probably not necessary. For instance, if we assume that Alice's trust model evaluates Dave the same based on any two articles written by him, as long as they are original, then a relevant set does not need to include statements about, for example, readability. We use θ-subsumption to define our generalisation operator ρ. This results in the following set of relevant observations:

$$\begin{aligned}O_{rel} = \{&author(\text{`}ArticleOne\text{'}, \text{`}Dave\text{'}) \wedge original(\text{`}ArticleOne\text{'}),\\ &author(\text{`}ArticleTwo\text{'}, \text{`}Dave\text{'}) \wedge original(\text{`}ArticleTwo\text{'})\}\end{aligned}$$

This set still supports $trust(Dave, 4)$ in Alice's trust model, but any generalisation does not: any generalisation removes some essential information. Either the information that there are two different articles, that *Dave* is the author of both articles, or the information that both articles are original. If, however, the trust model were only to require a single original article to support the trust evaluation for *Dave*, we would obtain $\{author(X, \text{`}Dave\text{'}) \wedge original(X), \ \top\}$ as the relevant set of observations from $Observe_{Alice}(\{article_1, article_2\})$. While this also matches with just $Observe(\{article_1\})$ or $Observe(\{article_2\})$ as well, the fact that there are two articles would, in this latter case, actually be irrelevant information.

Now that we have defined relevance for observations, we can define relevance for messages. The translation between observations and the description of the interactions in $\mathcal{L}_{Domain}$ should, in some way, maintain relevance. Relevance, however, is defined with regards to an agent's trust evaluation: relevance is agent-specific. We should thus define it in terms of how an agent translates its observations into $\mathcal{L}_{Domain}$. We define a translation between an agent's observation language $\mathcal{L}_{\mathfrak{O}_A}$ and $\mathcal{L}_{Domain}$ as a pair of functions:

Definition 3.25 (Translation between languages)**.** A *translation* between two languages, in our case $\mathcal{L}_{Domain}$ and $\mathcal{L}_{\mathfrak{O}_A}$ consists of two binary relations η^{up} and η^{down}, such that for a formula $\varphi \in \mathcal{L}_{Domain}$, $\eta^{up}(\varphi) \subseteq \mathcal{L}_{\mathfrak{O}_A}$ and for an $o \in \mathcal{L}_{\mathfrak{O}_A}, \eta^{down}(o) \subseteq \mathcal{L}_{Domain}$.[5]

A desirable property of a translation is for both η^{up} and η^{down} to be functional; in other words, they associate at most a single element with each element in their domain. If both are functional we speak of a *functional translation*.

Because we are mainly interested in translations of sets of observations, we define H^{up} and H^{down} as the extension of translation η to sets: for a set of

[5] We use functional notation for a relation $R \subseteq S_1 \times S_2$ to mean that, for $a \in S_1$, we write $R(a) = \{b \in S_2 \mid aRb\}$.

observations O, we have $H^{down}(O) = \prod_{o \in O} \eta^{down}(o)$ and similarly for H^{up}. It is trivial to check that if η is functional, then the corresponding H is also functional.

Example 3.14. Alice could use the following function η^{down} to translate her observations of the interactions $\{article_1, article_2\}$:

$$\eta^{down}(author(\text{`}ArticleOne\text{'}, \text{`}Dave\text{'})) = \{author(\text{`}ArticleOne\text{'}, \text{`}Dave\text{'})\}$$
$$\eta^{down}(publication(\text{`}ArticleOne\text{'}, \text{`}ConferenceA\text{'})) = \{publication(\text{`}ArticleOne\text{'}, \text{`}ConferenceA\text{'})\}$$
$$\eta^{down}(readable(\text{`}ArticleOne\text{'})) = \{\}$$
$$\eta^{down}(original(\text{`}ArticleOne\text{'})) = \{\}$$
$$\eta^{down}(author(\text{`}ArticleTwo\text{'}, \text{`}Dave\text{'})) = \{author(\text{`}ArticleTwo\text{'}, \text{`}Dave\text{'})\}$$
$$\eta^{down}(topic(\text{`}ArticleTwo\text{'}, \text{`}TopicX\text{'})) = \{topic(\text{`}ArticleTwo\text{'}, \text{`}TopicX\text{'})\}$$
$$\eta^{down}(\neg readable(\text{`}ArticleTwo\text{'})) = \{\}$$
$$\eta^{down}(original(\text{`}ArticleTwo\text{'})) = \{\}$$

This is a function, so if we have η^{up} functional as well, then this forms a functional translation between $\mathcal{L}_{\mathfrak{O}_{Alice}}$ and $\mathcal{L}_{Domain}$.

We can use such a translation to define the relevance of messages in $\mathcal{L}_{Domain}$:

Definition 3.26 (Relevance maintaining translation)**.** Let $O \subseteq \mathcal{L}_{\mathfrak{O}_A}$ be a set of relevant observations for agent A's trust evaluation α and let H be a functional translation between $\mathcal{L}_{\mathfrak{O}_A}$ and $\mathcal{L}_{Domain}$. The message $\Psi = H^{down}(O)$ is relevant to agent A's trust evaluation α iff the set $H^{up}(\Psi) \in \rho(O)$. In other words, the translation does not add any irrelevant properties. A translation *maintains relevance* if and only if for all relevant observations O, with respect to all of agent A's trust evaluations α, $(H^{up} \circ H^{down})(O) \in \rho(O)$.

A special case is if $(H^{up} \circ H^{down})(O) = O$ for all $O \subseteq \mathcal{L}_{\mathfrak{O}_A}$, in which case there is a subset $X \subseteq \mathcal{L}_{Domain}$ such that $\mathcal{L}_{\mathfrak{O}_A} \cong X$.

Note that this definition means that in the general case, if $O \models_A \alpha$, then $(H^{up} \circ H^{down})(O) \not\models_A \alpha$. This is consistent with our initial thoughts. Due to privacy and incomplete translation functions it isn't always possible to translate all the relevant observations to $\mathcal{L}_{Domain}$, but at least those properties which are communicated should be a generalisation of those which were used for calculating the trust evaluation.

In Section 3.4.3 we will return to the concept of relevance, and specifically, the translation function η, but first we move on to the second desirable property of the communicated information regarding a trust evaluation: that an agent is consistent in its information given similar situations.

3.4.2 Consistency

To define consistency between similar situations, we first need a method for deciding when situations are similar. We start with a very broad definition that

focuses on trust evaluations. Trust evaluations are normally quite easily compared. If they are numerical, we can simply use the absolute difference between the two numerical values as the distance between two values. Other evaluations require other distance metrics, but these are generally not problematic. For instance, Sabater-Mir and Paolucci [2007] give a distance metric for trust evaluations represented by probability distributions and Abdul-Rahman and Hailes [2000] use a discrete representation and translate this to corresponding numbers in order to use a distance metric.

We use such a distance metric to define equivalence classes of trust evaluations: two trust evaluations $\alpha_1[T]$ and $\alpha_2[T']$ are equivalent if, given a distance metric d, $d(\alpha_1, \alpha_2) = 0$. Note that this states nothing about the equality of the targets of α_1 and α_2. In fact, the point of this is mainly to allow us to abstract away from the target: in most cases the distance metric d will work only with the actual trust value and disregard the target. Thus, in our example $\mathcal{L}_{Trust}$ we can use Euclidean distance on the values and therefore $trust(Jimmy, 4)$ and $trust(Dave, 4)$ are equivalent. We use this as a first step in defining when an agent is consistent by defining when two sets of observations are comparable:

Definition 3.27 (Comparable observations)**.** Let $O_1, O_2 \subseteq \mathfrak{O}_A$ be two sets of observations of an agent A and d a distance metric on $\mathcal{L}_{Trust}$, then O_1 and O_2 are *comparable* for agent A iff $O_1 \models_A \alpha_1$, $O_2 \models_A \alpha_2$ and $d(\alpha_1, \alpha_2) = 0$.

This, however, is just the first criterion. Two sets of observations are comparable if they support equivalent trust evaluations, but two wildly different sets of observations might satisfy this condition. In addition, we thus need some kind of similarity measure on observations.

This similarity measure is defined as follows, using the generalisation operator as in Definition 3.23 (page 60).

Definition 3.28 (Similar observations)**.** Let $O_1, O_2 \subseteq \mathfrak{O}_A$ be two sets of observations of an agent A and ρ a generalisation operator in $\mathfrak{O}_A$, then these observations are *similar* iff there is an $O^* \in O_1 \sqcap O_2$ such that $O^* \not\equiv \top$. In other words there is a non-trivial generalisation of both O_1 and O_2.

Example 3.15. We assumed in Example 3.13 that Alice uses first-order logic in her belief base and a generalisation operator based on θ-subsumption. We recall the observations of interactions $\{article_1, article_2\}$ from Example 3.13:

$$\begin{aligned} O_1 =& Observe_{Alice}(\{article_1, article_2\}) = observe_{Alice}(article_1) \cup observe_{Alice}(article_2) \\ =& \{author(\text{`}ArticleOne\text{'}, \text{`}Dave\text{'}), publication(\text{`}ArticleOne\text{'}, \text{`}ConferenceA\text{'}), \\ & readable(\text{`}ArticleOne\text{'}), original(\text{`}ArticleOne\text{'}), \\ & author(\text{`}ArticleTwo\text{'}, \text{`}Dave\text{'}),\ topic(\text{`}ArticleTwo\text{'}, \text{`}TopicX\text{'}), \\ & \neg readable(\text{`}ArticleTwo\text{'}), original(\text{`}ArticleTwo\text{'})\} \end{aligned}$$

Now if Alice observes another set of articles, $\{article_3\}$ with the following observations:

$$O_2 = Observe_{Alice}(\{article_3\}) = \{author(\text{`}ArticleThree\text{'}, \text{`}Jimmy\text{'}),$$
$$publication(\text{`}ArticleThree\text{'}, \text{`}JournalAlpha\text{'}),$$
$$topic(\text{`}ArticleThree\text{'}, \text{`}TopicX\text{'}), original(\text{`}ArticleThree\text{'})\}$$

We see that these two sets are similar because there is an

$$O^* = \{author(X, A), publication(X, B), topic(Y, \text{`}TopicX\text{'}), original(X)\}$$

that θ-subsumes both sets with:

$$\theta_1 = \{X/\text{`}ArticleOne\text{'}, A/\text{`}Dave\text{'}, B/\text{`}ConferenceA\text{'}, Y/\text{`}ArticleTwo\text{'}\}$$
$$\theta_2 = \{X/\text{`}ArticleThree\text{'}, A/\text{`}Jimmy\text{'}, B/\text{`}JournalAlpha\text{'}, Y/\text{`}ArticleThree\text{'}\}.$$

Furthermore, if $Observe_A(\{article_3\}) \models_A trust(\text{`}Jimmy\text{'}, 4)$ then the sets of observations are comparable as well as similar.

An obvious example of dissimilar observations is if the interaction is completely unrelated, such as Alice's observations of a tennis match: $Observe_{Alice}(\{match_1\}) = \{players(\text{`}Jimmy\text{'}, \text{`}Dave\text{'}) \wedge score(\text{`}2\text{–}1\text{'})\}$. The only sentence that θ-subsumes Alice's observations of the initial two articles on the one hand, and her observations of the tennis match on the other, is $\top$. All the same, these observations might be comparable, because based on the tennis match, Alice might compute that $trust(\text{`}Jimmy\text{'}, 4)$.

The combination of comparable and similar observations allows us to define consistency for a translation.

Definition 3.29 (Consistency)**.** Translation H is *consistent* iff for all $O_1, O_2 \subseteq \mathfrak{O}_A$ that are similar and comparable observations of an agent A and all non-trivial $O^* \in O_1 \sqcap O_2$ we have that $H^{down}(O^*)$ generalises both $H^{down}(O_1)$ and $H^{down}(O_2)$ in $\mathcal{L}_{Domain}$.

For this definition we do not strictly need O_1 and O_2 to be comparable, only similar. Nevertheless, given that our aim is to give desirable properties of a translation for similar trust evaluations we give the broadest definition of consistency that allows us to do this. If O_1 and O_2 support dissimilar trust evaluations we are not interested in what their translation is.

Example 3.16. Alice's translation function η from Example 3.14 is also able to translate her observations O_2 of $\{article_3\}$:

$$\eta^{down}(author(\text{`}ArticleThree\text{'}, \text{`}Jimmy\text{'})) = \{author(\text{`}ArticleThree\text{'}, \text{`}Jimmy\text{'})\}$$
$$\eta^{down}(publication(\text{`}ArticleThree\text{'}, \text{`}JournalAlpha\text{'}))$$
$$= \{publication(\text{`}ArticleThree\text{'}, \text{`}JournalAlpha\text{'})\}$$
$$\eta^{down}(topic(\text{`}ArticleThree\text{'}, \text{`}TopicX\text{'})) = \{topic(\text{`}ArticleThree\text{'}, \text{`}TopicX\text{'})\}$$
$$\eta^{down}(original(\text{`}ArticleThree\text{'})) = \{\}$$

and also translates O^* as follows:

$$\eta^{down}(author(X, A)) = \{author(X, A)\}$$
$$\eta^{down}(publication(X, B)) = \{publication(X, B)\}$$
$$\eta^{down}(topic(Y, \text{`}TopicX\text{'})) = \{topic(Y, \text{`}TopicX\text{'})\}$$
$$\eta^{down}(original(X)) = \{\}$$

This is a *consistent* translation: the only similar and comparable sets of observations in $\mathfrak{O}_{Alice}$ are O_1 and O_2 and $H^{down}(O^*)$ clearly θ-subsumes $H^{down}(O_1)$ and $H^{down}(O_2)$. We can change this to an inconsistent translation by using, for instance $\eta^{down}(topic(\text{`}ArticleThree\text{'}, \text{`}TopicX\text{'})) = \{\}$. This way $H^{down}(O^*)$ does not θ-subsume $H^{down}(O_2)$. This is also the type of inconsistent communication we wish to avoid: if the agent chooses to communicate the topic of an article, it should do so consistently when the articles are similar and comparable.

We are interested in translations that *maintain relevance* (Definition 3.26) and are *consistent* (Definition 3.29). A further desirable property of agents in general is that they are *coherent*. With this we mean that observations are similar if and only if the underlying interactions are similar: this is useful so that the receiving agent, when identifying the interactions and its own observations of those interactions is guaranteed to observe similarly coherent properties. We do not formalise this property as we have the other two, because it depends on a similarity measure for the interactions, which is not always available. One way to do it, is in terms of interactions as Herbrand models of $\mathcal{L}_{Domain}$ and use θ-subsumption as a generalisation operator, but that does not seem in the spirit of what interactions really are. We therefore leave this as an open question, that depends on the domain and how interactions are actually defined in it.

3.4.3 Galois connection

Now that we have specified the desirable properties that the sending agent and its translation functions should have, let us look at properties of pairs of binary relations. Specifically, if the translation relations form a *Galois connection*, then the properties of consistency and relevance are guaranteed. A Galois connection is defined as a correspondence relation between two partially ordered sets (posets). We start by giving the definition of a Galois connection, then prove the properties hold, and finally describe why the fact that the translation can be any Galois connection is a useful property to know.

Definition 3.30 (Galois connection)**.** Let $(A, \leq)$ and $(B, \leq)$ be two partially ordered sets (posets). Let $F : A \to B$ and $G : B \to A$ be two monotone functions. We call these functions a (monotone) *Galois connection* between A and B iff for all $a \in A$ and $b \in B$: $F(a) \leq b \Leftrightarrow a \leq G(b)$. In this connection F is called the lower and G the upper adjoint and we will use the notation $(F, G) : A \overset{Galois}{\Longrightarrow} B$

This same definition holds for preordered sets (prosets) as well as for posets.

A useful property of Galois connections, that we will use later, is the following:

Property 3.4.1. Let $(A, \leq)$ and $(B, \leq)$, and functions F, G such that $(F, G) : A \xrightarrow{Galois} B$. Then for any $a \in A$ and $b \in B$: $a \leq (G \circ F)(a)$ and $(F \circ G)(b) \leq b$.

To even start showing that a Galois connection fulfils the properties of maintaining relevance and being consistent, we need to show that the observations and $\mathcal{L}_{Domain}$ are prosets with the right ordering. Because we want to talk about sets of observations and descriptions in $\mathcal{L}_{Domain}$, we will give the result directly for the power sets.

Lemma 3.4.1. *$(\mathcal{P}(\mathfrak{O}_A), \succeq)$, with $\succeq$ defined as $O_1 \succeq O_2$ iff $O_1 \in \rho(O_2)$, is a proset.*

Proof. We need to prove reflexivity and transitivity of $\succeq$.

- Reflexivity is trivial. Let $O \subseteq \mathfrak{O}_A$ then $O \in \rho(O)$ and thus $O \succeq O$
- Transitivity is also straightforward. Let O_1, O_2 and $O_3 \subseteq \mathfrak{O}_A$ such that $O_1 \succeq O_2$ and $O_2 \succeq O_3$. Then we know that $O_1 \in \rho(O_2)$ and thus, from Definition 3.23, $O_2 \models O_1$. Analogously we know that $O_3 \models O_2$. From the transitivity of entailment we know that $O_3 \models O_1$ and thus that $O_1 \in \rho(O_3)$. Therefore $O_1 \succeq O_3$.

□

Lemma 3.4.2. *$(\mathcal{P}(\mathcal{L}_{Domain}), \succeq_\theta)$ is a proset.*

Proof. The proof is given by De Raedt [2008] and is very similar to that of Lemma 3.4.1. □

We can now prove that a translation H is both *relevance maintaining* and *consistent* if it is a Galois connection. For H to be a Galois connection, both H^{up} and H^{down} must be monotone with regards to the ordering we use. This needs to be checked for any translation function. For the relevance-maintaining and consistent translation of Examples 3.14 and 3.16 the translation is monotone.

Theorem 2:
Let $(H^{down}, H^{up}) : \mathcal{P}(\mathfrak{O}_A) \xrightarrow{Galois} \mathcal{P}(\mathcal{L}_{Domain})$, then, given O, a set of relevant observations for agent A's trust evaluation α, $H^{down}(O)$ is a relevant message as in Definition 3.26 and therefore the translation H maintains relevance.

Proof. The proof follows trivially from Property 3.4.1. □

Theorem 3:
Let $(H^{down}, H^{up} : \mathcal{P}(\mathfrak{O}_A) \xrightarrow{Galois} \mathcal{P}(\mathcal{L}_{Domain})$, then H is consistent.

Proof. Let $O_1, O_2 \subseteq \mathfrak{O}_A$ be similar and comparable observations of an agent A and ρ a generalisation operator in $\mathfrak{O}_A$. We prove the theorem from contradiction.

1. Assume there is no $O^* \in O_1 \sqcap O_2$ such that $H^{down}(O^*) \succeq_\theta H^{down}(O_1)$.
2. We rewrite (1): for all $O^* \in O_1 \sqcap O_2$ we have $H^{down}(O^*) \not\succeq_\theta H^{down}(O_1)$.
3. From the fact that O_1 and O_2 are similar, we know that there is an $O^* \in O_1 \sqcap O_2$. This means that $O^* \succeq O_1$.
4. With H a Galois connection and Property 3.4.1 we also know that $(H^{up} \circ H^{down})(O^*) \succeq O^*$
5. From transitivity of $\succeq$ and (4) we know that $(H^{up} \circ H^{down})(O^*) \succeq O_1$.
6. Now from Definition 3.30 we know $H^{down}(O^*) \succeq_\theta H^{down}(O_1)$. However, this contradicts (2) and thus our assumption. From contradiction follows that there is an $O^* \in O_1 \sqcap O_2$ such that $H^{down}(O^*) \succeq_\theta H^{down}(O_1)$.

For O_2 the proof is analogous and we obtain that H is consistent. □

Therefore, if the translation an agent uses between its own representation language and $\mathcal{L}_{Domain}$ is a Galois connection, then it satisfies these two properties. This is a useful result, because Galois connections are a commonly used tool when trying to find useful representations of concepts. For instance, they form an integral part of the theory of Formal Concept Analysis [Ganter and Wille, 1999]. This means that, in general, the existing tools for finding translations between an internal representation and $\mathcal{L}_{Domain}$ will satisfy these properties.

Furthermore, there are relevance-maintaining and consistent translations that are *not* Galois connections; specifically, if H^{down} is not surjective. Let, for instance, $o \in \mathcal{L}_{\mathfrak{O}_A}$ and $\varphi, \psi \in \mathcal{L}_{Domain}$ such that $\psi \not\succeq_\theta \varphi$. Also let η be a translation such that $\eta^{down}(o) = \varphi$, $\eta^{up}(\varphi) = o$, $\eta^{up}(\psi) = o$, and there is no $o' \in \mathcal{L}_{\mathfrak{O}_A}$ such that $\eta^{down}(o') = \psi$. Then this translation is relevance-maintaining and consistent, but not a Galois connection.

Example 3.17. An example of a relevant and consistent translation that is not a Galois connection is obtained, for instance, if the $\mathcal{L}_{Domain}$ of our example were to, in addition to having the *author* predicate, allow further distinction between $first\_author$ and $other\_author$ as different classes of author. If an agent does not care for this distinction and its internal representation thus only uses the predicate author then we might run into problems. For instance, let us use the translation function η defined as follows: for any terms X, Y it has $\eta^{down}(author(X,Y)) = \{author(X,Y)\}$, $\eta^{up}(author(X,Y)) = \{author(X,Y)\}$ and $\eta^{up}(first\_author(X,Y)) = \{author(X,Y)\}$, then it is immediately obvious that η does not form a Galois connection, despite maintaining relevance. We have $\{author(X,Y)\} \preceq H^{up}(first\_author(X,Y))$, but not $H^{down}(\{author(X,Y)\}) \preceq_\theta \{first\_author(X,Y)\}$. To be able to conclude that $\{author(X,Y)\}$ is more general than $\{first\_author(X,Y)\}$ we need an extra rule, $first\_author(X,Y) \rightarrow author(X,Y)$, from the background information in $\mathcal{L}_{Domain}$, which is not used in θ-subsumption. In this case we can make the translation a Galois connection by using θ-subsumption with background information to consider generality in $\mathcal{L}_{Domain}$, but as shown above, in the general

case a relevance-maintaining and consistent translation is not always a Galois connection.

Nevertheless, the condition for relevance clearly does hold (on this fragment of the language): $(H^{up} \circ H^{down})(\{author(X,Y)\}) \in \rho(\{author(X,Y)\})$. Note that the lack of surjectivity alone is not sufficient to say that a relevant and consistent translation is not a Galois connection. Consider the same case, but $\eta^{up}(first\_author(X,Y)) = \bot$, and of course, $\eta^{down}(\bot) = \bot$ (where $\bot = \neg\top$). Now it clearly is a Galois connection.

In the normal case, if we use the $\mathcal{L}_{Domain}$ from Figure 3.3 (page 49), then the translation function in Example 3.14 is, in fact, a Galois connection; however, it also makes it clear that the properties so far are not the only desirable properties. For instance we have that $\eta^{down}(original(X)) = \emptyset$. This is problematic, because, given the description of Alice's trust model in Example 3.13, originality of an article is the main criterion she uses for evaluating it. The distinction between important and unimportant concepts is dependent on the agent and the trust model, and whether these can be translated into $\mathcal{L}_{Domain}$ also depends on the expressiveness of this language. The designer of the system thus has an important task in deciding *which* Galois connection to use for the translation, but in order to not add extraneous irrelevant information to a message and to maintain consistency between messages describing similar observations, this translation should be a Galois connection.

Finally, we return to another important property that we require of the translation. We talked about the $\mathcal{L}_{Domain}$ message needing to uniquely identify the set of interactions used to calculate the trust evaluations in Section 3.3 (page 47). An agent receiving an alignment message must be able to identify the interactions in order to use its own observations of these interactions and calculate the trust evaluation they support. The translation must thus maintain the identity of the observations.

Definition 3.31 (Invariance with regards to interactions)**.** To define invariance for a translation we must first define what it means for a set of observations to uniquely identify a set of interactions: let A be an agent and $O \subseteq \mathfrak{O}_A$ a set of observations, then O *uniquely identifies* a set of interactions $I \subseteq \mathfrak{I}$ iff $O \succeq Observe(I)$ and for all $I' \subseteq \mathfrak{I}$ we have that if $O \succeq Observe(I')$ then $I = I'$.

We can now define invariance as follows: let A be an agent and $I \subseteq \mathfrak{I}$ then a translation H is *invariant* with regards to I iff for all $O \subseteq Observe(I)$ that uniquely identify I we have that $(H^{up} \circ H^{down})(O)$ uniquely identifies I. A translation is said to be *invariant with regards to interactions* if it is invariant with regards to all $I \subseteq \mathfrak{I}_{|A}$.

An agent with a translation that is invariant to interactions translates its observations into $\mathcal{L}_{Domain}$ in such a way that the interactions can still be identified. It also translates $\mathcal{L}_{Domain}$ messages that identify interactions into its internal representation language in such a way that the interactions can be identified. This is essential for the translating of alignment messages, because agents need to identify the sets of interactions to be able to calculate their trust evaluations.

Invariance with regards to interactions is not automatically satisfied by a Galois connection, and in fact, creates problems for a translation. On the one hand we do not want to communicate irrelevant information, and on the other hand we need to identify interactions, usually using predicates that are irrelevant to the trust model. A possible solution is to extend the definition of alignment messages to have two parts in $\mathcal{L}_{Domain}$, one for communicating relevant information and the other for specifying what set of interactions was used. We do not take this approach in this thesis: in the next chapter we allow agents to communicate translations of supersets of the relevant set of observations, so the interactions can be uniquely specified. We allow agents to communicate somewhat irrelevant information so the translation is invariant with regards to interactions.

Example 3.18. We recall from Example 3.13 (page 60) that O_{rel} is the following set:

$$\{author(\text{`}ArticleOne\text{'}, \text{`}Dave\text{'}), original(\text{`}ArticleOne\text{'}),$$
$$author(\text{`}ArticleTwo\text{'}, \text{`}Dave\text{'}), original(\text{`}ArticleTwo\text{'})\}$$

This set identifies $\{article_1, article_2\}$. If Alice's translation H is invariant with regards to interactions, then her translation into $\mathcal{L}_{Domain}$ must also uniquely identify this set. A message sent, using this translation, thus allows any agent to identify the interactions $\{article_1, article_2\}$ and calculate its own trust evaluation based on these interactions.

Nevertheless, there is still the problem that there is no translation for the predicate *original* into $\mathcal{L}_{Domain}$, and Alice can therefore not communicate this, most relevant, aspect of her observations. Alice suspects that the venue an article was published at, is correlated with the originality of the work. Because this is the best information she can give, using the $\mathcal{L}_{Domain}$ ontology in Figure 3.3, she disregards the actual relevant information $original(\text{`}ArticleOne\text{'})$ and uses instead $publication(\text{`}ArticleOne\text{'}, \text{`}ConferenceA\text{'})$, that is not strictly relevant to her. Because she suspects the correlation, she hopes the receiver can use it to learn a relation between certain publication venues and her trust evaluations.

This example makes it clear that it is neither easy to decide what needs communicating, nor to learn based on this. In this section we have given some useful properties to aid in this process, but it remains a hard problem.

3.5 Summary

In this chapter we have described a formal framework in which we define what the problem of trust alignment is and give a formal description of a possible solution. Furthermore, we address the problem of deciding what content to include in the alignment messages. The definition of trust alignment is based on Channel Theory, which gives a way to model the flow of information within a distributed system, and forms a solid foundation for trust alignment. The method for aligning uses θ-subsumption in order to generalise from the exchanged messages and

results in an alignment that can be used to translate the other agent's evaluations based on non-shared interactions. Finally, we show that if the translation function that an agent uses to generate messages in the domain language, based on its own internal representation, is a Galois connection, then some desirable properties for the communication hold.

For our framework to work we assume the agents and domain meet some basic properties. The first is that for any type of alignment, agents need a set of shared information from which they can start to find the alignment (this is formalised in Section 3.2.2). In our framework we assume this set of shared information are interactions that both agents can observe. The environment must provide the opportunity for such shared interactions to occur. Secondly, agents must be able to identify these interactions to each other and describe some relevant objective properties of these interactions in a shared language. Similarly, we assume there is some knowledge about the environment that all agents share. Finally, we assume agents are truthful about the interactions they have observed and their properties. Moreover, learning a successful alignment is dependent on the sender being in some way benevolent: it must attempt to communicate the properties of interactions that are relevant to the trust evaluation, and additionally, be consistent among similar sets of interactions. If the agents and domain meet these criteria, then agents can align using our framework.

Chapter 4

Alignment in Practice

The true method of discovery is like the flight of an aeroplane. It starts from the ground of particular observation; it makes a flight in the thin air of imaginative generalisation; and it again lands for renewed observation rendered acute by rational interpretation.

–Alfred North Whitehead

4.1 Introduction

The framework described in the previous chapter gives an abstract, theoretical description of the solution we are searching for; however, as pointed out in Section 3.3.4, an implementation has to overcome the various computational limitations of calculating least general generalisations and the θ-subsumption of formulae. Additionally, whether an actual implementation can learn a generalisation at all depends on $\mathcal{L}_{Domain}$ and what the sender chooses to communicate using it, as described in Section 3.4. In this chapter we will look at these problems in more detail, with the actual implementation of a trust alignment method in mind.

In Section 4.2, we present FORTAM, our solution to the problem. It uses a First-Order Regression algorithm to learn an alignment. We analyse it empirically, using a simulated environment that is based on the case study of the previous chapter. We are particularly interested in the number of interactions required to learn an alignment, and this depends on how hard the problem is. In Section 4.2.4, we define the relative complexity between different trust models and use this as a measure of the hardness of the alignment problem.

In Section 4.3 we compare our solution to some of the other methods for solving the trust alignment problem. Most notably, we compare FORTAM to Abdul-Rahman and Hailes' method [2000] and POYRAZ [Şensoy et al., 2009]. The former learns an alignment without taking the context into account and the latter is one of the filtering methods we described in Section 2.3.1. We see that FORTAM performs significantly better than any of the other methods.

4.2 Using ILP to Learn a Trust Alignment

The aim of a trust alignment method is to, briefly summarised, use the underlying evidence that the agents share to translate the other agent's subjective trust evaluation, so that it may be used accurately. In Section 3.3 we formulated this problem as a machine learning problem using the ILP paradigm. The problem, thus formulated, is to find a hypothesis that best describes the set of Specific Rules for Alignment (SRAs). To narrow it down, though, we have some specific knowledge about the intended output of an alignment: given an alignment message, we want to find a corresponding trust evaluation in our own frame of reference.

The learning algorithm that best corresponds to this is a (multiclass) *classification* algorithm. We consider the own trust evaluations as labels for classes and the alignment messages as the descriptions of the cases in the classes. An ILP classification algorithm learns a generalisation of these descriptions, which can be used to classify future alignment messages and find the corresponding own trust evaluation. If, however, the representation space for trust evaluations is large, we run the risk that each individual class is very small. A better representation is to cluster *similar* trust evaluations together and consider them as cases of same class. Given that trust evaluations are represented using $\mathcal{L}_{Trust}$, we need a way of describing similarity between concepts in this language.

In a first approach we make no assumptions about $\mathcal{L}_{Trust}$, except that there is an adequate distance measure on it, which we can then use to cluster the SRAs. This groups similar trust evaluations together and we can then find a prototype trust evaluation for each cluster as the representative label of the class. Finally, we use an ILP algorithm to learn a generalised description of the alignment messages corresponding to each class. If this ILP learner fails for any of the clusters, then we make this cluster larger and try again. This results in a set of Generalised Rules for Alignment (GRAs): the head of a GRA is the prototype of the cluster and the body is the generalised description learnt by the ILP algorithm. Furthermore, these GRAs can be ranked from specific to general, and for translation of any new incoming communication we simply find the most specific GRA that covers it and use the prototype of the cluster as the translation.

Previous work by Koster et al. [2010a,b] presented this method in detail. In this thesis, we choose not to go into further details, focusing rather on an improvement of this algorithm. By assuming we learn separate alignments for any predicate in $\mathcal{L}_{Trust}$ that we are interested in, and furthermore, that the values are represented numerically, we can learn far superior alignments. This assumption is justified by the fact that the majority of computational trust models (discussed in Section 2.2) do use some form of numerical representation; additionally, Pinyol et al. [2007] show how other representation methods for trust can be converted into a numerical value. Regardless, we include the clustering-based alignment method in our experimental comparison of Section 4.3, where we show how much First-Order Regression improves on it and go into more detail on the reasons for this.

4.2.1 First-Order Regression

First-Order Regression is a combination of ILP and numerical regression [Karalič and Bratko, 1997]. ILP was discussed in the previous chapter and numerical regression is a method for function approximation [Bishop, 2006]. The goal is to learn the (functional) relationship between the dependent and independent variables. In numerical regression (or regression analysis), the independent variables are assumed to be numerical, whereas First-Order Regression allows the independent variables to be sentences in a first-order logic. The dependent variable is, in both first-order and numerical regression, a numerical value [Russel and Norvig, 2010].

By assuming the trust evaluations use numerical values for the evaluation, we can use First-Order Regression, a form of supervised learning, rather than clustering, which is a form of unsupervised learning. Specifically, the assumption allows us to consider the SRAs as sample values for a function, with as domain the alignment messages and as range the own trust evaluations. For instance, consider the general form of an SRA $\alpha \rightarrow \beta, \psi$ and let $a = value(\alpha)$ with $value$ a function that extracts the numerical value of a literal in $\mathcal{L}_{Trust}$ (for instance, $value(trust(\text{`}Dave\text{'}, 4)) = 4$). We can interpret such an SRA as a sample from the unknown function $F : \mathcal{L}_{Trust} \times \mathcal{P}(\mathcal{L}_{Domain}) \rightarrow \mathbb{R}$, such that $F(\beta, \psi) = a$. Both $\mathcal{L}_{Domain}$ and $\mathcal{L}_{Trust}$ are first-order languages, so we can use First-Order Regression to learn an approximation of this function F, which allows us to translate alignment messages into the own frame of reference. There are two ways in which algorithms for First-Order Regression approach the problem, a top-down and bottom-up approach. We chose to use Tilde [Blockeel et al., 2002], which uses the top-down approach, because it shows impressive results and has a good interface, both for defining the examples and hypothesis language, as well as to interpret the results.

In Algorithm 2 we give the overview of Tilde-RT, the algorithm for learning a regression decision tree. The way it works is that it learns a (binary) *decision tree*, starting with just the root. A binary decision tree is a representation of a function that, given a new message, finds the corresponding own trust value by providing a sequence of binary *tests*. These tests are the tree's inner nodes, that consist of a sentence in the hypothesis language. If the sentence covers the new message, then the decision is to go down the left subtree and if it fails, go down the right subtree. This process is repeated until the node is a leaf. The leaf contains the result of the function, or the translation of the message: a numerical value that is the expected own trust value. Tilde represents this tree in a Prolog program and this program is the output of the learning algorithm. The Prolog program can be used with new messages to automatically translate them.

A very simple example of a decision tree is sketched in Figure 4.1. This decision tree has tests about the received message. For instance, if the trust evaluation in the received message has a value greater than three, then we go to the left branch of the tree, otherwise we go right. Eventually we end up in a leaf: this is the translation of the message into the own frame of reference.

Algorithm 2: The TILDE-RT algorithm (see Blockeel and De Raedt [1998])

Tilde-RT:
begin
  Input: $\mathcal{E}$, the set of SRAs to be generalised
  Input: $\mathcal{L}_H$, the hypothesis language
  Input: t, a target predicate
  $Value$:= **Predict(**t, $\mathcal{E}$**)**
  $Basic\_tree$:= **inode(**$\top$,**leaf(**$t(Value)$**)**, **leaf(**$\bot$**))**
  T' := **Grow_Tree(**$\mathcal{E}$, $Basic\_tree$, t, $\mathcal{L}_H$**)**
  T := **Prune(**T'**)**
  Output: T, the decision tree that best fits the data $\mathcal{E}$
end

Grow_Tree:
begin
  Input: E, a set of examples
  Input: T, an inner node of a decision tree
  Input: t, the target predicate
  Input: $\mathcal{L}_H$, the hypothesis language
  $candidates$:= **Generate_tests(**$T, E, \mathcal{L}_H$**)**
  T_{best} := **Optimal_split(**$candidates$, E, t**)**
  if Stop_crit(T_{best}, t, E**) then**
    $Value$:= **Predict(**t, E**)**
    $result\_tree$:= **leaf(**$t(Value)$**)**
  end
  else
    $T_{node} := T - T_{best}$
    $E_{left} := \{e \in E | T_{best} \models e\}$
    $E_{right} := \{e \in E | T_{best} \not\models e\}$
    $left\_tree$:= **Grow_Tree(**E_1, T_{best}, t, $\mathcal{L}_H$**)**
    $right\_tree$:= **Grow_Tree(**E_2, T, t, $\mathcal{L}_H$**)**
    $result\_tree$:= **inode(**$T_{node}, left\_tree, right\_tree$**)**
  end
  Output: $result\_tree$, the subtree starting in the current node
end

Algorithm 2 constructs such a decision tree by starting with a single node, with $\top$, the test that always succeeds, and thus all the examples are in the left child with as value the "predictor" of the set of SRAs $\mathcal{E}$, as calculated by **Predict**. In the standard implementation of TILDE this function simply calculates the mean of the target variable, in our case the own trust evaluations. In each subsequent step the tree is grown with a binary decision: the method **Grow_Tree** finds the test, using **Generate_tests**, that optimally splits the examples covered

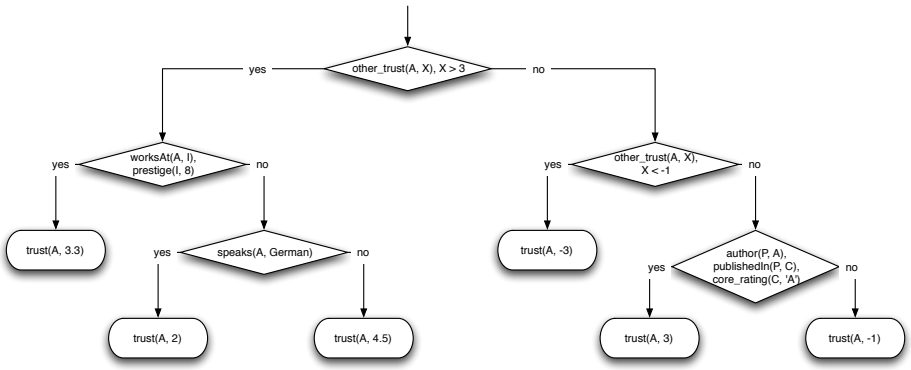


Figure 4.1: An example of a decision tree for translating the other's trust evaluation.

so far in that branch and extends the tree in this manner. The examples are then split according to whether they are covered or not by the new test and so it continues. **Optimal_split** uses the heuristic for regression as given by Blockeel et al. [1998], namely an F-test:

$$F = \frac{SS/(n-1)}{(SS_L + SS_R)/(n-2)}$$

with SS the squared error of the examples in the node: $SS(E) = \sum_{e \in E} \left(target\_value(e) - \textbf{Predict}(E)\right)^2$ and in our case $target\_value$ of course returns the value of the own trust evaluation of the example SRA. SS_L and SS_R is the same error for the left and right children in the test. **Optimal_split** chooses the test that gives the maximum F-value. **Stop_crit** also uses the F-value: if this is under a certain threshold it is not worth expanding the tree. Additionally, there is another stop criterion: if a split results in two children that cover a single example each, it is not worth expanding the tree.

The process continues until there are no more leaves to expand. After this TILDE-RT has the option to prune the resulting tree to reduce overfitting to the training data. Blockeel et al. [1998] describes how to do this using a validation set.

Decision trees and θ-subsumption

Learning a decision tree is a form of generalisation. Specifically the initial tree can be written as the most general generalisation $trust(Target, x) \leftarrow \top$, with x the mean value of all own trust evaluations. Each subsequent tree results in more specific generalisations. At any time we can represent the tree as a conjunction of Horn clauses, which is, in fact, what TILDE does when it generates the output as a Prolog program. While in the previous chapter we generally used a bottom-up approach to explain how ILP could find a trust alignment, it

turns out that in practice top-down is often a more suitable approach, because it is easier to find good heuristics for guiding the search. This is especially so, if there are only positive examples available, as is the case in regression. In classification it is usually the case that we have positive and negative examples for each class, but in regression we are learning a numerical function and thus any example should be interpreted as the result of the function for some element in its domain. Despite performing refinement, rather than generalisation, θ-subsumption still forms an integral part of the algorithm. It is used, together with the user-imposed language bias to generate the set of possible tests in the function **Generate_tests**.

Two alternative algorithms can be used to perform First-Order Regression, FORS [Karalič and Bratko, 1997] and S-CART [Kramer and Widmer, 2001]. S-CART is very similar to TILDE and in the literature performs similarly [Blockeel et al., 2004], although we did not test it for learning a trust alignment. FORS works differently and learns the rules directly, rather than finding a decision tree. In order to do this they use a different heuristic (coverage instead of splitting) for deciding how to search, which does not suit our specific problem description quite as well.

The language bias

One of the ways in which an ILP algorithm reduces the computational complexity of θ-subsumption is by not considering the entire search space. The type of tests it considers for adding to the tree are defined by a language bias; however, one has to be very careful in implementing such a bias: if we restrict the search space too much, then the algorithm might not find certain tests that could have improved the decision tree. For the example scenario we use in this section, we know what types of tests can lead to good choices, and in such cases, specifying the parameters in order to encompass interesting parts of the search space for good tests is relatively straightforward. In the general case, very good care must be taken to design the language bias correctly.

As briefly mentioned above, **Generate_tests** does not only use the user-defined bias to generate the set of possible tests. It also considers what tests could even be a generalisation of the examples in the first place. The information in $\mathcal{L}_{Domain}$ that the examples contain depends entirely on what the sender has communicated. This is why it is so important for the sender to communicate relevant information consistently, as we pointed out in Section 3.4.

In the continuation of this section we describe the experimental setup and give some results of the learning algorithm, under the assumption that the sending agent does its best to send relevant and consistent information, and using a well-designed language for learning.

4.2.2 Experimental setup

We demonstrate how TILDE learns an alignment, using the same scenario that we used as a running example in Chapter 3. An agent, Alice, is writing a book and is

searching for a guest author to write the introduction. The agent asks for Bob's help, but Bob might evaluate authors differently from Alice. The agents need to align before Bob's advice is useful. In this experiment our aim is only to show that a regression algorithm learns a useful alignment. In Section 4.3 we address the question of whether alignment is useful and what alternative algorithms can be used. For the experimentation, we use the $\mathcal{L}_{Trust}$ and $\mathcal{L}_{Domain}$ from the example: $\mathcal{L}_{Domain}$ uses the ontology of Figure 3.3 and $\mathcal{L}_{Trust}$ contains a single predicate $trust(Target, Value)$ with $Target$ an agent and $Value \in [-5, 5] \cup \mathbb{Z}$.

In order to prepare the experimental scenario, we use the ontology of $\mathcal{L}_{Domain}$ to automatically generate articles written by authors in the system. The background information about conferences, journals and authors is assumed to be known by all evaluating agents in the system. By generating the interactions from the $\mathcal{L}_{Domain}$ ontology we are simplifying the problem discussed in Chapter 3. In the theory we work under the assumption that there is a correlation between the properties of interactions described in $\mathcal{L}_{Domain}$ and the subjective observations of the agents. This correlation is what we try to capture using a suitable translation that captures the properties of an interaction that are relevant to a trust evaluation. We recall Example 3.18, in which we realised that the agent's observation that an article is *original* could not be represented in $\mathcal{L}_{Domain}$, so we used the publication venue instead. In the experimentation we flip this around: we generate articles using $\mathcal{L}_{Domain}$and the agents' observations that these are readable and original are then deduced from properties in $\mathcal{L}_{Domain}$ (although each agent does this differently and has different internal names for these properties). That way, when the agents communicate in $\mathcal{L}_{Domain}$ we are guaranteed that the subjective observations of agents are correlated with some sentence in $\mathcal{L}_{Domain}$. We emphasise again that all agents use different properties and evaluate these differently, so they still have to learn the alignment.

We run the experiments with a finite set of randomly generated articles. We then apply hold-out validation, using 60% of these interactions for the alignment process and 40% to evaluate it [Snee, 1977].

4.2.3 Trust models

In the experiments, we initially use four different agents, each with its own trust model. All these models use the same general structure, given in Algorithm 3. It is easy to see that this is an instance of the abstract trust model of Algorithm 1 on page 17: lines 2–5 perform the **process_experiences** function, while the rest corresponds with **calctrust**. The **process_witness** function is simply the identity function, as we assume we have already translated witness information using the trust alignment method.

The models distinguish between direct trust and communicated trust. If the agent has observed any articles written by the author T, it uses direct trust. This depends on the **evaluate** and **aggregate** functions to calculate a trust evaluation. If no articles have been observed, then communicated trust is used: each communicated evaluation has an uncertainty associated with it, which is dependent on the alignment method used. The agent selects the single communication with

Algorithm 3: Abstract Trust Model

Input: $t \in Authors$, the target author to be evaluated
Input: $Articles$, a set of articles, written by t
Input: $Communicated\_Evaluations$, a set of communicated evaluations from other evaluator agents in the system
Input: $default\_eval$, a default trust evaluation, used in case no articles have been observed and no communicated evaluations have been received

```
if Articles ≠ ∅ then
    article_ratings := ∅
    foreach Article a ∈ Articles do
        article_ratings := article_ratings ∪ evaluate(t, a)
    end
    trust_eval := aggregate(article_ratings)
end
else if Communicated_Evaluations ≠ ∅ then
    certainty := 0
    foreach Evaluation e ∈ Communicated_Evaluations do
        if certainty(e) ≥ certainty then
            certainty := certainty(e)
            trust_eval := value(e)
        end
    end
end
else
    trust_eval := default_eval
end
```

Output: trust$(t, trust_eval)$

the highest certainty to use. If there are also no communicated evaluations available, then a default trust evaluation is used. This is a very basic trust model and most models in the literature use a more sophisticated method of aggregating information from different sources (e.g. direct trust, reputation, communicated evaluations). Nevertheless, this model is sufficient to show the problems that arise if the agents do not align and to evaluate the different alignment methods. Sophisticated aggregation methods have large advantages at the individual level, because they allow for richer models and more predictive methods; however, if two agents use different aggregation methods, it is hard to distinguish whether the difference in trust evaluation is because the agents use a different aggregation method, or because they use different aspects of the interactions.

Work has been done on learning aggregated values [Uwents and Blockeel, 2008]. Unfortunately, this work is not yet applicable to the more complicated aggregation methods used in modern trust models. We avoid this issue by aligning the ratings of individual interactions. The agents can then use their *own* aggreg-

ation method, thereby obviating the need to solve the more complex problem of finding an alignment after aggregation. For the **aggregate** we take the average of the article ratings, although as we just explained, an agent could equally well use a probabilistic method such as BRS [Jøsang and Ismail, 2002] or a more social network orientated approach, such as Yu and Singh's model [2002].

The **evaluate** function is where each of the reader's trust models differs. As described in the previous section, each agent uses the objective description of an interaction in $\mathcal{L}_{Domain}$ to calculate some subjective properties of an interaction, such as readability or originality. Based on these, the agent calculates the rating of the author, using a list of "if-then-else" rules in Prolog, such as the following:

```
evaluation(Target, Article, 5) :- authors(Article, Authors), member(Target, Agents),
                significance(Article, Sig), Sig > 0.7, originality(Article, Ori),
                Ori > 0.7, readability(Article, Read), Read > 0.7, !.
```

This rule states that if the target agent is an author of the article and the observations of significance, originality and readability are all greater than 0.7 then the evaluation of the author, based on that article has value 5. All five of the readers' trust models are comprised of such rules, but they only coincide in the structure. The trust models differ in the actual content of the rules, such as the values attributed to different combinations of the subjective properties. Furthermore, the way in which the subjective properties, such as readability, are calculated, is different.

We call the two different parts of the **evaluate** function, the "observation"-module and "trust"-module. These perform the different functions in the theory: the "observation"-module is the implementation of the *observe* function (see Definition 3.1) and calculates the values for the subjective properties, such as readability or originality. The "trust"-module is the implementation of the trust classification (see Definition 3.3) using the Prolog rules above. Note that this implementation limits the sets of interactions whose observations can support trust evaluations to singleton sets for the reasons described above.

We combine the "observation"- and "trust"-modules to have a couple of different models for both Alice, the agent requesting a trust evaluation, and Bob, the witness supplying the trust evaluation. We can use this for two different things: the first is to verify that regression actually learns a useful alignment and secondly, to see if we can find a useful predictor of whether alignment will succeed, based on the aligning agents' prior knowledge of their trust models.

4.2.4 Estimating the difficulty of alignment

For Trust Alignment using regression to have any hope of succeeding, it requires that the agents communicate about many shared interactions, and additionally, execute a computationally intensive learning algorithm. It would, therefore, be nice if two agents could predict whether alignment can be expected to succeed for their different trust models, and additionally, how many shared interactions are necessary for a successful alignment. Such a predictor is what we describe

$$\begin{pmatrix}
0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 17 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
11 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 10 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 15 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0
\end{pmatrix}$$

(a) A simplest case for two trust models with 11 discrete values

$$\begin{pmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{pmatrix}$$

(b) and a hardest case

Figure 4.2: Matrices for possible combinations of trust values for the least and most complex case, with on the rows the own trust evaluation and columns the other's.

in this section and part of the aim of the experimentation in Section 4.2.5 is to verify that the predictor works as intended.

If two agents use the same trust models, then we should expect the alignment to be very simple and the communicated trust evaluations can be used immediately; however, if the models are different, the agents' trust evaluations based on the same interactions may be different: in this case alignment is necessary. How hard this alignment task is depends for a large part on how different these trust models are.

Theoretically, the simplest case is if there is a function, which completely independent of the $\mathcal{L}_{Domain}$ part of the SRAs, can translate the other's communicated trust evaluation into the own trust evaluation. In this case, the alignment is simply to apply this function, which is an easy task to learn: we do not have to take the information about the interactions into account.

Similarly, the hardest case is if any value the other agent communicates, can result in any value for our own trust evaluation: in this case the pairs of trust evaluations in the SRAs can be seen as a uniform distribution over all the trust values. In this case the other's trust evaluation gives no real information at all and the only useful part of an alignment message is the domain information.

Example 4.1. In our trust models each agent evaluates the target with an integer value between -5 and 5. This gives us an 11×11 matrix for the possible combinations of trust values. The simplest case corresponds to any matrix in which there is exactly one non-zero value in each row and column, such as the matrix in Figure 4.2a. In this case there is a one-to-one correspondence between the trust models without even considering $\mathcal{L}_{Domain}$.

The hardest case, however, corresponds to maximum entropy, given by, for instance, the matrix in Figure 4.2b. Given a matrix like this, we can say nothing about the own trust evaluation when given the other's: the probability distributions are completely uniform, and we must hope the $\mathcal{L}_{Domain}$ information allows us to learn an alignment.

While these two cases are theoretical boundaries for the complexity of the problem, they do give us a way of measuring how hard we expect it to be to align two trust models. We can measure the entropy in the distribution defined by counting how often an own trust evaluation coincides with any of the other's trust evaluations. The nearer this entropy is to 0, the more the problem looks like the simple case (for instance, the matrix in Figure 4.2a has an entropy of 0 in each row and column). The nearer this is to the maximum entropy distribution, the more the problem looks like the hardest case (the matrix in Figure 4.2b has a maximal entropy in each row and column). This complexity measure is inspired by Shannon's use of entropy [1948], in which the higher the entropy the more information a message contains. It reflects the cases that Nardin et al. [2009] found when attempting to align ontologies for different trust models — they found that it is often not possible to match a single concept from one ontology to a single concept from another ontology. We feel the problem goes deeper: not just is it impossible to always match concepts, but it is impossible to match concept-value pairs, which we call trust evaluations. The complexity measure gives an estimate of how hard the problem of finding a matching is expected to be.

We define the complexity measure for the case in which trust evaluations have a finite number of discrete trust evaluations (for instance, in our case the integers between -5 and 5).

Definition 4.1 (Problem complexity measure)**.** Given a set $\mathcal{E}$ of SRAs, we construct a matrix M, with in the rows the possible values of the own trust evaluations and in the columns the possible values of the other's trust evaluations. Each element in the matrix is the count of that specific combination of trust evaluations in $\mathcal{E}$. Let Z be a set with all the different possible values for the trust evaluation, with size z. M will then be a grid of size $z \times z$ and we use this to calculate the *complexity* as follows:

$$complexity(\mathcal{E}) = \frac{\sum_{i \in Z} \left(\left(\sum_{j \in Z} m_{ij} \right) \cdot entropy(m_{i*}) \right) + \sum_{i \in Z} \left(\left(\sum_{j \in Z} m_{ji} \right) \cdot entropy(m_{*i}) \right)}{2 \cdot |\mathcal{E}| \cdot max\_entropy(z)}$$

Where m_{ij} is the value of the element of matrix M with row index i and column index j, m_{i*} is the ith row of matrix M and m_{*j} the jth column. $max\_entropy(z)$ is the entropy of the uniform distribution with z values, also known as the maximum entropy distribution. *entropy* is the Shannon entropy function [1948]. Let $X = \{x_1, \ldots, x_n\}$ be a discrete random variable with n possible values, then $entropy(X) = -\sum_{i=1}^{n} prob(x_i) \cdot log(prob(x_i))$ and for any $x_i \in X$: $prob(x_i) = \frac{x_i}{\sum_{j=1}^{n} x_j}$

The first summation in the numerator sums the entropy in the rows, multiplied by a weight: the number of times the other's evaluations correspond to the own evaluation identified by the row. The entropy is a measure of the uncertainty about what the other's trust evaluation is, given that the own trust

evaluation is fixed. The second summation in the numerator does the reverse, it calculates a weighted sum of the entropy of the columns. This corresponds to the uncertainty about the own trust evaluation if the other's evaluation is fixed. The reason for adding the two is because we want the measure to be symmetric. Finally we normalise the numerator, to obtain a value between 0 and 1, where 0 corresponds with the least possible complexity between two models and 1 with the hardest: it is the maximum entropy distribution. One of the aims of our experimentation is to show that this measure can be used to estimate how hard a problem of alignment will be. We hypothesise that a complex problem requires more SRAs to learn an alignment.

Example 4.2. We end this section with an example explanation of the complexity calculation for a matrix between trust models E and F below, using 1000 interactions:

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 172 & 274 & 299 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 42 & 23 & 259 & 155 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 185 & 223 & 157 & 8 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 14 & 87 & 18 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 212 & 112 & 66 & 66 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

The complexity given by this matrix is $\frac{2628+2883}{2 \cdot 2372 \cdot 2.39} = 0.48$. We go through the calculation a bit slower. First we calculate the numerator and start with the first row. Because this is a zero vector. Technically this should result in the maximum entropy, but the weight is zero in any case, so the entropy calculation does not matter. The same for the second and third row. The fourth row is more interesting. When the own trust model evaluates an agent with value -2, the other agent evaluates sometimes with value 1, sometimes with 2 and sometimes with 3. We thus have three non-zero values in this row. In total the own model has evaluated an agent with -2 745 times in our set of SRAs. We interpret this row as a probability distribution over the other's trust evaluations for the conditional probability $P(Other\_Value = X | Own\_Value = -2)$. This probability is clearly 0 for all other trust values except $X = 1$, $X = 2$ and $X = 3$, for which it is $\frac{172}{745}$, $\frac{274}{745}$ and $\frac{299}{745}$, respectively. We therefore see that this row's contribution to the numerator is: $-172 \cdot \frac{172}{745} \cdot log(\frac{172}{745}) - 274 \cdot \frac{274}{745} \cdot log(\frac{274}{745}) - 299 \cdot \frac{299}{745} \cdot log(\frac{299}{745}) = 799$. We do the same for rows six, seven, eight and ten, and sum these together for a total of 2628. This process is repeated for the columns, obtaining 2883. This is normalised by dividing by $2 \cdot |\mathcal{E}| \cdot max\_entropy = 2 \cdot 2372 \cdot 2.39$ and the final result for the complexity between the two trust models is 0.48.

4.2.5 Experiment: evaluating First-Order Regression

As stated in Section 4.2.1, we choose TILDE [Blockeel et al., 2002] to perform First Order Regression. The result of the algorithm is an alignment and we

need to evaluate whether this alignment is useful or not. We perform hold-out validation and we can use our test set of 40% of the interactions to calculate the correlation between the outputs of the alignment from an alignment message and the trust evaluations as calculated by the own trust model. The higher this correlation, the better the alignment.

This measure, however, doesn't give an indication of how much the alignment improves the utility of communicated information. For this, we need another measure. We can use the Mean Square Error (MSE)[1] of the trust alignment, rather than the correlation. The advantage is that this value can give an indication of how much of an improvement the alignment is, by comparing it with the MSE of two "null" methods of alignment:

- do no alignment and simply use the other's communicated trust evaluation
- always take the mean of the own trust evaluations

The goal of this experiment is to show, firstly that First-Order Regression learns a good alignment, and secondly that the more different two trust models are, the more interactions are required for alignment. This seems like a straightforward experiment, but in actual fact it does three things:

- validates First-Order Regression as a way of learning an alignment
- validates the complexity measure of Definition 4.1
- gives practical knowledge of how many interactions are needed to learn an alignment

We use four different trust models of the kind we described in Section 4.2.3, calculate complexity of the alignment problem between them and learn an alignment using 10, 25, 50, 100, 200, 500 and 1000 shared interactions. Models A and B use both a different "observation"-module and a different "trust"-module from each other, while model C is designed as an intermediate model, using A's "observation"- and B's "trust"-module. Model D is designed to be an example of the simplest case when aligning with model A: it uses the same "observation"- and "trust"-modules as A, but we replaced the values of the trust evaluations in the "trust"-module, thus in any situation model A has evaluation trust(agent, 5), for instance, model D has trust(agent, -1).

Each experiment is run 50 times to get a statistically significant estimate of how good the alignment is, and the results can be found in Table 4.1. Each value is the statistical mean over the 50 experiments with the corresponding settings, and we also give the standard error for each set of 50 experiments.

[1] Given a set of authors A and for each, the set of articles they wrote, interactions I_a, the MSE is calculated as : $MSE = \frac{\sum_{a \in A} \sum_{i \in I_a} (estimate(a,i) - trust(a,i))^2}{\sum_{a \in A} |I_a|}$, where $estimate(a, i)$ is the estimated trust evaluation of the target author a, based on interaction i, using the alignment, and $trust(a, i)$ is the agent's true trust evaluation of a, based on i.

Trust models	Num. Inters	Complexity	Corr. training	Corr. control	MSE control	MSE null other	MSE null avg
A-B	10	0.13±0.06	0.83±0.25	0.73±0.29	1.38±1.36	2.96±0.61	2.37±0.9
A-B	25	0.15±0.04	0.88±0.16	0.57±0.27	1.57±0.83	3.23±0.4	2.76±0.85
A-B	50	0.16±0.03	0.82±0.17	0.60±0.27	1.82±0.97	3.21±0.38	2.89±0.72
A-B	100	0.17±0.03	0.87±0.09	0.72±0.18	1.34±0.76	3.18±0.3	2.87±0.64
A-B	200	0.17±0.03	0.88±0.11	0.79±0.17	0.96±0.74	3.12±0.22	2.68±0.46
A-B	500	0.17±0.02	0.98±0.01	0.96±0.02	0.24±0.11	3.13±0.2	2.84±0.46
A-B	1000	0.17±0.02	0.98±0.01	0.96±0.01	0.19±0.07	3.10±0.17	2.75±0.47
A-C	10	0.04±0.04	0.85±0.25	0.76±0.44	0.81±1.08	1.29±0.73	2.37±0.9
A-C	25	0.05±0.03	0.98±0.02	0.97±0.03	0.27±0.31	1.32±0.6	2.76±0.85
A-C	50	0.06±0.03	0.98±0.02	0.96±0.03	0.32±0.27	1.34±0.4	2.89±0.72
A-C	100	0.07±0.03	0.98±0.01	0.96±0.03	0.26±0.2	1.32±0.41	2.87±0.64
A-C	200	0.06±0.02	0.98±0.01	0.97±0.01	0.18±0.1	1.23±0.28	2.68±0.46
A-C	500	0.07±0.02	0.98±0.01	0.96±0.02	0.23±0.12	1.29±0.28	2.84±0.46
A-C	1000	0.07±0.02	0.98±0.01	0.97±0.01	0.19±0.11	1.27±0.27	2.75±0.47
A-D	10	0±0	0.91±0.21	0.87±0.34	0.67±1	3.00±0.64	2.37±0.9
A-D	25	0±0	0.99±0.02	0.99±0.02	0.06±0.14	3.07±0.31	2.76±0.85
A-D	50	0±0	1±0	0.99±0.02	0.09±0.13	3.01±0.34	2.89±0.72
A-D	100	0±0	1±0	0.99±0.01	0.04±0.07	2.99±0.29	2.87±0.64
A-D	200	0±0	1±0	1±0	0.01±0.02	3.03±0.22	2.68±0.46
A-D	500	0±0	1±0	1±0	0±0	2.99±0.21	2.84±0.46
A-D	1000	0±0	1±0	1±0	0±0	3.00±0.21	2.75±0.47

Table 4.1: Results with different trust models

Results: number of interactions

The first thing to note is that even at 10 interactions, which is often too few to give a good alignment of even simple trust models (A and C), using the alignment still performs significantly better than either of the baseline methods. There is a risk involved, though, because due to the very low number of samples, there are a couple of experiments which gave a higher error than using either of the baselines. In these cases the regression algorithm learnt to overfit on a bad set of SRAs, leading to a high error on the test set. This happened at 10 interactions for alignments between all models, even the simplest case, and at 25 for the hardest setting we tested (models A and B). On the positive side, these negative outliers can be detected: the correlation between the learnt values and the actual values in the training set is a good indicator of whether the alignment is good or not. Simply by disregarding all results with a correlation on the training set of less than 0.8 we throw out all negative outliers. An agent can calculate this correlation after the alignment process, and if it finds such a low value it can conclude the alignment failed. However, even with such outliers, on average the alignment performs far better than either baseline method for all trust models and all numbers of interactions, as Table 4.1 indicates.

10 interactions is also too few to accurately calculate the complexity. In fact, at low numbers of interactions, the estimate of the complexity is lower than the actual complexity. This is expected and we can see this by revisiting the

definition of the complexity measure. Let us consider, for instance, the case where we only have a single SRA. In this case the entropy will be 0 regardless of the trust values and therefore the complexity will also be 0. If we have two SRAs, then, given completely random trust evaluations, there is a $\frac{20}{121}$ probability that the two SRAs are different, but either in the same row, or the same column[2], resulting in a complexity of 0.07 and a probability of $\frac{81}{121}$ that the complexity is 0. However, the more samples we have, the higher we can expect the complexity to get until it converges on the true complexity. Luckily, our method seems to converge fairly quickly and at just 25 interactions it already gives a fairly accurate estimate of the complexity, which is a lot less interactions than are required to learn an alignment between most models.

We see that the learning algorithm starts working better with more than 50 interactions, although to always find the 100% correct alignment of the simplest models (aligning A and D) we still require 500 interactions. Note that we are using 40% of these to test the alignment, so the learning is done with 300 interactions. We also see that at 50 interactions the algorithm learns a perfect alignment on the training set. Nevertheless, there are cases that simply have not arisen in the training data, and thus correlation drops on the test data. Luckily, also in the test set such situations only arise rarely, so at 50 interactions the alignment is already very functional and gives far better results than using either of the baseline methods.

Aligning models A and B requires the largest number of interactions, but with 200 interactions the MSE drops below 1.

Results: complexity

Note that our most complex alignment problem still only has a complexity of 0.17±0.02. To properly verify that complexity is a good measure of how many interactions will be required for alignment, we designed another two trust models, with a higher complexity for aligning. Model E has model A's observation module, but has a more complicated trust module. It first checks whether the target for evaluation is the first author of an article, or a different author, and evaluates the first authors with more exaggerated values than the other authors (in other words, if the article is evaluated badly then the first author gets a very bad evaluation, while other authors have an evaluation nearer to 0). Model F uses model B's observation module and its trust module makes a similar distinction to model E's, but rather than distinguishing between the first and other authors, it distinguishes between articles published in journals or conferences. We run the same experiment with these models, and the results can be found in Table 4.2. We have a complexity of 0.48±0.02 and can still learn quite a good alignment (with an MSE under 1) in 500 interactions. It does not quite

[2]Given an SRA with own evaluation a and other's evaluation b, if we take another SRA with their values in the random distribution, there is a $\frac{1}{11}$ chance that the own evaluation is equal to a. There is also a $\frac{10}{11}$ chance that the other's evaluation is unequal to b: there is a $\frac{10}{121}$ chance of having the two SRAs end up in the same column, but in different rows. The same calculation holds for the same row, but different columns, for a total probability of $\frac{20}{121}$.

Trust models	Num. Inters	Complexity	Corr. training	Corr. control	MSE control	MSE null other	MSE null avg
E-F	10	0.31±0.06	0.86±0.24	0.59±0.38	2.03±2.54	4.78±1.98	±1.22
E-F	25	0.40±0.04	0.93±0.07	0.57±0.31	2.19±2.24	4.95±1.99	2.60±1.56
E-F	50	0.44±0.04	0.92±0.07	0.62±0.23	1.90±1.19	5.12±1.23	2.77±1.14
E-F	100	0.46±0.03	0.93±0.05	0.71±0.17	1.41±0.75	5.19±1.21	2.76±1
E-F	200	0.47±0.02	0.93±0.06	0.76±0.18	1.03±0.8	4.75±0.73	2.40±0.67
E-F	500	0.48±0.02	0.98±0.01	0.93±0.02	0.36±0.15	5.01±0.72	2.63±0.69
E-F	1000	0.48±0.02	0.98±0.01	0.95±0.02	0.27±0.13	4.99±0.7	2.48±0.67

Table 4.2: Results for more complex trust models

achieve the results of the alignment between A and B, but it is very near. This is remarkable, given the other results and will require future testing. A possible explanation is that there is a complexity "threshold" in the scenario we have designed: above a certain level of complexity, the learning algorithm must take all information in $\mathcal{L}_{Domain}$ into account. In this case it is to be expected that the number of interactions required to learn the alignment will not vary. Above this threshold we need a fairly large sample of interactions covering all situations, regardless of the complexity of the problem. To test this properly, experimentation is required in a different scenario with a different $\mathcal{L}_{Domain}$, which is outside the scope of this work.

4.2.6 Discussion

The experiment demonstrates the functioning of the alignment and evaluates the measure we propose for estimating the complexity of the problem. We can draw two conclusions from this experiment:

- If agents can accurately communicate using a numerical value for their trust and represent the underlying interactions used in $\mathcal{L}_{Domain}$, then the regression algorithm can learn a trust alignment. Even with few interactions, there is less error in the communication than the best we can achieve if we do not align. While our trust models use discrete values for trust, the regression algorithm used does not require this, and any numerical representation of trust could be used. Furthermore, the different representation methods for trust can be converted into a numerical value [Pinyol et al., 2007], although in such cases other learning methods may perform better.

- The complexity measure in Definition 4.1 is a measure of how many interactions are needed to achieve the quality of alignment required. The number of interactions required for a given complexity is also dependent on the environment, namely on the expressivity of $\mathcal{L}_{Domain}$ and the language bias in the learning algorithm. Normally these are fairly static in an environment and the system designer could make a lookup table available to the agents, giving the expected quality of the alignment at a certain complexity level using a specific number of interactions.

A limiting factor, which we have already discussed in Section 4.2.3, is that our approach only considers the evaluation based on a single interaction at a time, while the majority of research into trust models focuses on methods for aggregating the evidence from a large number of interactions. Uwents and Blockeel [2008] present a model for learning aggregated values, but even if this were capable of dealing with the complex aggregation methods of contemporary trust models, there is a further complication. The problem would not be to learn an aggregated value, using different aggregation methods, from the same data, but rather to learn the relation between two aggregated trust evaluations from underlying evidence (or interactions). Firstly the single values derived from this evidence are different, because each agent takes different criteria into account and secondly the aggregation technique used is different. This presents too many unknown variables to be learnt at once. The method we present can therefore either be seen as a necessary first step for the solution of this problem, or as an alternative solution to the problem in the situations that communication about evaluations of single interactions is possible.

4.2.7 Summary

In this section we discussed some of the practical issues we encounter when implementing a method for performing trust alignment. Specifically we dealt with the large search space of θ-subsumption, the relative complexity between trust models and the problem of learning aggregated values. We described our use of TILDE, a state-of-the-art ILP algorithm, to address these issues and provided evidence that this learns a trust alignment that can translate new incoming messages in a useful manner.

However, we have not tested how useful this alignment is, nor whether it is worthwhile having to address the problems inherent in using a machine learning approach that takes the context into account. In the next section we will compare our method with some of the other solutions out there and show that, firstly, trust alignment is a real problem and it must necessarily be solved in some way or another, and secondly, that the method we propose is a useful way of addressing it.

4.3 Comparing Trust Alignment Methods

So far in this chapter we have presented our own method for trust alignment, using a First-Order Regression algorithm. However, we have not given empirical evidence that trust alignment is a problem that needs solving. We hypothesised that the problem of trust being subjective hampers the communication thereof, and thus the accuracy of trust models, in any situation where an agent can only obtain information about the trustworthiness of potential partners by communicating. Nevertheless, neither we, nor anybody else, have provided empirical evidence supporting this hypothesis. In this section we aim to do two things: firstly, we show that some form of trust alignment is a necessity for communic-

ating about trust; and secondly, we compare our method to a number of other solutions to the problem.

This section is organised as follows: first we give an overview of the experimental setup we use. In Section 4.3.2 we briefly recap the alignment methods we described in Section 2.3 and describe the implementation of the ones we compare here. Sections 4.3.3 and 4.3.4 detail the empirical evaluation.

4.3.1 Experimental setup

The aim of the experiments is to measure the effect of communication about trust on the accuracy of agents' trust evaluations. We are explicitly not interested in evaluating trust models and whether they choose the correct target. For this there are other methods, such as the ART testbed [Fullam et al., 2006]. To measure the effect of communication we need to compare two situations: (1) an agent's *estimated* trust evaluations, initially given incomplete information about the environment, but allowing communication about trust; and (2) that same agent's *most accurate* trust evaluations, given perfect and complete information about an environment. This allows for the comparison between the two evaluations and gives a measure for the *accuracy* of the estimated trust evaluation. By varying the amount of communication allowed and the type of alignment used we can measure the influence that alignment has upon the accuracy of the agents' trust evaluations.

The scenario is, once again, based on the running example in Chapter 3 and is largely the same as described in the previous section. However, rather than just Alice needing to find a guest author, we evaluate multiple "reader agents" simultaneously: they have to recommend authors to each other, basing these recommendations on the articles they have written. These articles are generated in the same manner as described in Section 4.2.2: we generate synthetic articles written by between one and five authors each, using the description of the article in $\mathcal{L}_{Domain}$ as given in Figure 3.3 (see page 49). Each of these articles represents a shared interaction, upon which a reader can base a trust evaluation about its authors.

In an initialisation phase, the articles are divided over the reader agents, such that each reader only receives articles written by a configurable percentage of the author agents. The goal is to give each reader only partial information, so that each of them has incomplete information about only some of the authors in the system, thus creating the need for communication. For this communication we use the same $\mathcal{L}_{Trust}$ and $\mathcal{L}_{Domain}$ languages as in Chapter 3 and Section 4.2.

After the initialisation the experiment runs for n rounds, in which each round represents the opportunity for the readers to communicate. In each round the agents may pick one other reader agent to communicate with. A communication act may be: a request to either align, or to get the other's trust evaluation of a single author. After n rounds of communication a measure of each agent's individual accuracy is calculated, and averaging these individual measures, the score of the entire run is determined. This score can then be compared to runs with a different value for n or using different methods of alignment.

The reader agents have the same trust models we used in the previous section, described in detail in Section 4.2.3; however, this experiment no longer compares them pairwise, but all together. We thus have 5 different reader agents, using models A, B, C, E and F (we omit D, for obvious reasons). We realise that there are big differences in complexity between some of these models; as we described in the previous section, the complexity between models influences the results of the alignment. We feel that evaluating all these models at once is an interesting reflection of reality, in which we may very well encounter diverse trust models. Nevertheless, as discussed in Section 4.2.6, in real encounters, agents might want to exchange some trust evaluations first to discover the relative complexity of their trust models, and given a choice, only align in those situations where the complexity is low. In this experiment we do not give the agents such an option and all reader agents simply align with all other agents, thereby giving an average accuracy of the alignment methods over a range of different complexities.

Strategy

In addition to the trust model, each agent must have a strategy to choose what to do in each round. While we cannot focus too much on this in the scope of this thesis, we realise that this choice may have a large influence on the outcome of the experiment. We therefore implement two strategies for comparison. The first is a simple random strategy. Each agent chooses an author at random. It then chooses a reader agent at random to ask about that author. If it has not previously aligned with that reader, rather than asking for the agent's evaluation, it asks to align. If it has already aligned, it asks for the other agent's evaluation of the chosen author.

The second strategy is a non-random strategy in which each agent first chooses the author it has the least certain evaluation of. We use a very simple notion of certainty: an agent's certainty is equal to the percentage of the author's articles that the agent has observed. This notion may not be particularly accurate (for instance, if the author has written only very few articles[3]), but it is only a heuristic for selecting which author to obtain more information about. It does not affect the trust evaluation. After choosing the target author, it picks the reader agent that has the most observations of that target and whose opinion has not yet been asked. After choosing the author and evaluator agent, this strategy behaves the same as the random strategy: if the agent has already aligned with the chosen evaluator it asks for a trust evaluation and otherwise it asks to align. While there are many optimisations possible, they are also further distractions from the main tenet of this research. We do not doubt that there are ways of improving the strategy of choosing when to align or with whom to communicate; however, the main idea is that if we can show that the trust evaluations are more accurate with alignment than without, performance should only improve if the strategy is optimised.

[3]Because we generate articles artificially, we can guarantee that this does not occur in our simulation.

4.3.2 Alignment methods

Before discussing the experiments in detail we need to introduce the trust alignment methods we compare. In Section 2.3 we discussed the state-of-the-art methods that can be considered as methods for trust alignment, whether they were intended that way or not. We distinguished between roughly three different types: filtering mechanisms, translation mechanisms using just the trust evaluations and translation mechanisms that take the context into account. In the first experiment we evaluate some of these methods.

Filtering

We test one filtering method, which we base on POYRAZ [Şensoy et al., 2009]. POYRAZ uses something they call "private credit" to evaluate whether or not an adviser is offering useful advice. They do this by finding past experiences that were similar for both agents. Similarity in their model is based on time: they assume that a target behaves similar if the time of two interactions is sufficiently close together. An experience, in POYRAZ, is a description of both the promised and the provided service, using a domain language. This language is shared, so the sender communicates its experiences rather than its trust evaluations, and they use the similarity in time to specify when one of these experiences is similar to one of the receiving agent's own experiences (or if we consider it in our framework, when two experiences can be considered as the same token). The method then calculates the receiving agent's satisfaction (that we have called a trust evaluation) of *both* experiences: its own and the received, similar, experience. If these are the same (either the receiver would be satisfied in both situations, or the receiver would be dissatisfied), then the pair is classified as a positive example and otherwise as a negative example. They then use the beta probability distribution to calculate the probability that, given the received past experiences, a future communication from the same will be useful. Working out their formula is equal to performing Maximum Likelihood Estimation (MLE) with a Laplacian smoothing factor of 1[4].

As argued in Section 2.3, there are a number of reasons for not being able to communicate the entire experience: firstly there may be subjective concepts that cannot be communicated (such as originality of an article in our example) and secondly agents might have private reasons for not wanting to communicate an experience fully. We therefore relax the communication condition of communicating an entire experience, but, instead, work with our own concept of shared interactions. We use the SRAs and a distance measure: if, given an SRA, the distance between the own and other's trust evaluation is under a threshold, we classify the SRA as positive and otherwise as negative. We then continue on in the same manner as POYRAZ does, by using maximum likelihood estimation. If the probability of the other agent's information being useful is above a certain

[4]The formula is given as follows: let N_{all} be the total number of samples and N_+ the number that were positive, then $MLE_{k=1}(N) = \frac{N_+ + 1}{N_{all} + 2}$, where $k = 1$ stands for the Laplacian smoothing factor [Russel and Norvig, 2010].

threshold we accept communications from it. POYRAZ uses a threshold of 0.5, so we follow suit. The distance measure we use for trust evaluations is simply Euclidean distance and we set the threshold to 1 (in other words, two trust evaluations are similar if they are neighbouring integers). We call this *POYRAZ filtering method*.

In addition to using this filtering method we will use two baselines, which can be seen as the extreme ranges of the filtering method: the first is to filter out no communication at all, or simply to accept all recommendations from all agents (called *no alignment*). The second baseline is to filter out all communication, or simply put, not communicate at all (called *no communication*).

For both the standard filtering and especially for filtering out everything, there will be authors the agent knows nothing about. In this case we have to use a default trust evaluation, which can be seen as the agent's initial evaluation of any author, before learning anything about it. We identify four options for this default evaluation:

A mistrusting agent always gives its most negative evaluation to any agent it has no knowledge of.

A trusting agent always gives its most positive evaluation to any agent it has no knowledge of.

A neutral agent always gives a middle evaluation to any agent it has no knowledge of.

A reflective agent calculates the mean of all its previous trust evaluations of other agents and uses this for any agent it has no knowledge of.

The first three options give a fixed value, independent of the agent's evaluations of other targets in the system, whereas the last option allows the agent some type of adaptability, depending on what trust evaluations it has so far given to other targets. If the targets it has knowledge of are all bad agents, then it will be more similar to the first option, whereas if they are all good it will be more similar to the second. Of all options for no communication we expect this will be the best choice for an agent, although it is also the only option which requires extra computation. Any comparison of trust alignment methods will use the same default evaluation for all methods.

Numerical translations

In Section 2.3 we discussed Abdul-Rahman and Hailes' alignment method as the first method to deal with trust alignment. This method uses a numerical translation and in the experiment we compare two such methods.

Average Distance Bias Our first numerical translation method is a very simple method, which does not take the context into account. When aligning, it calculates the mean difference between the other's recommendations and the

own trust evaluations and use this as a single bias. We will call this method the alignment using an *average distance bias*.

Abdul-Rahman and Hailes' Method (AR&H) AR&H's method cannot be applied directly, because it requires discrete values to calculate the bias. In our models the trust evaluation is the average of an author's ratings and is therefore not a discrete value. We can, however, apply AR&H's alignment method at the level of the ratings of individual articles, which are discrete: specifically, in our experiment they are natural numbers between -5 and 5. The method applied is almost the same as that already described in Section 2.3. Instead of first-order regression it performs numerical regression and finds a function of the form $F(x) = x + b_x$, where b_x is a bias. The main difference between AR&H's method and the average distance bias is that in AR&H's method the bias is dependent on the other's trust evaluation: for each possible value a different bias is learnt. We thus end up with 11 different biases: we group the SRAs by the value of the other's trust evaluation and for each group calculate the average distance bias separately.

The only difference with the method originally proposed by Abdul-Rahman and Hailes [2000], is that we use the mean to calculate the bias, rather than the mode. We can do this, because we take the average of these values afterwards in any case, whereas Abdul-Rahman and Hailes needed to keep evaluations in their discrete categories and used the mode.

Machine learning using contextual information

The last two methods that we compare are ones that use the information in $\mathcal{L}_{Domain}$. The first uses clustering and classification to learn an alignment [Koster et al., 2010a] and the second is the one that we have presented in this thesis, using First-Order Regression to learn an alignment.

We do not compare our method to the other approach that uses a machine learning algorithm to learn an alignment, BLADE [Regan et al., 2006]. This approach uses a propositional Bayesian Inference Learner. Comparing a first-order approach and a propositional one is not straightforward, because of the difference in representation. Friedman et al. [1997] demonstrate empirically that propositional logic decision tree learners (which are propositional ILP algorithms) and Bayesian Inference Learners perform approximately equally, although ILP algorithms perform computationally better in large problems. Unfortunately BLADE is not equipped to deal with the more complex problem we consider here, in which a first-order — rather than a propositional — logic is used to describe articles. To learn relations in this language would require a different, first-order Bayesian network, which falls outside the scope of this thesis.

The implementation of the clustering-based approach uses a bottom-up incremental clustering algorithm to cluster based on the trust evaluations. For learning classifications of each cluster, we use Tilde [Blockeel et al., 2002], but we set it to learn a binary classification, rather than a regression tree. We call this method *Koster Clustering*.

The First-Order Regression method is the exact same one we described in Section 4.2.1 and we call it First-Order Regression Trust Alignment Method, or *FORTAM*.

4.3.3 Comparing alignment methods

The first experiment aims to compare the alignment methods with each other as well as with the two default modes: no communication at all and communication without alignment. As described above, if an agent has no knowledge of an author, it uses a default trust evaluation. Because the agents have incomplete information about the environment, this case will occur when no, or too little, communication is allowed.

Setting up the experiment.

We start by running a number of experiments to ascertain which parameters should be used for a fair comparison between the alignment models. By changing the total number of articles and the percentage of articles observed by each agent we can change the average number of articles shared by the agents. This mainly influences the functioning of AR&H's method and our own learners, Koster Clustering and FORTAM. At low numbers of shared articles AR&H's method outperforms FORTAM; however, with around 100 articles shared between any two agents, FORTAM starts to outperform AR&H's. This difference in performance increases until approximately 500 articles are shared, on average. Running the experiment at higher numbers of shared interactions is unnecessary, because all algorithms have reached peak performance. We opt to run our experiments with 500 shared articles, thus achieving the optimal results obtainable with each of the alignment methods. The goal of the experiment is to measure the influence the different alignment methods have on the accuracy of an agent's trust evaluations. Therefore we require each agent's information about the environment to be incomplete. We achieve this by only allowing each reader agent to observe articles by 40% of the author agents. This means that to find out about the other 60% of the authors, communication is required. By having a total of 1000 articles written by different combinations of 50 authors, we can measure the influence of communication while still allowing agents to, on average, share 500 articles. We run each experiment 50 times with different articles to have a decent statistical sample. In this first experiment we vary two parameters: the number of rounds in which agents may communicate and the baseline trust evaluation an agent uses to evaluate targets it has no information of. The results are plotted in Figure 4.3. The y-axis represents the error with respect to the most accurate evaluation: what would be the agent's evaluation if it were to have perfect information about all articles. Given the probability distribution of a trust model's evaluations, the error is the probability of the agent's evaluation of a target being between the estimated and most accurate evaluation[5]. It is a

[5]calculated as the cumulative probability between the two values

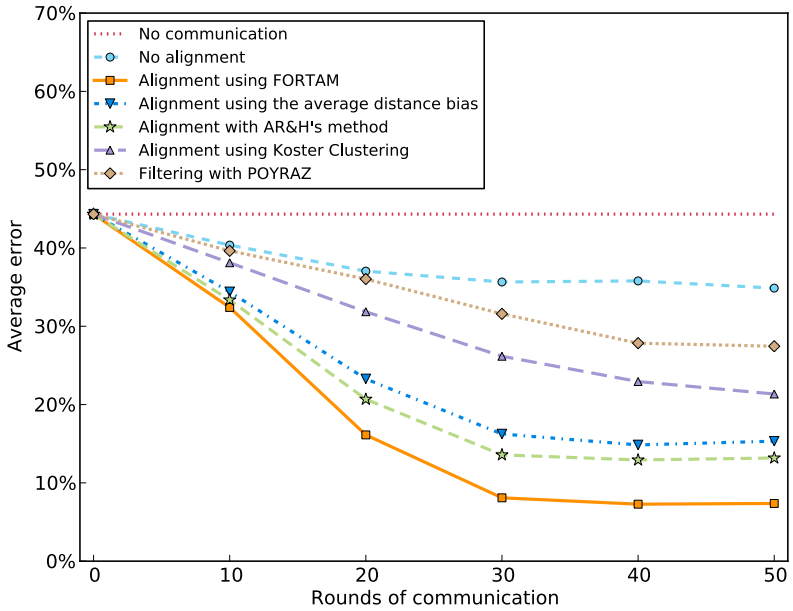
measure of the inaccuracy of the alignment method, because the percentage on the y-axis is not the chance that an agent's evaluation is wrong, but rather a measure of *how* wrong an agent is on average.

Results.

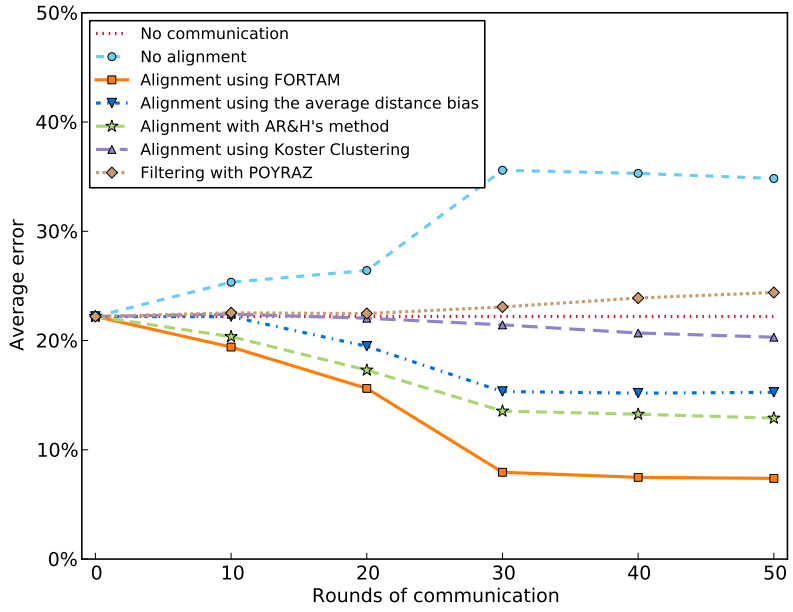
We firstly see in Figure 4.3a that if we use the neutral baseline (using 0 as the default evaluation), then all communication is preferable over no communication. The same is not true if we use the reflective baseline (taking the average of past evaluations of other targets), as seen in Figure 4.3b. In this case communication without alignment gives worse results than not communicating at all. This is easily explained: if the observed articles are a representative sample of the population then the mean of trust evaluations based on these will be near the mean of the most accurate trust evaluations. Consequently, always using the default will be quite good. On the other hand, the other evaluators' trust evaluations are based on different properties of the articles and may thus be further from the most accurate trust evaluation. The more of these unaligned communicated evaluations an agent incorporates, the less accurate its evaluations will become. We allocate articles at random and therefore each agent does observe a representative sample of them. This same would not be true if the network were not a random network or the location of an agent in the network influenced its trustworthiness: the targets observed would not be a representative sample of the other agents in the network and the error from using the default would be larger. If this error becomes large enough it would resemble the situation with the neutral baseline, in which case the error from using unaligned communications results in an improvement. We have omitted the experiments using the trusting and distrusting baselines, because their results are very similar to those of the experiment with the neutral baseline and thus add very little information.

Another interesting observation is that filtering out deceptive agents does not deal particularly well with subjective trust evaluations and its performance is worse than even for average distance bias, the simplest of the alignment methods. Specifically we see that one of two cases can occur when filtering. Either the witness' prior evaluations are sufficiently similar for its communications to be accepted, in which case they are used without translation, similar to the situation with communication but no alignment. Alternatively, the witness' prior evaluations are not sufficiently similar for its communications to be accepted, in which case they are discarded. Given these two possibilities we expect the filtering method to perform between "no communication" and "no alignment", which is exactly what we see: it stays near to the baseline, indicating that most of the time witnesses are deemed deceptive and their communications are ignored. In this experiment no agents are deliberately lying about their trust evaluation and this seems to indicate that a filtering method, such as POYRAZ, is not adequate for dealing with subjectivity in trust.

The main result of this experiment is that communication *with* alignment *always* gives significantly better results than either no communication or communication without alignment. In Figure 4.3b we have plotted the average accuracy



(a) Using the neutral baseline (always evaluating the target with value 0)



(b) Using the reflective baseline (taking the average of its previous evaluations of other targets)

Figure 4.3: Average accuracy of trust evaluations, using the different methods for processing witness information. When no information is available about the target, the evaluator uses the corresponding baseline.

for all five of the agents. However, as discussed in Section 4.2.3, the individual trust models play a large role in this performance. The different alignment methods give different returns for the individual agents, but always significantly outperform the situations without alignment. Furthermore the differences seen in the graphs are significant. Because the accuracy measure is not normally distributed we evaluated this by using a Kruskal-Wallis test for analysis of variance [Corder and Foreman, 2009]. The pair-wise difference is also significant, as tested using Mann-Whitney U-tests[6]. We included our initial approach to trust alignment, described by Koster et al. [2010a], but we see that it does not perform particularly well. Clustering based on the evaluator's trust evaluation groups together many of the witness' evaluations that are unrelated. It clusters SRAs together based on the agent's own evaluations, which, if it manages to learn a classification of the alignment messages, is accurate, but unfortunately fails to learn a classification in most cases. This is due to the relative complexity between the various trust models. By clustering SRAs together based on the own trust evaluation, a cluster usually contains SRAs with the other's trust evaluations being very different from each other. Moreover, the algorithm cannot even learn a classification using only the information in $\mathcal{L}_{Domain}$, because we have limited alignment messages to descriptions of the interaction that are relevant to the sender: similar descriptions in $\mathcal{L}_{Domain}$ are therefore similarly distributed over the various clusters as the trust evaluations. This, specifically, explains the difference in performance between our experiment and the results of Koster et al. [2010b]. In that experiment, the entire description of an interaction was communicated, using a similar approach to that of Şensoy et al. [2009]. While this method might be a good solution if trust evaluations are not numerical (and thus none of the other alignment methods can be used), it does not perform as well as even the simplest numerical method and we shall omit it from the rest of the experiments. FORTAM, on the other hand, seems to take advantage of the contextual information in a useful manner, allowing it to perform better than any of the other methods tested. The experiment, however, was designed to allow agents to take advantage of the information in $\mathcal{L}_{Domain}$. It remains to be seen whether real scenarios provide equally useful information about the interactions.

The first variation on this experiment we explore is to change the strategy for selecting a communication action. The first experiment uses the non-random strategy and we compare these results to the exact same experiment, but using the random strategy. For this experiment we use the reflective baseline and the results up to 300 rounds of communication are plotted in Figure 4.4. As is to be expected, we see that in the short term picking the communication at random does quite significantly worse than using a heuristic to choose whom to communicate with: after 50 rounds of using the non-random strategy (see Figure 4.3b) all alignment methods are doing significantly better than after 50 rounds of using the random strategy (Figure 4.4). Nevertheless, in the long run

[6] for all tests we obtain $p \ll 0.01$: the probability that the different data sets were obtained from the same population is very small

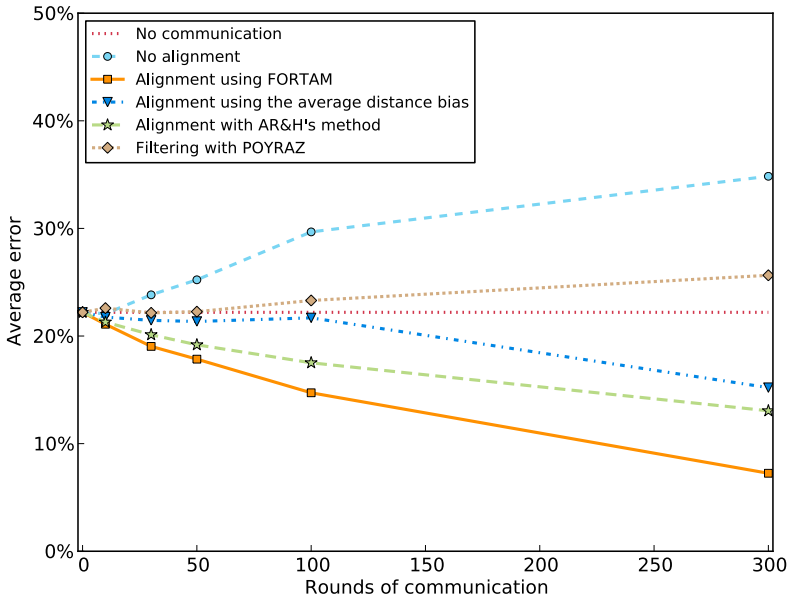


Figure 4.4: The random strategy for partner selection

the effect is flattened out and eventually the random strategy achieves the same optimum alignment as the non-random strategy. This implies that, after enough rounds of communication, the optimum is fixed by the alignment method and the strategy does not influence it. To show that the value they converge on really is the lowest average error an agent can achieve using the given alignment method, we run the non-random strategy for 150 rounds, which is enough rounds for all possible communications to take place. For all the methods tested we compare this with the outcome after 50 rounds for the non-random strategy and 300 rounds for the random strategy: the values are mutually indistinguishable[7], showing that even after exhausting all possible communication the alignment is not further improved and truly is an optimum.

The strategy, however, does have a strong influence on how fast this optimum is reached. Using a different strategy will change the speed of convergence.

This means that from an agent designer's viewpoint the strategy and alignment method can be completely separated: if an evaluator agent requires information about a target agent, the alignment method defines an optimal accuracy for this information while the strategy defines how many agents on average the evaluator agent must communicate with before it has communicated with the agent giving the most accurate information.

[7] obtaining $p \gg 0.05$ for all Mann-Whitney U-Tests

4.3.4 Simulating lying agents

In the first experiment we tacitly assumed all agents are truthful and willing to cooperate. If they do not cooperate with the alignment process there is obviously nothing we can do, but assuming other agents are truthful is a rather strong assumption. This experiment is therefore set up to see what happens with the accuracy of the communication if we simulate the other agents passing incorrect information. Note that if the agents are entirely consistent in their lies, the AR&H and FORTAM methods will be able to deal with this perfectly, as they learn a translation from the other's trust evaluation. Additionally, FORTAM can even deal with lying if it is not always consistent, but based on some specifics of the underlying article (such as: always lie if the author works at a certain institute). The problem for all alignment algorithms appears if agents just invent a random value. It is illuminating to compare the alignment methods with the filtering method in this situation, and we run another round of experiments, this time increasingly replacing truthful agents by lying ones. A lying agent, rather than giving an actual trust evaluation, communicates random ratings of articles. The results can be seen in Figure 4.5. The agents use the reflective baseline as their default evaluation in the case they do not have other information available.

Results

We focus first on graph (e) in Figure 4.5 and see that if all agents are lying then communication with no alignment converges to the neutral baseline. We can explain this convergence by seeing that the mean of all possible trust evaluations is also the mean value of a random distribution over the possible trust values. A similar thing happens using AR&H's method, which calculates what its own trust evaluation should be if the other agent communicates a certain value, but it converges on the reflective baseline. Because the other's trust evaluations are random, choosing all those at a certain value will give a random sample of the own trust evaluations, the mean of which will, on average, be the mean of all the own trust evaluations, so AR&H's model stays approximately flat on the default baseline (using the average of all the agent's own trust evaluations). For similar reasons the average bias does slightly worse, converging to a value between the two baselines. FORTAM, on the other hand, appears to hardly be affected by the noisy trust evaluations. This shows a large advantage of taking the context into account: FORTAM maintains its performance, because the communications in the domain language can be used for the alignment method to compensate for the noisy trust evaluations. It ignores the noisy trust evaluations and learns by using *only* the information about the underlying articles. If we were to add noise to this part of the communication as well, FORTAM would collapse to AR&H's and thus stay flat as well.

With this explanation of what happens when all agents lie we can see that by slowly adding more liars to the system, the performance of the various algorithms morphs from the system with no liars to the system with all liars (Figure 4.5(a)-(e) progressively). To prevent this from happening a further refinement would

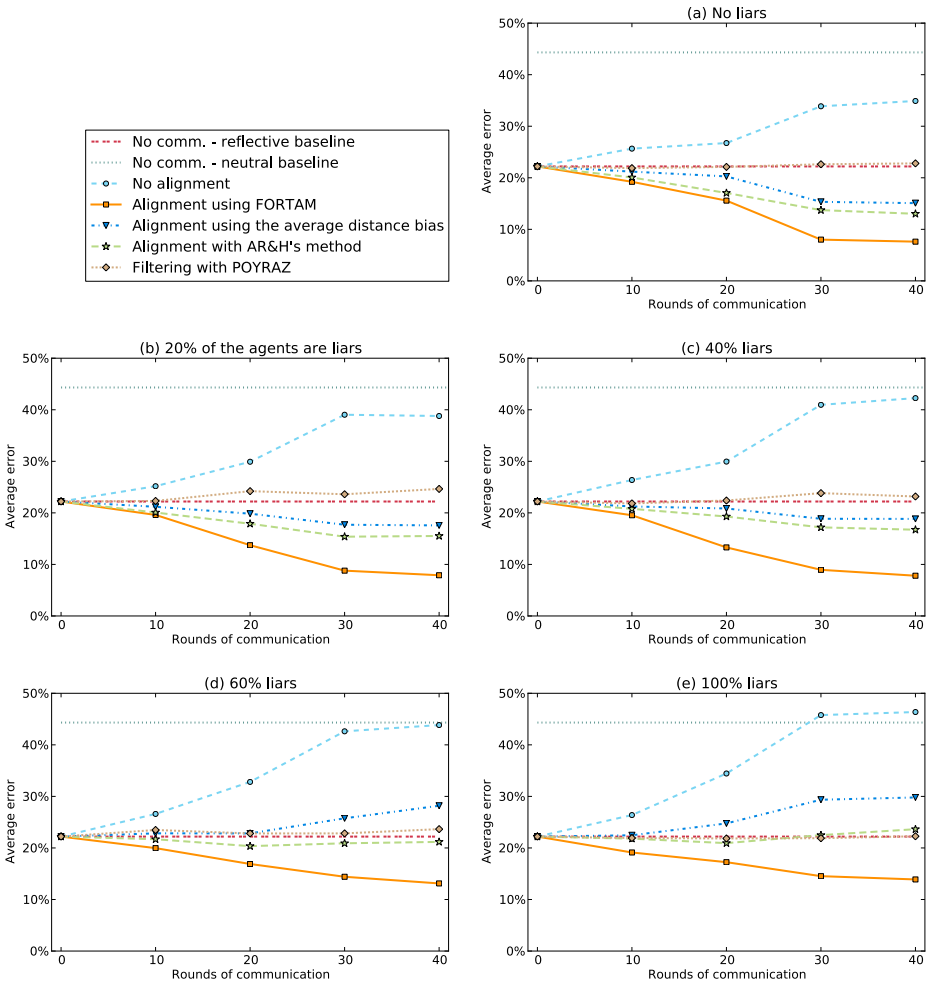


Figure 4.5: Slow degradation from a domain with no lying agents to a domain with all lying agents

be necessary: detecting which agents are the liars and disregarding their communications.

POYRAZ, as expected from a method designed to deal with deception, deals correctly with lying agents. It identifies the liars and filters out their communications. Because the liars give random evaluations this is an easy task and is done with 100% accuracy and thus, with 100% liars in the system (Figure 4.5(e)), it filters out all information and stays exactly flat, always using the baseline evaluation. However, FORTAM is, despite the agents lying, able to make good use of the contextual information to still obtain a fairly accurate alignment. Once again, if the agents were to deceive each other about the context as well, it would

not perform as well. In such a case it could be beneficial to use POYRAZ to filter out the liars, rather than try to learn an alignment with them.

4.3.5 Discussion

The experimentation in the previous section demonstrates that trust alignment improves the accuracy of agents' trust evaluations. FORTAM even works in situations where the communicated evaluations are 100% noise. All the same, we must take care when interpreting these experiments. The first thing to note is that the trust models used, as described in Section 4.2.3, are simplifications of those used in the literature. Agents only communicate the evaluations based on their own direct experiences, rather than having an evaluation which is aggregated from a number of different sources. This, however, only strengthens the point we are trying to make: the more complex an agent's trust evaluation can be, the greater the probability that two agents, despite using the same ontology for their trust evaluations, *mean* different things, because the actual way they calculate the evaluations are completely different. The use of more complex trust models thus leads to an even greater need for alignment. Unfortunately, the more complex the trust models, the more information will be required to apply a method such as FORTAM, which requires numerous samples of different types of evidence supporting the trust evaluations. Luckily, the worst case for FORTAM is that the domain information is too complex to use, in which case it will perform similarly to AR&H's method. In such cases there may be other machine learning techniques, such as Case-Based Reasoning [Aamodt and Plaza, 1994], which is designed to handle large sets of complex data, which could offer a solution.

Additionally, the alignment is required to take place before aggregation. This means that regardless of how complex the aggregation method is, as long as what is being aggregated is not too complex, the alignment can work. It also means, however, that a large amount of information needs to be communicated. There may be scenarios in which this communication is prohibitive and a simpler form of alignment, such as AR&H's method, or even the average bias, must be used. Luckily, in domains such as e-commerce, a lot of data is readily made available: on eBay[8] for example, for any transaction it is public knowledge what item was sold and how much it cost. Similarly in social recommender systems, which is how we would classify the example scenario in this chapter, people are often willing to explain their evaluation of an experience in great detail (such as on Tripadvisor[9]). This is exactly the type of information that is needed for aligning. If necessary this could be combined with a method of incentivising truthful feedback, such as described by Witkowski [2010]. This could also be helped to mitigate the harm from deception, which is the final point for discussion.

In our experimentation we only generate noise in the trust evaluation, not in the description of the evidence. Furthermore, if a hostile agent has knowledge of

[8] www.ebay.com
[9] www.tripadvisor.com

the aligning agent's trust model, it could tailor its alignment messages so that it can send false evaluations undetected. Luckily a lot of work has been done in detecting fraudulent, or inconsistent information, as described in the discussion of the state of the art (see Section 2.3.1). While our experimentation with a filtering method shows that it is not able to deal with subjective evaluations very well, it is designed to deal with deception. By combining a filtering method, such as POYRAZ, with an alignment method, such as FORTAM, the agent should be able to reap the rewards from both methods: using alignment to learn a translation from most agents, but, if the alignment fails according to some criteria (for instance, the coverage of the training set of the learning method drops below a certain percentage), the agent uses the filtering method to test whether the witness with whom alignment failed should be marked as deceptive.

Applications

While the experimental setup is limited, it shows that agents using trust alignment improve the utility of the communication, under the assumptions required for performing alignment. As stated in Section 3.2, these assumptions are:

- The agents aligning both observe a single, sufficiently large, set of interactions, which we call the set of shared interactions.
- Agents are willing and able to communicate their evaluations based on a single interaction.
- There is a domain ontology in which to describe objective properties of interactions.
- If background knowledge about the domain is used in evaluating trust, this background knowledge is shared between the agents in the system.

While at first these assumptions seem restrictive, we give a number of examples of real world applications in which these assumptions hold.

P2P routeing: One of the problems encountered in P2P networks is that of routeing information. Deciding which peer can be trusted to transfer the required information does not have a trivial answer, especially if the network is used for diverse purposes, such as streaming different types of media, for which different agents have different requirements. Current trust and reputation models offer a possible solution [Perreau de Pinninck Bas et al., 2010], but because the system is very dynamic, it is often impossible to have enough direct information about agents for calculating an agent's trustworthiness. Due to the decentralised nature of P2P protocols, there is no trusted authority that can store reputation and it is up to agents to communicate between each other about trust. Many P2P protocols allow for the tracking of the path along the various peers, with additional information available, such as time between hops or the physical location of the peers [Vu et al., 2010]. This information is freely available and we can consider such a path as the interaction that is shared among the peers. This

does require more information to be shared than current P2P protocols do: for agents earlier in the path to know about agents later in the path, this information needs to be made public, or at least shared among all the peers in a path. After this, however, all agents in the path share the entire interaction and can use such interactions to align. Trust evaluations may be based on a number of criteria, such as timeliness in forwarding a packet, sending the packet in the correct physical direction and not compromising the packet's contents. Different agents may evaluate this differently and thus to communicate about trust, they need to align. Because of the sheer amount of packets, it is reasonable to expect that two peers share enough interactions to perform alignment. Moreover, many P2P protocols provide an RDF-based language for the description of such interactions.

E-commerce: Trust and reputation mechanisms are often linked with e-commerce systems, with eBay's reputation system as the most famous example. Nevertheless, communication is problematic in this domain as well as people evaluate sales interactions differently: there are different attributes such as price, delivery time or, especially, the quality of the product, which can have different interpretations for each agent. However, the assumption of two agents observing the same interaction is probably too strong for this domain. Instead, we could relax this condition and work with similar interactions. For this, the agents must use some kind of similarity measure between their interactions. The problem of finding an adequate similarity measure for semantic data has been studied extensively for ontology matching and similar problems of the semantic web [Resnik, 1999; Rodríguez and Egenhofer, 2003]. If two interactions are sufficiently similar it is a trivial extension of the theoretical framework of Chapter 3 to consider these as the same token and we can continue to use our alignment methods in these cases. Each agent observes its own interaction, but the alignment between agents is based on similar interactions, rather than shared interactions. Note, however, that this requires extra communication about these interactions before aligning to establish the set of similar interactions. Furthermore, we add more uncertainty by using such a similarity measure, because there may be hidden factors that are very dissimilar in interactions that the agents think are similar. Despite this, the potential financial gain from choosing interaction partners more effectively in an e-commerce environment offsets these disadvantages.

4.3.6 Summary

The experimentation shows clearly that communication without alignment may have a negative influence on the accuracy of an agent's trust evaluations and thus that alignment is a necessary step when talking about trust. In our experiments all agents were sufficiently different from each other that the best a filtering method (POYRAZ) could achieve in the scenario was to filter out out all other agents' communication, resulting in a situation similar to no communication at all. In contrast, all alignment methods improve on this. We see that

even a simple alignment method such as calculating an average bias, can give a significant boost to the trust model's accuracy. AR&H's method and FORTAM function at least as well as not communicating even if all other agents are liars. FORTAM outperforms all other methods tested, by taking *the context* in which a trust evaluation was made into account.

FORTAM is especially effective, in comparison to other methods, when there are deceptive agents, who lie about their trust evaluations. Because it takes the contextual information into account, it can still learn an accurate alignment and thus maintain performance. The best other methods can hope to do is correctly filter out the lies, which both POYRAZ and AR&H's method manage to do. If the information provided about the context is also degraded FORTAM can no longer learn good alignments and it might be beneficial to use a hybrid approach with POYRAZ or another filtering method.

4.4 Summary

The contributions of this chapter are threefold:

- We present the FORTAM alignment method, a practical method for generalising SRAs using the TILDE ILP algorithm for performing First-Order Regression.
- We give a measure of the complexity of an alignment problem, that calculates the entropy between two different models. The higher the entropy the harder we expect the alignment to be. Although results seem to indicate that this could be true, the experiments are not conclusive. Specifically it seems that there is a threshold above which an increase in entropy does not make the problem harder. This could be due to the use of the $\mathcal{L}_{Domain}$ language, but further experimentation is necessary.
- We compare FORTAM to a number of other methods to deal with communication of trust evaluations. This experimentation shows that some form of alignment or filtering is necessary for communication about trust to be useful. Moreover, even a very basic alignment method that uses an average bias, is better at dealing with subjectivity in trust than a filtering method, although this changes if agents can be deceptive. FORTAM significantly outperforms the other methods, mainly because it can make use of the contextual information.

One additional note about FORTAM seems necessary: the learning method it uses is quite computationally intensive, compared to the other methods we tested. Additionally it requires communication about not only ratings of individual interactions, but also an objective description of the interaction it is based on. The functioning of this alignment method may very well depend on the expressiveness of the language for describing interactions. If such a language is very basic, then alignment may not be possible and a simpler method must be used. Similarly, privacy issues may arise in scenarios where agents are willing

to exchange trust evaluations, but not anything more. In such cases the best we can do is the method taking an average bias. Whether the increased complexity and communication load is worth the added performance should be evaluated per domain.

Part III

Trust Adaptation

Chapter 5

Trust Adaptation: State of the Art

In the sciences, we are now uniquely privileged to sit side-by-side with the giants on whose shoulders we stand.

–Gerald Holton

5.1 Introduction

In Part II of this thesis we presented Trust Alignment; a method for dealing with communicated witness information by finding a translation of the witness' evaluation into the receiver's frame of reference. A drawback, however, is that, in order to learn an alignment, a large number of shared interactions are necessary. If these are not available, then other solutions are necessary. Additionally, while Trust Alignment addresses the problem of a trust evaluation depending on the interactions, we have so far not considered how an agent's beliefs, and possibly more importantly, the task for which a trust evaluation is needed, affect the trust model. This is especially important, because this makes a trust model dynamic: it changes in accordance with the agent's beliefs, and with the goal for which an agent needs to interact. If trust models are dynamic, then alignments may be outdated, requiring agents to perform the alignment process multiple times with the same agent.

These two drawbacks of Trust Alignment can both be addressed with the method we present in this part of the thesis: Trust Adaptation. Rather than the receiver of a trust evaluation trying to learn a translation, the aim is for the sender of a trust evaluation to adapt the evaluation to the receiver's needs. For this, both agents must know, to a certain extent, how their beliefs and goals influence their trust model. Each agent's computational trust model must thus be fully integrated into its cognitive reasoning process: the agent must have knowledge about, and control over, the factors that influence the way it computes

a trust evaluation. In Section 5.2 we discuss the state-of-the-art methods that integrate trust into the reasoning process and distinguish AdapTrust from them. We present AdapTrust, an extension to the BDI framework for reasoning about trust, in Chapter 6.

In addition to trust being fully incorporated into the reasoning process, agents must communicate what beliefs and goals influence their trust evaluations, and in what manner. Furthermore, they must convince each other to adapt their model to their own needs. Argumentation is the usual approach to this type of communication, and the combination of argumentation with trust has recently received quite a lot of attention. We discuss the various approaches to combining trust and argumentation in Section 5.3, and in Chapter 7 we describe our argumentation framework and dialogue protocol for Trust Adaptation.

5.2 Cognitive Integration of Trust Models

The idea of integrating the trust model into the cognitive process of an agent is not new. There are various different methods for representing this cognitive process, but the integration of trust into this process is done mainly for BDI agents. BDI stands for Beliefs-Desires-Intentions and is a logical model for designing intelligent agents [Rao and Georgeff, 1991]. The beliefs represent the agent's knowledge about its environment, the desires the state of the environment the agent desires to achieve and the intentions represent the plans it intends to perform to achieve its desires. The BDI model has met with considerable success, and many systems for implementing intelligent agents follow this model to some degree or another [Bordini et al., 2007; Dastani, 2008]. From the point of view of trust, it offers clear advantages: by providing a crisp definition of the agent's beliefs and its goals, the role trust plays can be made explicit, incorporating it into the logical framework. There are different ways of doing so, and thus trust is treated differently by different models.

We focus on how an individual agent can reason about its trust model and we therefore require the agent to have introspection into its own trust model. We emphasise that computational trust models are for obtaining a trust evaluation from a set of inputs, as we stated in Section 2.2 (page 14). When given a finite set of inputs, such as beliefs about direct experiences or reputation, the trust model calculates a trust evaluation for a target. This can be represented mathematically as a function; however, the output of this function is dependent on, not just the inputs, but also the criteria an agent has for evaluating trust (we also briefly discussed this in Section 2.2).

In order for an intelligent agent to take full advantage of trust evaluations, the trust model needs to be integrated into the reasoning system, and the work that deals with an integration of trust into a BDI-agent can be divided roughly into two different approaches. The first approach is to consider computational trust models that, for the most part, take a belief-centric approach to trust. Work that takes this approach focuses mainly on how a trust model can be integrated into a BDI-agent and we discuss such work in Section 5.2.1. The

second approach is a more logic-orientated approach, and works along this line to provide formal logics for reasoning about trust. Such work is less focused on the computational aspects of trust, but focuses more on proving interesting properties of trust from an information-theoretic perspective. We describe such logics for trust in Section 5.2.2.

5.2.1 Cognitive computational trust models

Falcone et al. [2005] describe the socio-cognitive underpinnings of trust and show its functioning through experimentation using a computational model. They use a Fuzzy Cognitive Map (FCM) to model trust. An FCM is a graphical representation of the knowledge in a system. The nodes are concepts, and edges are causal conditions between the concepts. The value of an edge represents the strength with which one concept influences another. The values of the nodes can be interpreted as strengths and are subject to change, because each node's value depends on the strength of its neighbouring nodes and the influence these nodes have upon it. There are computational methods to calculate an equilibrium of an FCM. In Falcone et al.'s trust model the beliefs an agent has are nodes in an FCM, as is the trustworthiness of a target. Although they explicitly mention cycles and feedback being a possibility in FCMs, their computational model relies heavily on the fact that the FCM is a tree with trustworthiness as its root. This greatly facilitates the computation, because in the general case convergence of FCMs is not guaranteed [Zhou et al., 2006], and in order for an FCM to compute a trust evaluation convergence to a stable value is necessary. The difference between the theoretical, cyclical model, and the limitations on a practical use of it is important, because in the former beliefs can reinforce each other, thus causing a stronger influence on the trust evaluation depending on the values of other beliefs: the trust model could adapt to a changing environment. In the latter, however, the weights of the edges and aggregation functions within nodes, which are fixed, are the only factors that influence the effect any belief has upon the trust evaluation. It is thus unclear how an agent could adapt its trust model to different goals. Despite this, Falcone et al.'s model is a very interesting approach to tying the trust model into the cognitive structure of the agent (in this case only the belief base) and should be largely compatible with the AdapTrust model we describe in Chapter 6: Falcone et al.'s model could serve as an adaptive trust model, while AdapTrust provides a method for proactively adapting it to changing situations.

Burnett et al. [2011] take a very different approach and emphasise trust as a tool in fulfilling plans. They assume agents need to delegate tasks and discuss the role trust and reputation play in deciding whom an agent should optimally delegate tasks to. They assume different agents will perform differently at the various tasks and thus agents' trustworthiness depends partially on what task they need to perform. This is quite similar to the idea of role-based trust, but is integrated into the decision-making process of the agent. This is somewhat similar to the approach we take, in which the goal and plan of the agent can influence the computation of the trustworthiness of an agent, but their approach

focuses purely on the decision of delegating and does not deal with other cognitive aspects of trust. Specifically they do not use the beliefs of an agent.

BDI+Repage [Pinyol and Sabater-Mir, 2009a] attempts to take a more comprehensive approach, by integrating the Repage reputation model [Sabater et al., 2006] into the cognitive process of a BDI agent. While this is an interesting approach, it leaves the Repage model as a black box; specifying only how Repage should be connected to the various inputs and how the outputs of the calculation should be dealt with in the belief base. As such, it does not allow for reasoning about how to evaluate reputation, just about what to do with the evaluations after Repage has calculated them. This system presents the basic idea we build upon: it uses a multi-context system [Giunchiglia and Serafini, 1994] to represent the BDI agent and reputation model, which provides a clear way of integrating a trust or reputation model into the cognitive agent architecture. It does not, however, provide the mechanism needed to actually reason about the trust model and represents the trust computation in a single, monolithic context. We propose a more sophisticated method of representing the trust model in a BDI-agent that allows an agent to reason about and adapt its trust model.

5.2.2 Logics for reasoning about trust

Liau [2003] presented a modal logic for explicitly reasoning about trust. The modal logic used is specifically intended for reasoning about beliefs and information and shows how interesting and desirable properties of trust emerge from basic axioms about the logic. Dastani et al. [2004] extend the logic to deal with topical trust, leading to the logic being able to infer trust evaluations about specific topics, or tasks. This notwithstanding, these logical models do not give a definition of the computational process. Trust is an abstract concept, and if certain axioms about it are true, then other properties can be proved. This makes it possible to show the consequences of specific types of trust, such as if trust is transitive, or symmetric. It says nothing, however, about whether actual methods of computing trust fulfil such properties. An approach that attempts to rectify this somewhat is ForTrust [Lorini and Demolombe, 2008]. This logic integrates trust into a BDI logic and also allows for the proving of certain properties of trust from the basic axioms of the logic. However, in this case, the logic is conceptually nearer to a practical trust model, and there are implementations of ForTrust [Hübner et al., 2009; Krupa et al., 2009]. We show that these implementations can also be modelled within our framework in Section 6.5.2.

The logic-based approaches we have discussed differ significantly from our approach, in which we assume that a computational trust model is given, and provide a methodology for adapting this computational method. Specifically our aim is to give a declarative description of a computational trust model and allow the agent to reason about this: we take a bottom-up approach, starting with a computational trust model, rather than a top-down approach in which desirable properties are derived, regardless of whether these are based on realistic assumptions or not.

5.3 Argumentation and Trust

In the previous section we discussed how trust models are incorporated into BDI agent models, in order to reason about trust. In this section we discuss how trust can be combined with another important method for reasoning, namely argumentation. Trust and argumentation have been combined in a number of different manners. In Chapter 7 we present a new way for doing this, focusing on argumentation between two different agents in order to adapt the trust model to each other. Most work on argumentation and trust focuses on a single agent and how to incorporate information from different sources. We distinguish three manners for combining trust and argumentation in the current literature. The first is to use the trustworthiness of a source of information within an argument to decide whether it is acceptable or not and we discuss this in Section 5.3.1. The second is to incorporate information from the argumentation into the computation of a trust evaluation, highlighted in Section 5.3.2. Finally, argumentation has been used as a method for communicating more accurately about trust, just as we propose to do, and we discuss the state of the art in Section 5.3.3.

5.3.1 Trusted arguments

One of the problems encountered in a multi-agent society is that agents use information from a variety of sources in their reasoning process. Such sources may be more, or less, reliable. Argumentation frameworks [Rahwan and Simari, 2009] provide a way of reasoning using such information, by giving a formal method for dealing with conflicting information, often with degrees of uncertainty. If we consider the sources' trustworthiness as a measure of confidence in the information they provide, then the link between argumentation and trust is obvious. This is precisely the approach taken by Tang et al. [2010]. Their work uses the trustworthiness of the information's sources as a measure of the probability that information is true.

Parsons et al. [2011] generalise this work and give a formal account of the properties of an argumentation framework, when considering different ways of calculating trust and combining arguments. Specifically, they try to satisfy the condition [Parsons et al., 2011, Property 9]:

> If an agent has two arguments A_1 and A_2 where the supports have corresponding sets of agents Ag_1 and Ag_2 then A_1 is stronger than A_2 only if the agent considers Ag_1 to be more trustworthy than Ag_2.

This condition states that arguments grounded in information from less trustworthy sources cannot defeat arguments with grounds from more trustworthy sources. The work then describes some computational methods for treating trust and argumentation that satisfy this condition. Unfortunately, these methods have very strict properties. The most troublesome is the assumption that trust is transitive, while current sociological research indicates that this is only true in very specific situations [Castelfranchi and Falcone, 2010]. Despite this, the work lays a solid theoretical foundation for incorporating trust into argumentation.

Another approach to incorporating trust into argumentation is taken by Villata et al. [2011]. Their work takes a similar approach to Tang et al.'s work and explicitly represents the sources providing the different arguments. The major contribution is in allowing argumentation about sources' trustworthiness. It allows meta-arguments to support and attack statements about the trustworthiness of sources. The effect of this meta-argumentation is to change the confidence agents have in the sources providing different arguments, which, in turn, changes the strength of the various arguments. This is thus a combination of two different forms of combining trust and argumentation. On the one hand, the meta-argumentation is used to evaluate sources. On the other hand, the trustworthiness of these sources is used as the strength of a different set of arguments. This combination seems very powerful, but in relying purely on argumentation for evaluating sources' trustworthiness, the complexity inherent in a trusting relationship is lost. As Villata et al. [2011, page 81] state themselves:

> Trust is represented *by default* as the absence of an attack towards the sources, or towards the information items and as the presence of evidence in favour of pieces of information.

Most contemporary trust models take a more fine-grained approach to representing trust (such as a probability that the target will act in a trustworthy manner), that more accurately reflects the view that trust is a decision based on, often conflicting, pieces of information. In the next section we discuss some methods for incorporating argumentation into such models of trust.

5.3.2 Argument-supported trust

Prade [2007] was, insofar as we know, the first to present a model for incorporating argumentation into a trust model. In this work, trust is considered along a variety of dimensions. Specifically, trust is split into trust categories, which represent different behavioural aspects of the target. Each behavioural aspect may be qualified as good, or not good, for a target agent. The trust model consists principally of a rule-base in which levels of trust are related to the target's behaviours. The trust model then uses the target's actual behaviour to perform abduction and find the range in which the trust evaluations must fall. This range is the trust evaluation of a target.

The arguments in Prade's work thus constitute the trust model itself. By performing the abduction with the rules in the trust model, the agent constructs arguments for its observations. The arguments are thus not part of the input of the trust model, but an inherent part of the calculation process.

Matt et al. [2010] do consider arguments as a separate source of information for calculating the trustworthiness of a target. They propose a method for combining justified claims about a target with statistical evidence for that target's behaviour. These justified claims provide context-specific information about an agent's behaviour. The basis for their trust model is Yu and Singh's model [2002], which uses a Dempster-Shafer belief function to provide an estimate of whether an agent will fulfil its obligations, given some evidence about that

agent's past behaviour. Matt et al. propose to extend this model with a method for evaluating arguments about the contracts, in which an agent's obligations are fixed and guarantees are provided about the quality of interactions. Specifically these contracts specify the requirements along a number of dimensions. These dimensions are aspects of an interaction, such as availability, security or reliability. For each dimension an agent wishes to take into account when evaluating trust, it can construct an argument forecasting the target's behaviour with regards to that dimension, given the specification of a contract. For each dimension d, Matt et al. can construct the following arguments:

- An argument forecasting untrustworthy behaviour, based on the fact that the contract does not provide any guarantee regarding d.
- An argument forecasting trustworthy behaviour, based on the fact that there is a contract guaranteeing a suitable quality of service along dimension d.
- An argument that mitigates a forecasting argument of the second type, on the grounds that the target has, in the past, "most often" violated its contract clauses concerning d.

They then integrate these arguments into Yu and Singh's trust model, by providing new argumentation-based belief functions that combine the information from forecast arguments with evidence. By incorporating more information, the agent should be able to obtain more accurate trust evaluations, and Matt et al. show this empirically.

All the methods discussed so far highlight the different aspects of argumentation and trust for dealing with uncertain information. Either by applying trust to argumentation in order to get more accurate arguments, or by applying argumentation to trust to obtain more accurate trust evaluations. There is, however, another useful way to combine trust and argumentation that has not been discussed so far. Evaluating trust often requires communication, but this communication may be unreliable, simply because trust is a subjective concept. By having agents argue about the trust evaluations themselves and seek information about *why* an agent has a specific trust evaluation, one may discover whether the other's communicated trust evaluation is useful to it, or whether its interpretation of the various criteria for evaluating trustworthiness are too different from the own criteria [Pinyol and Sabater-Mir, 2009b].

5.3.3 Arguments about trust

We recall our adaptation of Castelfranchi and Falcone's definition of trust as a choice, in a certain context, to rely on a target agent to perform a task, in order for a specific goal to be achieved (see Section 2.2 on page 14). The context in which an agent trusts, is represented by an agent's beliefs about the environment, and the goal is something the trustor wishes to achieve. Therefore trust is an agent's personal and subjective evaluation of a target. When communicating

such a subjective evaluation it is often unclear how useful this evaluation is to the receiving agent: it needs to discover whether the context, in which the communicated evaluation was made, is similar to the context in which the receiver needs to evaluate the target.

Pinyol [2011] proposes a framework to argue about trust evaluations and decide whether another agent's communicated evaluations can be accepted or not. In Chapter 7 we discuss this framework in detail because we extend this framework in our own work. Pinyol's decision of whether a communicated evaluation should be accepted or not can be considered as a method for filtering out unreliable witness information and it thus has the disadvantages associated with these methods that we discussed previously: if many agents in the system are too different, filtering out all their communications results in a sparsity of information; furthermore, if sanctions are applied to sources of "unreliable" information, agents might stop providing evaluations entirely (see Section 2.3.1, page 27).

Our solution is to allow agents to extend their argumentation, and on the one hand convince each other of the beliefs they have about the environment, and on the other hand adapt their trust models to each other's needs, thereby computing a trust evaluation that is personalised to the other agent. In the next chapter we present AdapTrust, an extension of a BDI-logic that allows computational trust models to be integrated into a cognitive reasoning process, and in Chapter 7 we present the communication model that uses AdapTrust and an argumentation framework to enable this personalised communication about trust.

5.4 Summary

In this chapter we have given an overview of the ways in which trust can be integrated — and used — in agent reasoning. We started with a discussion of the methods that integrate trust into a cognitive model for agent reasoning, most prominently the BDI agent model. The aim is to allow pre-existing reasoning mechanics, mainly in the belief base of an agent, to deal with trust evaluations. Nevertheless, the approaches discussed do not describe how the beliefs and goals an agent has affecth the trust computation. Approaches that describe trust using formal logics abstract away from the actual agent model, and thus the problem of how goals and beliefs affect the computation, but at the cost of not giving a clear methodology for how the formalisation can be applied in a computational agent. The approach we propose in the next chapter falls somewhere between the two approaches. AdapTrust provides a methodology for integrating an agent designer's choice of computational trust models into an abstract model of a BDI agent.

Argumentation provides another way of reasoning about trust. Firstly, trust information can aid in the reasoning process. This approach is similar to the integration of trust models into a BDI model for reasoning, but discusses its implementation using argumentation, a form of defeasible reasoning, as the spe-

cific method for reasoning in the belief base. Other works discuss the use of arguments in the computation of the trustworthiness of targets and finally we discussed Pinyol's work on using arguments about trust to decide whether witness information is reliable or not. In Chapter 7 we return to this work and extend it, using AdapTrust to allow an agent to adapt its trust model, based on an argumentation dialogue about trust with another agent.

Chapter 6

AdapTrust

If you try and take a cat apart to see how it works, the first thing you have on your hands is a non-working cat.

–Douglas Adams

6.1 Introduction

In Section 2.2 we described computational trust models as algorithmic methods for calculating an agent's trust in a target based on the agent's own interactions with the target, and information about the target that is available in the environment. The inputs for this model may be direct interactions with the target or direct communications from other agents in the system, giving their own trust evaluations of the target; it may be reputation information; or it may be any other source of information available in the system. The trust model evaluates the various inputs and then aggregates the intermediate evaluations to calculate the evaluation of the target.

The problem with this way of approaching trust models, and thus the way they have been discussed so far in the literature, is that an agent is unable to change its trust model if it notices a change in the environment. From the agent's perspective, the trust model would appear to be a "black box" with as input the various information sources and as output an evaluation of how trustworthy the target is. In contrast, the definition of trust that we use, adopted from Castelfranchi and Falcone [2010], is that "trust is a (conscious and free, deliberated) reliance based on a judgement, on an evaluation of the target's virtues, and some explicit or entailed prediction/expectation", and trust is thus not just an evaluation of a target, but an integral part of the decision making process of an agent in a social environment. For a trust evaluation to be meaningful, the evaluation process must take the environment and the reason why the evaluation is being made into account. This is especially so in an open MAS, where the environment may change.

As an example, consider the following situation. An agent has the task of routinely buying items on an electronic marketplace and to decide, by using its trust model, which seller to interact with. In general the agent's owner requires the agent to buy items as cheaply as possible and does not mind if, to achieve this, the item is delivered late. One day, however, the agent is assigned specifically to buy an item prioritising speedy delivery. This is problematic if the agent's trust model is hard coded to disregard delivery speed when evaluating salesmen. This is an example for evaluating direct experiences, but in other parts of the trust model the same thing can happen. For instance, at the level of aggregation, if the agent's environment contains mostly truthful agents, then it can use reputation information. If the environment changes, however, and most agents are liars, the agent using reputation information is misguided. Some models are equipped to deal with these changes, but they do so reactively [Teacy et al., 2006; Vogiatzis et al., 2010]. If the agent *knows* the environment has changed it cannot proactively adapt its trust model. Such issues arise at all levels during trust computation and can be triggered by changes in the beliefs and the goals of an agent. Contemporary trust models are not equipped to deal with this type of proactive adaptation, in which the agent's goals and beliefs can precipitate a change in the way an agent calculates its trust evaluations. Furthermore, because trust models are treated as a black box, their integration into a cognitive agent also does not allow for proactive adaptation.

Another important aspect of trust, which few computational models deal with, is that trust is inherently multifaceted. An agent is evaluated with respect to some role, or action, which it is expected to perform. The evaluating agent requires this action to be performed to achieve a specific goal, which is why it requires the trust evaluation in the first place. This goal, and the action the target is required to perform to achieve it, can change the parameters of the trust model: an agent selling an item in an auction may be evaluated differently to an agent buying an item.

In this chapter we present AdapTrust, an agent model capable of reasoning about trust and proactively changing the trust model to reflect this reasoning. We do not present a new trust model, but rather propose an extended BDI framework designed to work with existing models. We provide a method for integrating computational trust models into a cognitive agent model. We do this by considering the trust model in as declarative a way as possible, while still relying on the underlying computational process for calculating trust evaluations. While this does not provide the agent with introspection into the actual computational mechanism its trust model uses, it opens the black box of trust sufficiently to allow the agent to proactively adapt its model. Because of the way that trust is fitted into the agent model, it is possible to plug in different trust models quite easily.

First, however, we take a closer look at how computational trust models function. We stated above that computational trust models are functions that calculate a trust evaluation of a target, based on a number of inputs. Nevertheless, the output may depend on more than just these inputs: there may be

constants that play a role in the actual calculation. Let us consider a trivial example in which the trust model computes an evaluation as a linear transformation of a single input value. The computational trust model therefore computes the function $f(x) = ax + b$ and the output depends, not just on x, but on the precise values of a and b. Actual trust models obviously compute very different functions with different, generally non-numerical, input. The principle, however, is the same and different values for the constants in the functions can lead to very different outputs. In general, we are not interested in the actual algebraic operations the functions use and abstract away from these. Nevertheless, we are interested in the constants that are used in the functions and how these can be *adapted*, within the limitations of the algebraic operations. For example, if the agent uses the linear transformation above, our idea is to allow for change in the values for a and b, but not to replace the formula with, for example, a quadratic equation. These changeable constants are what we call the parameters of the trust model.

The parameters available for adaptation, and the effects they have on the actual computation, depend entirely on the computational trust model an agent uses. In order to use the computational trust model in AdapTrust, we require it to have at least one parameter. Such parameters are, for instance, the weights in a weighted average, the decay rate at which older evidence is discarded, or the distance in a social network at which point to disregard opinions.

Many trust models use a number of different parameters. By considering these trust models as functions, we see that each of these parameters has a different effect on how the input variables are used to compute the output. Specifically, each parameter makes the influence of input variables more or less important and thus the value of a parameter should be "governed" by the importance the agent wants to assign to the variables it affects: the more important the variable, the larger its influence should be upon the output of the trust model. For instance, consider a trust model that has a parameter which governs the relative importance of direct experiences and witness information, such as is the case in ReGreT [Sabater et al., 2002]. If the agent believes it is surrounded by liars it should give little importance to witness' communications. The value assigned to the parameter should reflect this, ensuring that direct experiences are given more importance in the trust calculation than communicated ones. In general, we require that the agent is capable of instantiating a trust model with different values for its parameters, which reflect the importance the agent assigns to the different variables that play a role in calculating trust evaluations. In the continuation we call these variables *factors*, because they are the factors that contribute to a trust evaluation.

A further requirement is that the function is valid for any combination of values for the parameters: the parameters are linked to the importance of different factors taken into account in calculating trust. Thus if some factor becomes more important this can cause a change in the parameters. Any such change should result in a valid trust model. If, for instance, we have $f(x) = log(ax)$ as the function that the trust model computes, and we know $x \in \mathbb{R}^+$, then we cannot

choose the parameter $a <= 0$, because then the trust model is no longer valid: the function is no longer defined on its domain. We are thus not completely free in the choice of parameters and must be especially careful if multiple parameters can be changed to make sure any combination of possible values results in a valid trust model.

AdapTrust is designed in such a manner that a large number of current computational trust models can be incorporated into a BDI reasoning system by considering the parameters they take into account and specifying what factors can influence these parameters. It allows these factors to influence the trust model automatically, thus integrating the trust model into the reasoning system of the agent. Furthermore AdapTrust is inherently multifaceted, as the entire cognitive stance of the agent can influence the trust model. In Section 6.2 we present the various logics we use and the basic BDI framework we extend. Section 6.3 describes the first part of this extension: a manner of specifying a trust model to allow the agent to reason about it, using priorities over the factors that influence its parameters. Section 6.4 introduces the mechanics for performing this reasoning. This is done with rules that link the goals and beliefs of an agent to the priorities that specify its trust model. Finally we demonstrate how AdapTrust allows an agent to reason about its trust model, using BRS [Jøsang and Ismail, 2002], ForTrust [Lorini and Demolombe, 2008] and ReGreT [Sabater, 2003], three well known contemporary trust models, in Section 6.5.

6.2 Preliminaries

AdapTrust provides a way to integrate computational trust models into a cognitive model of an agent. We use a multi-context representation of a BDI-agent as the cognitive model, and in this section, present all the logics and formalisms needed for this. Accordingly, we start by introducing the multi-context representation of a BDI-agent. The specification of an agent using a multi-context system (MCS) has several advantages for both the logical specification and the modelling of the agent architecture [Parsons et al., 1998]. From a software engineering perspective, an MCS supports modular architectures and encapsulation. From a logical modelling perspective, it allows the construction of agents with different and well-defined logics, and keeps all formulae of the same logic in their corresponding context. This increases the representational power of logical agents considerably, and at the same time, simplifies their conceptualisation. The MCS paradigm is therefore a popular formalism for extending the basic BDI logic. For instance, Criado et al. [2010] and Joseph et al. [2010] use an MCS to allow a BDI agent to reason about norms, and Pinyol and Sabater-Mir [2009a] use an MCS to incorporate trust into a BDI agent. We follow a similar approach to this last work, but as explained in Section 5.2, Pinyol et al.'s approach firstly only deals with Repage, and secondly does not allow for the adaptation of the model: their integration focuses on making decisions to interact, based on trust. Finally, Sabater et al. [2002] show how the specification of BDI-agents using an MCS corresponds directly with the concept of modules in

software engineering, and allows for the rapid and easy design of an executable agent architecture.

6.2.1 Multi-context systems

Multi-context Systems (MCS) provide a framework that allows several distinct theoretical components to be specified together, with a mechanism to relate these components [Giunchiglia and Serafini, 1994]. An MCS consists of a family of contexts, which are logically connected by a set of bridge rules. Each context contains a formal language and a theory of that language. We say a sentence is in a context if it is a logical consequence of the theory in that context. Bridge rules serve to relate theories between contexts. They can be seen as inference rules, but rather than inferring conclusions within a context, the premises and conclusion are in different contexts. Bridge rules have the form:

$$\frac{C_1 : X; C_2 : Y}{C_3 : Z}$$

where X, Y and Z are *schemas* for sentences in their respective contexts. The meaning of a bridge rule is that if a sentence complying with schema X holds in context C_1, and a sentence with schema Y holds in context C_2, then the corresponding sentence with schema Z holds in context C_3. This is true in the logical sense, but the bridge rules have a second use: they represent the operational procedures in the system. The schema in the conclusion might be the outcome of some operation. In such cases we will give an abstract description of the function that performs this operation.

Let I be the set of context names, then an MCS is formalised as: $\langle \{C_i\}_{i \in I}, \Delta_{br} \rangle$ with contexts C_i and Δ_{br} a set of bridge rules. In Section 6.2.3 we show how to represent a BDI-agent as an MCS, but first we introduce the logics that we require in our integration of a trust model into a BDI system.

6.2.2 Logics

In an MCS, each context is specified using a logic. In this section we present the various different logics that we use in the contexts.

First-Order Logic (FOL)

While we stick to the standard definition of first-order predicate logic, we need to introduce the notation, which is used throughout this chapter. The syntax of FOL is defined using a set of constants $\mathcal{C}$, a set of function symbols $\mathcal{F}$, and a set of predicate symbols $\mathcal{P}$. Each function and predicate symbol has a fixed arity, greater than one. Terms and formulae are given in the usual manner, using quantifiers $\forall, \exists$ and connectives $\neg, \wedge, \vee, \rightarrow$.

First-Order Dynamic Logic (FODL)

We use first-order dynamic logic (FODL), as first proposed by Harel [1979]. FODL is first-order logic that is extended by adding action modalities to it. We use FOL as defined above, and any FOL-wff is a *program-free* FODL-wff. We now define the set RG of first-order regular programs and the set of FODL-wffs by simultaneous induction as follows:

- For any variable x and term e, $x := e$ is in RG.
- For any program-free FODL-wff φ, $\varphi?$ is in RG.
- For any α and β in RG, $(\alpha;\beta)$, $(\alpha \cup \beta)$ and α^* are in RG.
- Any FOL-wff is an FODL-wff.
- For any FODL-wffs φ and ψ, α in RG and variable x the following are FODL-wffs:
  - $\neg\varphi$
  - $\varphi \wedge \psi$, $\varphi \vee \psi$, $\varphi \rightarrow \psi$
  - $\forall x : \varphi$, $\exists x : \varphi$
  - $[\alpha]\varphi$

Programs formed by the application of the first and second rule are called *assignments* and *tests*, respectively. We will refer to both of these as *basic actions*, while any other type of program is a *composite* program. For the formal semantics of FODL we refer to Harel [1979], but for this chapter it is sufficient to have an intuitive understanding. The semantics of *program-free* formulae is the same as for standard FOL. The semantics of $[\alpha]\varphi$ can be summarised as: after performing α, φ holds. $\alpha;\beta$ is sequential composition: $[\alpha;\beta]\varphi$ means that after first performing α and then performing β, φ holds. Similarly, $\alpha \cup \beta$ is non-deterministic choice: $[\alpha \cup \beta]\varphi$ means that after either performing α or performing β, φ holds; and α^* is repetition: $[\alpha^*]\varphi$ means that φ holds after performing α any finite number of times (including none).

Priority Logic (PL)

In our specification of the trust model we will require a logic to formalise a structure of priorities among predicates in our MCS. We call this logic Priority Logic (PL)[1]. It is a subset of first-order logic, following the system defined by Schorlemmer [1998] for a logic of binary relations. In PL we admit only two predicate symbols: the predicate symbols of order, denoted $\succ$ and $=$. As is customary, we use infix notation for such predicates. For any two constants a and b, $a \succ b$ and $a = b$ are PL-atoms. Using this we define PL-wffs as:

- Any PL-atom π is a PL-wff.

[1] Note that this logic is unrelated to Wang et al.'s logic [1997] with the same name.

- For any PL-atom π, $\neg\pi$ is a PL-wff.
- For any PL-wffs π and ξ, $\pi \wedge \xi$ is a PL-wff.

The semantics are given by standard FOL-semantics, but with the following axiom schemas for the predicates, for any constants a and b:

$\succ$-irreflexivity: $\neg(a \succ a)$

$\succ$-transitivity: $(a \succ b) \wedge (b \succ c) \rightarrow (a \succ c)$

$=$-reflexivity: $a = a$

$=$-symmetry: $(a = b) \rightarrow (b = a)$

$=$-transitivity: $(a = b) \wedge (b = c) \rightarrow (a = c)$

incompatibility of $\succ$ and $=$: $a = b \rightarrow \neg(a \succ b)$

6.2.3 Multi-context representation of a BDI-agent

With all the logics described, it is time to present our specification of a BDI-agent using an MCS. We will not yet define the contexts used for reasoning about the trust model: in this section we present the contexts and bridge rules for a BDI-agent, whereas Sections 6.3 and 6.4 extend this framework to include reasoning about trust. A BDI-agent is defined as an MCS: $A = \langle\{BC, DC, IC, PC, XC\}, \Delta_{br}\rangle$.

We first describe the contexts, before defining the bridge rules. The first three contexts correspond to the classical notions of beliefs, desires and intentions as specified by Rao and Georgeff [1991]. The first, the Belief Context (BC) contains the belief base of the agent. We use an FODL language, as described in the previous section, to represent beliefs. Let $\mathcal{P}_{Bel}$, $\mathcal{F}_{Bel}$ and $\mathcal{C}_{Bel}$ be the predicates, functions and constants required to describe the agent and its environment, then $\mathcal{L}_{Bel}$ is the FODL language generated from them. If $\varphi \in \mathcal{L}_{Bel}$ is in the belief context, this means the agent believes φ holds.

In the Desire Context (DC) we use an FOL language $\mathcal{L}_{Des}$ to represent the agent's desire base, generated from the same set of predicates, functions and constants as the belief base. The agent's desires are thus represented in the program-free segment of the logic for the belief base. If $\psi \in \mathcal{L}_{Des}$ is in the desire context, then the agent desires ψ.

The Intention Context (IC) holds the intention base of the agent and uses the FODL language $\mathcal{L}_{Int}$, a subset of the language that the belief context uses: let $\alpha \in RG$ and ψ an FOL-wff, then $[\alpha]\psi \in \mathcal{L}_{Int}$. In other words, an intention is a sentence in an FODL language consisting only of program-free formulae preceded by a program. Intuitively, the meaning is that if $[\alpha]\psi$ is in the intention context, then the agent has the intention to achieve $\psi \in \mathcal{L}_{Des}$ by acting on plan $\alpha \in RG$. We will use goal and intention interchangeably and write the symbol γ as shorthand for such a goal.

In addition to these mental contexts, we follow Casali's lead [2008], and define two functional contexts: the Planner Context (PC) and the Interaction Context (XC). The former is in charge of finding plans to achieve an agent's desires, while the latter is an agent's way of interacting with the world: it controls an agent's sensors, performs basic actions and sends and receives messages. To connect these contexts to the mental contexts of beliefs, desires and intentions, we use an additional notational device: quoting.

Definition 6.1 (Quotation operator)**.** The *quote-operator* '.' transforms programs, PL-wffs, FODL-wffs and sets of FODL-wffs into first-order terms.

The Planner Context uses a first-order language restricted to Horn clauses, and a theory of planning uses some special predicates for representing plans:

- $basic\_action(\text{`}\alpha\text{'}, \text{`}\Phi_{Pre}\text{'}, \text{`}\Phi_{Post}\text{'})$, where α is a basic action in RG, and Φ_{Pre} and Φ_{Post} are program-free subsets of $\mathcal{L}_{Bel}$. This allows for the definition of basic capabilities of the agent, together with their pre- and post-conditions: for α to be executable, Φ_{Pre} must hold (or in other words, be in the agent's belief base). Φ_{Post} is guaranteed to hold on successful completion.
- $plan(\text{`}\alpha\text{'}, \text{`}\Phi_{Pre}\text{'}, \text{`}\Phi_{Post}\text{'})$, where α is any program in RG, and Φ_{Pre} and Φ_{Post} are program-free subsets of $\mathcal{L}_{Bel}$. The meaning of this is the same as above, but allows for composite plans.
- $bestplan(\text{`}\psi\text{'}, \text{`}\alpha\text{'}, \text{`}\Phi_{Pre}\text{'}, \text{`}\Phi_{Post}\text{'})$ corresponds to the agent's notion of the best instance of a plan to achieve desire ψ. At the very least we require $\psi \in \Phi_{Post}$.

In addition, the planner context contains two special predicates to choose plans based on the agent's beliefs and desires:

- $belief(\text{`}\Phi\text{'})$, with $\Phi \subseteq \mathcal{L}_{Bel}$.
- $desire(\text{`}\psi\text{'})$, with $\psi \in \mathcal{L}_{Des}$.

This context is more operational than the other contexts described thus far: its internal logic is unimportant, and its function in the reasoning is to select a specific plan for the agent to execute in order to achieve a goal ψ, given the current beliefs about the world. The agent has some method of selecting the *best* plan to achieve a desire ψ, and this is the plan returned. How formulae of the form $belief(\text{`}\Phi\text{'})$ and $desire(\text{`}\psi\text{'})$ are introduced into the PC is described in the next section about bridge rules. This is the only context presented in this section that requires modification to allow for reasoning about trust because, being orientated towards single agents, it does not accommodate actions that require other agents' participation. We will revisit this context in Section 6.3.3 to add this possibility.

The Interaction Context is also a functional context, but in a different sense from the planner context, as it encapsulates the sensors and actuators of an agent. It contains the following special predicates:

- $do(\text{`}\alpha\text{'})$, where $\alpha \in RG$. This has the meaning that α is performed by the agent, although at this abstract level we do not deal with the possibility of failure. Note that if α is a composite action, the Interaction Context has some way of decomposing this into a sequence of basic actions which can be executed.

- $sense(\text{`}\varphi\text{'})$, with φ an FOL-wff. This has the meaning that φ is observed in the agent's environment.

Bridge rules

One of the main advantages of using an MCS to represent an agent is that it separates the deduction mechanism inside the context from the deduction between different contexts. Reasoning internal to the contexts is defined in terms of the deduction rules of the logic used within a context, while bridge rules allow for inference between the contexts. The former corresponds to reasoning about beliefs, or desires, individually, while the latter corresponds to the BDI notion of a deliberation cycle: inference between the contexts. In the definition of the bridge rules below, and throughout the remainder of this chapter, we use symbols $\Phi, \Phi_{Pre}, \Phi_{Post} \subseteq \mathcal{L}_{Bel}$, $\psi \in \mathcal{L}_{Des}$ and $\alpha \in RG$. We start with the two bridge rules to add the beliefs and desires into the planning context, so that the beliefs can be used to find plans to achieve the desires:

$$\frac{BC : \Phi}{PC : belief(\text{`}\Phi\text{'})} \qquad \frac{DC : \psi}{PC : desire(\text{`}\psi\text{'})} \tag{6.1}$$

With this information the planning context deduces formulae, with predicate *bestplan* for any desire ψ the agent wishes to fulfil and any belief base Φ, such that $\Phi_{Pre} \subseteq \Phi$:

$$\frac{PC : bestplan(\text{`}\psi\text{'}, \text{`}\alpha\text{'}, \text{`}\Phi_{Pre}\text{'}, \text{`}\Phi_{Post}\text{'}) \qquad DC : \psi}{IC : [\alpha]\psi} \tag{6.2}$$

Similarly an agent has a bridge rule to allow for the execution of an intention:

$$\frac{DC : \psi \qquad IC : [\alpha]\psi}{XC : do(\text{`}\alpha\text{'})} \tag{6.3}$$

Upon execution, the effect of the agent's actions are added back into the belief base:

$$\frac{PC : basic\_action(\text{`}\alpha\text{'}, \text{`}\Phi_{Pre}\text{'}, \text{`}\Phi_{Post}\text{'}) \qquad XC : do(\text{`}\alpha\text{'})}{BC : \Phi_{Post}} \tag{6.4}$$

Note that none of these bridge rules take temporal considerations into account, or the possibility of failure: Φ_{Post} is instantly added to the belief base. We also require some form of belief revision to keep the agent's beliefs consistent and up-to-date. Here we just give the minimal specification to allow for reasoning about trust, but one could think of extending the model to deal with such

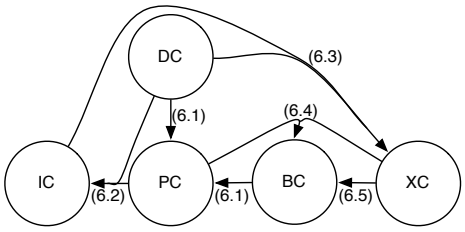


Figure 6.1: The MCS specification of a BDI-agent, without the contexts required for reasoning about trust. Nodes are contexts and (hyper)edges are bridge rules, with the labels corresponding to those in the text.

issues. The last bridge rule we define now is a simple rule for receiving sensory input:

$$\frac{XC : sense(\text{`}\Phi\text{'})}{BC : \Phi} \tag{6.5}$$

For the correct specification of a BDI-agent further bridge rules are necessary. Most notably those to allow for a chosen level of realism. Rao and Georgeff [1995] specify three different levels of realism: strong realism (if an agent desires a formula to hold, it will believe the formula to be an option), realism (if an agent believes a formula, it will desire it) and weak realism (if an agent desires a formula to hold, it will not believe the negation of that formula to be inevitable). These each correspond to a different set of bridge rules between the belief base, desire base and intention base [Parsons et al., 1998]. Luckily, none of this affects our specification of trust, and we leave the choice of the level of realism open. The MCS as specified thus far is summarised in Figure 6.1.

6.3 Specifying Trust Models

The MCS described in the previous section lays the groundwork for an agent that can reason about trust. We assume a trust model is an algorithm to calculate a trust evaluation based on a number of inputs. These inputs are formulae in one of the contexts of the agent, and we distinguish different types of input:

1. the target agent to be evaluated
2. beliefs about direct experiences
3. communicated trust evaluations
4. the desire the agent wishes to fulfil, for which it needs the target agent to perform some action

We will assume that the agent has a specific unary predicate $agent \in \mathcal{P}_{Bel}$, which is used to identify the agents in the system. We will use the shorthand

$Agents = \{X \in \mathcal{C}_{Bel} \mid BC : agent(X)\}$, the set of identifiers of agents in the system. The first input of the trust model, the target agent to be evaluated, is thus an agent $a \in Agents$. The second and third inputs, the beliefs about direct experience and communicated trust evaluations, are also formulae in the belief context. While we did not model communication in our MCS, we can consider it as part of the interaction context. Sending a message is simply a basic action of the agent and *do* performs that basic action, sending the specified message. Similarly, *sense* also receives communications from other agents. We therefore have two subsets of the belief base Φ_{DE} and Φ_{Comm} representing the agent's beliefs about direct experiences and communications respectively. Furthermore, we assume that such beliefs form a distinguished subset in the language of beliefs. We take $\mathcal{P}_{DE} \subset \mathcal{P}_{Bel}$, the set of predicates about direct experiences and $\mathcal{L}_{DE}$, the set of FODL-wffs generated in the same way we described in Section 6.2.2. It is easy to see $\mathcal{L}_{DE} \subset \mathcal{L}_{Bel}$. The same holds for $\mathcal{P}_{Comm}$ and $\mathcal{L}_{Comm}$. The last input corresponds to the reason why the agent is evaluating the trustworthiness of the target in the first place: trustworthiness is a tool used to help the agent in choosing a partner to achieve some goal. Depending on the goal the agent wishes to achieve, the way the trust evaluation is computed changes. As such, the goal γ that the agent is attempting to achieve is the fourth input for the trust model.

Some trust models, such as the one described by Hermoso et al. [2009], also consider norms as input for the trust model, and Pinyol and Sabater-Mir [2010] present a method of incorporating this into an MCS. A discussion about norms is outside the scope of this thesis. We will assume the agent's background knowledge is affected by norms and that this may affect how the agent specifies its trust model; however, we will disregard norms in the use of the trust model: possible norm violations could already be encoded in the agent's direct experiences and received communications.

The output of the trust calculation is a trust evaluation: a predicate in a domain-specific language for describing trust. This predicate is an element of $\mathcal{P}_{Bel}$, and the set of all possible trust evaluations $\mathcal{L}_{Trust}$ is a program-free subset of $\mathcal{L}_{Bel}$.

This abstract description of a trust model, in terms of its in- and output, corresponds with the abstract algorithmic view of a trust model in Section 2.2: a trust model is a Turing computable function. We call this function ***trust_calculation***, and regardless of the actual computational trust model, we see that the domain of this function is the set of the agent's beliefs, goals and agents under evaluation, and its range is the set of possible trust evaluations. For example, if an agent wants to evaluate a salesman s for the goal of having a bicycle by performing the basic action *buy_bicycle*, then it can evaluate s using its beliefs about interactions $\Phi_{DE} \subseteq \mathcal{L}_{DE}$ and communications $\Phi_{Comm} \subseteq \mathcal{L}_{DE}$ using its trust model. This may result in a trust evaluation of agent s with regards to the goal $[buy\_bicycle]have\_bicycle(me)$ with, for example, value *trustworthy*. We can represent this in a functional form as the function ***trust_calculation***, with the inputs Φ_{DE}, Φ_{Comm},

$[buy\_bicycle]have\_bicycle(me)$, and the agent s as described above:

$$\textbf{\textit{trust\_ calculation}}(\Phi_{DE}, \Phi_{Comm}, [buy\_bicycle]have\_bicycle(me), s) = trust(s, \text{`}[buy\_bicycle]have\_bicycle(me)\text{'}, trustworthy)$$

In Section 6.1 we stipulated that, for reasoning about the trust model to have any effect, this model must have parameters that can be adapted to the agent's needs. In other words, ***trust_ calculation*** is a parametric function.

Definition 6.2 (Parameters of a trust model)**.** Let ***trust_ calculation*** be a trust function, then we define the set $Params_{\textbf{\textit{trust\_ calculation}}}$ as the set of *parameters* of ***trust_ calculation***.

We also define the function $labels : Params_{\textbf{\textit{trust\_ calculation}}} \rightarrow 2^{\mathcal{C}_{PL}}$ that associates subsets of constants $\mathcal{C}_{PL}$ in a priority logic (PL) with the parameters of ***trust_ calculation***.

These constants are the factors that are of importance to the agent in calculating the trust evaluation. This will be formalised in Section 6.3.2, but first we give an example of a ***trust_ calculation*** function.

6.3.1 An illustrative trust model

We illustrate the parametrisation of a trust model with the following small model to evaluate agents in an auction environment. The model calculates a trust evaluation of a target agent t with respect to a goal γ, by using a weighted average to aggregate information from three different sources: the agent's own direct experiences as a buyer and as a seller with the target, specified in $Sales_t$ and $Purchases_t$, respectively, and the communicated trust evaluations from other agents in the system, specified in Φ_{Comm_t}. The ***trust_ calculation*** function of this example is therefore as follows:

$$\textbf{\textit{trust\_ calculation}}(Sales_t \cup Purchases_t, \Phi_{Comm_t}, \gamma, t) = trust(t, \text{`}\gamma\text{'}, \frac{w_{buyer} \cdot DE_{Buyer}(Sales_t) + w_{seller} \cdot DE_{Seller}(Purchases_t) + w_{reputation} \cdot Comm\_Trust(\Phi_{Comm_t})}{w_{buyer} + w_{seller} + w_{reputation}})$$

w_{buyer}, w_{seller} and $w_{reputation}$ are the three weights, and the functions DE_{Buyer}, DE_{Seller} and $Comm\_Trust$ calculate intermediate values for direct experiences with buyers, sellers and communicated evaluations, defined as follows:

$$Comm\_Trust(\Phi_{Comm_t}) = \sum_{C \in \Phi_{Comm_t}} \frac{value(C)}{|\Phi_{Comm_t}|}$$

$$DE_{Buyer}(Sales_t) = \sum_{S \in Sales_t} \frac{w_{profit} \cdot eval\_profit(S) + w_{time} \cdot eval\_paytime(S)}{|Sales_t| \cdot (w_{profit} + w_{time})}$$

$$DE_{Seller}(Purchases_t) = \sum_{P \in Purchases_t} \frac{w_{cost} \cdot eval\_cost(P) + w_{delivery} \cdot eval\_delivery(P)}{|Purchases_t| \cdot (w_{cost} + w_{delivery})}$$

$Comm\_Trust$ aggregates the communicated trust evaluations in a straightforward manner: it simply takes the communicated values as they are, and

averages these over all the received communicated evaluations. DE_{Buyer} and DE_{Seller} both use a straight up average over all interactions in which a sale, or a purchase, was made. Each interaction is evaluated by taking a weighted average of the evaluation of two aspects of the interaction. For sales interactions this is the profit made and whether the payment was on time. For purchases this is the cost incurred and whether the delivery was on time. The weights w_{profit} and w_{time} determine the importance of these two factors for DE_{Buyer}, while w_{cost} and $w_{delivery}$ do the same for DE_{Seller}. Each individual sales interaction is evaluated using the following functions:

$$eval\_profit(Sale) = max(-1, min(1, \frac{profit(Sale) - expected\_profit(Sale)}{profit\_threshold}))$$

$$eval\_paytime(Sale) = \begin{cases} -1 & \text{if } date\_payed(Sale) > payment\_deadline(Sale) \\ 1 & \text{otherwise} \end{cases}$$

$eval\_profit$ calculates the normalised value of profit. The $profit\_threshold$ is a constant: if the difference between the actual profit and expected profit is greater than the threshold, then the output of the function is capped at 1, or -1, depending on whether the difference is positive or negative. $eval\_paytime$ is a binary function: it is 1 if the price of the item was paid on time, and -1 if it was not. Two very similar functions, $eval\_cost$ and $eval\_delivery$, perform the same task for evaluating purchase interactions:

$$eval\_cost(Purchase) = max(-1, min(1, \frac{expected\_price(Purchase) - price(Purchase)}{cost\_threshold}))$$

$$eval\_delivery(Purchase) = \begin{cases} -1 & \text{if } delivery\_date(Purchase) > expected\_delivery(Purchase) \\ 1 & \text{otherwise} \end{cases}$$

This trust model serves as a first example of a model that can be incorporated into the agent's reasoning system. The key points of this demonstration are twofold: the first is to stress the intuition described so far that a trust model can be seen as a function that takes inputs from the agent's mental contexts and calculates a trust evaluation. Obviously the actual calculation of this example is quite simple; however, we demonstrate this in a similar manner for actual computational trust models in Section 6.5.

The second point is to demonstrate that the designer of the system has a choice in what to consider as parameters. In the above system it would be intuitive to take the weights as parameters. Such a choice, however, does not fit our model: the value of a single weight in a weighted average is meaningless because it is the ratio between weights that defines their relative importance within a weighted average. We thus take these ratios as our parameters. The actual value of the weights can follow trivially from these.

Note that fixing the values for all possible ratios between all weights may lead to unsatisfiable equations. We stated in Section 6.1 (on page 119) that any combination of values for the parameters should yield a valid trust model, and in our example, having parameters representing all ratios between all weights violates this requirement: the parameters would be interdependent. This prob-

lem is easily solved in our example by choosing a mutually independent set of ratios on which the remaining ratios are dependent. We thus choose the set of parameters $Params_{\textbf{\textit{trust\_calculation}}} = \{\frac{w_{profit}}{w_{time}}, \frac{w_{cost}}{w_{delivery}}, \frac{w_{buyer}}{w_{seller}}, \frac{w_{buyer}}{w_{reputation}}\}$ for the ***trust_calculation*** function. It is easy to see that any value in the range of the parameters can be chosen. While *profit_threshold* and *cost_threshold* could technically also be parameters of the calculation, they serve no purpose in adapting the trust model to the agent's behaviour. Rather they need to be chosen correctly for the trust model to be of use in any domain.

Adapting the trust model

The parameters above can take any number of values, which result in different behaviours of the trust model. The behaviour of the trust model should be linked to the agent's beliefs and goals, allowing it to adapt the model to the current situation. To do this, the values of the parameters should depend on certain factors – the labels of the parameters – that the agent can prioritise over. For instance, if in the example the agent regards the delivery time of an item as more important than its cost, then the parameter $\frac{w_{cost}}{w_{delivery}}$ should be smaller than 1, resulting in the corresponding weights $w_{delivery}$ and w_{cost} in the trust calculation reflecting that delivery time is more important than cost.

We define this more generally, with the set of constants $\mathcal{C}_{PL}$, which can be used in a priority logic (PL), as defined in Section 6.2.2. In the above example we choose: $\mathcal{C}_{PL} = \{delivery\_time, cost, payment\_time, profit, outcome\_buying, outcome\_selling, reputation\}$, which corresponds exactly with the variables in the equations of the trust model. These are the factors that the agent uses to describe interactions in which it buys and sells items in an auction, as well as the factors describing the intermediate stages of the trust calculation. Consequently, changes in the relative importance between these factors should cause the trust model to change. In a real scenario far more factors can influence the parameters, but we aim to keep the example simple.

The different parameters of the trust model are not all influenced by the same factors. As specified in Definition 6.2, the *labels* function defines which factors influence which parameter. In our example we have:

- $labels(\frac{w_{profit}}{w_{time}}) = \{payment\_time, profit\}$
- $labels(\frac{w_{cost}}{w_{delivery}}) = \{delivery\_time, cost\}$
- $labels(\frac{w_{buyer}}{w_{seller}}) = \{outcome\_buying, outcome\_selling\}$
- $labels(\frac{w_{buyer}}{w_{reputation}}) = \{outcome\_buying, reputation\}$

In the continuation of this chapter, we describe a method that allows an agent to reason about the labels affecting its trust model. Viewing the trust model as a function allows us to abstract away from the actual computation and give this specification in more general terms than would be the case if we had to consider each method of computation separately.

6.3.2 A priority system

Our language must allow for the specification of the importance of the different factors that are taken into account in a trust calculation. Another thing to note is that these factors may be both the initial inputs, such as communicated information or direct experiences, as well as internal predicates, such as the concepts of image and reputation in the case of Repage [Sabater et al., 2006]. Some comparisons, however, make no sense. For example, in the case of Repage it is senseless to specify that direct experiences are more important than image, because nowhere in the algorithmic process do the two concepts occur together. In fact, image is considered as the output of an aggregation of, among other things, direct experiences. We see that we do not need to specify the importance ordering over all possible factors due to the algorithmic design of the trust model making some comparisons are pointless.

This notwithstanding, we do need a way of identifying those factors that require ordering. For this we turn to the parameters of the trust model. We only need to define the importance between any factors that appear together in the labels related to a parameter. In our example, for instance, there is no need to define the importance between the factors *payment_time* and *reputation*, because there is no parameter that is influenced by the relative importance between these two factors.

The parameters and their labels give us a natural way to specify what is important in the calculation of trust. *How* each of these labels influences the trust evaluation is dependent on the calculation itself. Take for instance, our example of a weighted average: the higher the ratio $\frac{w_{profit}}{w_{time}}$, the more weight is given to the profit made in a sale, in comparison to the punctuality of the payment. This is a natural translation from the relative importance of profit as opposed to punctuality. We do not elaborate on how exactly this translation is implemented. For instance, as long as the parameter $\frac{w_{profit}}{w_{time}}$ is given a value greater than 1, it complies with the idea that profit is more important than delivery time, independent of the actual value it has. Nevertheless, the actual value may still be important: the values for this parameter influence the trust model in very different ways.

We will assume the agent has a way of translating the relative importance into actual values for the parameters in a reasonable manner. One way to choose the values is through the relative importance of the priority rules, which we will discuss in Section 6.4, but first we need to finish formalising the idea of relative importance between the various factors. We do this with our priority logic (see Section 6.2.2), which allows us to specify a set of priorities, for each of the parameters, over the labels which influence that parameter's value. Such a set of priorities is called a priority system.

Definition 6.3 (Priority System)**.**
The *priority system* context PSC contains a set of priorities for each parameter of the trust model. We recall that a set of priorities is a theory in a PL-language $\mathcal{L}_{PL}$. Let ***trust_calculation*** be a trust model with parameters $Params_{\textbf{\textit{trust\_calculation}}}$, then for each parameter $p \in Params_{\textbf{\textit{trust\_calculation}}}$

we define the PL-language $\mathcal{L}_{PL_p}$ with predicates $\succ$ and $=$ and constants $\mathcal{C}_{PL_p} = labels(p)$. A theory in $\mathcal{L}_{PL_p}$ is denoted as Π_p. The PSC therefore contains the indexed family $\{\Pi_p\}_{p \in Params_{\textbf{trust\_calculation}}}$.

For instance, in our auction example, we could have the following PSC:

$$\{cost \succ \ delivery\_time\}_{\frac{w_{cost}}{w_{delivery}}}$$
$$\{profit \succ \ payment\_delay\}_{\frac{w_{profit}}{w_{time}}}$$
$$\{outcome\_buying = \ outcome\_selling\}_{\frac{w_{buyer}}{w_{seller}}}$$
$$\{outcome\_buying \succ \ reputation\}_{\frac{w_{buyer}}{w_{reputation}}}$$

Using these priorities means that any actual instantiation, of the example trust model in Section 6.3.1, must give more importance to cost than delivery time, profit to payment delays, and outcomes from either type of direct experience to reputation, when evaluating an agent.

Now we recall that trust is multifaceted: the priority system is dependent on what goal the trust evaluation serves. For instance the priorities may differ, depending on whether the agent intends to buy or sell an item. As such the priorities must be dependent on the goal. We call any goal, whose achievement depends on other agents, a *socially-dependent goal*.

6.3.3 Socially-dependent goals

We recall from Section 6.2 that a goal of an agent is an FODL sentence that combines a desired outcome with a plan. Some goals cannot be achieved by the agent working in isolation. We call a goal which requires interaction with another agent a socially-dependent goal. Different plans, however, may require different interactions to fulfil the same desire, so the need to interact must come from the specific manner in which to achieve the goal. Some desires may be fulfilled in strict isolation, but have alternative methods for fulfilment if interaction is considered. Other desires may inherently require interaction. Either way, it is the plan in which the interaction is defined. Such a plan is formed in the Planner Context and is a sequence of actions. We should, therefore, specify in the actions, whether there is a need for interaction. Furthermore, we should define the type of action, or set of actions, we expect the other agent to perform (such as buying or selling an item). For this we will use an abstraction, so that rather than specifying the actual actions, we specify which *role* the agent should fulfil. We will assume such roles are defined in the MAS itself [Odell et al., 2003; Hermoso et al., 2009], and the agent knows what can be expected from an agent performing any given role. The agent, therefore, has knowledge about a set of roles which can be performed in the system, using the special unary predicate $role \in \mathcal{P}_{Bel}$. We define the set $Roles$ analogously to $Agents$: $Roles = \{r \mid BC : role(r)\}$. These roles are used in the definition of a social action:

Definition 6.4 (Social action)**.** We define the planning predicate $social\_action(\text{`}\alpha\text{'}, \text{`}\Phi_{Pre}\text{'}, \text{`}\Phi_{Post}\text{'}, r)$ in an analog manner to how we defined $basic\_action$ in Section 6.2.3 (on page 124). Thus $\alpha \in RG$ is a basic action, and $\Phi_{Pre}, \Phi_{Post} \subseteq \mathcal{L}_{Bel}$ are the pre- and post-condition, respectively. The distinction between a $basic\_action$ and a $social\_action$ is that the latter requires that some other agent, performing role $r \in Roles$, participates in the action. With this extension of the syntax of the planning context's internal logic, we also need to extend the definition of the planning predicate: $plan(\text{`}\alpha\text{'}, \text{`}\Phi_{Pre}\text{'}, \text{`}\Phi_{Post}\text{'}, R)$, where $R \subseteq Roles$ is the set of roles required by social actions in the program α.

Note that this is a simplified version of a more general definition, in which an action might require the participation of multiple agents performing multiple roles. This entire framework can be extended in a trivial manner to such multi-agent social actions, but to keep the notation clean we limit the actions to two agents and thus a single other agent performing a single role. A socially-dependent goal is defined as any goal $[\alpha]\psi$ where α is a plan in which at least one social action is involved. For convenience we explicitly write the roles involved in the goal as $[\alpha]\psi\langle R\rangle$, as shorthand for $PC : plan(\alpha, Pre, Post, R) \wedge IC : [\alpha]\psi$. In other words R is the set of roles required by the social actions in the plan. We also write $[\alpha]\psi\langle r\rangle$ to denote any role $r \in R$ with the above condition. Whether an agent performs a certain role adequately is directly linked to its trustworthiness.

From an organisational perspective of the MAS we consider all agents inherently capable of performing any role. Whether they perform this role adequately is a matter for the individual agents to decide. It is also in this aspect that our multifaceted view of trustworthiness comes into play, as trust is based not just on agents' willingness to perform an action, but also their capability in performing this action [Castelfranchi and Falcone, 2010]. When an agent wishes to accomplish some socially-dependent goal, it must attempt to find the agents best able to fulfil the required roles, using the corresponding priority system to instantiate the trust model to perform this evaluation.

6.4 Goal-based Instantiation of Trust Models

In the previous section we showed how to abstract away from a computational model and specify the factors that influence the trust computation. We can now tie this together with the BDI-reasoning of an agent. Particularly, we specify how the beliefs and goals affect the trust model. The factors that influence the trust model are defined in the priority system, but which PL-theory is used depends on the agent's reasoning system. The socially-dependent goal the agent wants to accomplish, the set of roles required to achieve this goal, and the beliefs an agent has about its environment may all influence the importance of the different factors, and thus the trust model itself. A priority system, as described in Definition 6.3, is thus not a static set of rules, but is defined dynamically for each goal an agent intends to achieve: given a set of beliefs Φ, for each goal γ for which role $r \in R$ is required, the agent has a PL-theory $PS_{r,\gamma,\Phi}$ describing the

priorities at that moment. We need to specify how priorities come into existence and how they are used in instantiating a trust model for a specific goal. We also need to specify how inconsistencies in a PL-theory are resolved, so that the parameters of the trust model can be instantiated. We return to the question of resolving inconsistencies at the end of Section 6.4.1. First, we show how the trust model and priorities are involved in the reasoning system of the agent and how the agent's beliefs and desires can cause the adaptation of the trust model. We use the MCS described in Section 6.2 and specify the bridge rules required.

6.4.1 Reasoning about the priority system

We work from the intuition that an agent's beliefs, and the goal it is attempting to achieve, *justify* the priorities. We illustrate this by taking another look at the example of Section 6.3.1: if an agent believes it is beneficial to be frugal, it could choose to prioritise cost over delivery time and profit over payment delay. Nevertheless, there might be a specific goal, such as if an agent needs to buy an item urgently, that could change these priorities: if the agent wants the item delivered quickly, it could prioritise delivery time over cost. Notice that such a change possibly has the effect of changing the agent's decision to select one provider or another: this decision is based on the trust model, which in turn is dependent on the prioritisation of the aforementioned criteria.

To allow an agent to adapt its priorities — and thus its trust model — in such a manner, we specify rules encoding causal relationships between cognitive elements and the priority system, formalised using a first-order logic restricted to Horn clauses. These rules are deduced in the priority rule context (PRC); a functional context similar to the planner context. We have three special priority rule predicates: $belief\_rule, role\_rule$ and $goal\_rule$. These predicates specify that a set of beliefs, a role or a goal support a certain ordering of priorities. Additionally, these predicates have a third argument, specifying the preference value for that rule. This is a numerical indicator of how important that rule is, which is used in the *resolve* function below. Let $\Phi \subset \mathcal{L}_{Bel}$, $r \in Roles$ and $\gamma \in \mathcal{L}_{Int}$. Additionally let $v \in \mathbb{N}$ and $\pi \in \mathcal{L}_{PL_p}$ for some $p \in Params_{\textbf{trust\_calculation}}$, then the following are priority rule predicates:

- $belief\_rule(`\Phi', `\pi', v)$. This allows for the definition of *priority rules* stating that the set of beliefs Φ supports the priority π with preference v.
- $role\_rule(r, `\pi', v)$. The use is similar to the previous rule, but with a role, rather than beliefs, supporting a priority.
- $goal\_rule(`\gamma', `\pi', v)$. A priority rule stating that goal γ supports priority π with preference v.

Additionally, we have the predicates $belief, role$ and $goal$ to represent the agent's mental attitudes in the PRC, which are needed in the deductions. To see how this works in practice, we once again consider the example and formalise the scenario from the start of this section: the agent has the belief that it is frugal.

This belief supports prioritising cost over delivery time. The formalisation of this is with the priority rule $belief\_rule(\text{`}frugal(me)\text{'}, \text{`}cost \succ delivery\_time\text{'}, 1)$.

How an agent obtains such priority rules is dependent on the implementation. Similar to the planning context, such rules can be predefined by the programmer, or the agent can be equipped with some reasoning system to deduce them. We use Υ to denote the set of all priority rules an agent has. The rules are *triggered* by a specific set of beliefs, a goal or a role, for which the agent needs to evaluate other agents' trust. As such, we need bridge rules to connect the priority rule context to the belief and intention contexts:

$$\frac{BC : \Phi}{PRC : belief(\text{`}\Phi\text{'})} \qquad \frac{BC : role(\rho)}{PRC : role(\rho)} \qquad \frac{IC : [\alpha]\psi}{PRC : goal(\text{`}[\alpha]\psi\text{'})} \tag{6.6}$$

Let $p \in Params_{\textbf{\textit{trust\_calculation}}}$ be a parameter of the trust calculation, with its corresponding set of labels. Given a set of beliefs Φ, a role r and a goal γ we can now use the PRC to find a set of priorities $\Pi_{p,r,\gamma,\Phi}$ over the set $labels(p)$:

$$\begin{aligned}\Pi_{p,r,\gamma,\Phi} \;=\; & \bigcup_{\Phi' \subseteq \Phi} \{\pi | belief\_rule(\text{`}\Phi'\text{'}, \text{`}\pi\text{'}, v) \in \Upsilon \wedge terms(\pi) \subseteq labels(p)\} \\ & \cup \{\pi | role\_rule(r, \text{`}\pi\text{'}, v) \in \Upsilon \wedge terms(\pi) \subseteq labels(p)\} \\ & \cup \{\pi | goal\_rule(\text{`}\gamma\text{'}, \text{`}\pi\text{'}, v) \in \Upsilon \wedge terms(\pi) \subseteq labels(p)\}\end{aligned}$$

Note that while $\Pi_{p,r,\gamma,\Phi}$ is a wff in PL, it may be inconsistent. We illustrate this by continuing the formalisation of the example. The agent in the scenario has the goal to buy an item, for instance a book, urgently. This can be formalised in the rule $goal\_rule(\text{`}[buy\_book]have\_book(tomorrow)\text{'}, (delivery\_time \succ cost), 2)$. The set generated with the singleton set of beliefs $\{frugal(me)\}$ and goal $[buy\_book]have\_book(tomorrow)$ contains both $delivery\_time \succ cost$ as well as $cost \succ delivery\_time$ and is thus inconsistent.

We see that, to obtain usable sets of priorities, the agent must perform some form of conflict resolution that guarantees a consistent PL-theory. This is done using *resolve*: a function that, when given a set of priority rules, a set of beliefs, a goal, a role and a trust model, returns a consistent PL-theory. It is immediately obvious that there are many different ways of obtaining a consistent PL-theory from an inconsistent set of priorities and there are intuitively better, and worse, ways of doing this. For instance, $\emptyset$ is a consistent PL-theory, but it is a thoroughly useless one. To maximise the utility of the obtained PL-theory we use the preference factors for each of the rules in Υ. The output of the *resolve* function must fulfil the following two conditions: (1) it is a consistent PL-theory and (2) the total preference value of the rules used is maximal. This allows the agent designer a lot of freedom in defining how to resolve the priorities. For instance, one way would be to guarantee that *resolve* returns a maximal subset of $\Pi_{p,r,\gamma,\Phi}$ that is a consistent PL-theory. Another possibility ties in with the idea that the trust model should be goal-orientated. In such a case the priorities supported by the goal should supersede the priorities supported by the role, which in their

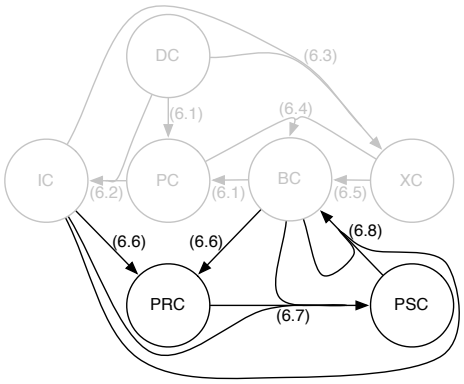


Figure 6.2: The MCS specification of a BDI-agent that can reason about trust. This is an extension of Figure 6.1 (on page 126), whose bridge rules are colored grey. The bridge rules added for reasoning about trust are black, with labels corresponding to those in the text.

turn should supersede those supported by the beliefs. Both of these methods can be expressed by choosing adequate preference factors for the rules: if we just want the maximal subset, all preferences can be 1, but if the implementation calls for some distinction among the rules, this can be implemented using the preferences of the rules. In our example we have chosen to distinguish among rules. Specifically, we have chosen to make the goal rule more important than the belief rule. In Section 6.5.1 we will give more examples of priority rules and the preference values. Note that, while we have not placed an upper bound on the preference values, in practice such a limitation is useful. Especially if, as we will explore in Chapter 7, priority rules can be exchanged between agents, the preference values must have a similar range to be acceptable. In Chapter 7 we propose to use $[1, 10]$, but a designer of a MAS is free to decide differently.

We have now defined all the individual parts of the extended BDI model and can define the first bridge rule related to reasoning about trust, namely the one for generating a priority system:

$$\frac{BC : \Phi \qquad IC : \gamma = [\alpha]\psi\langle r\rangle \qquad PRC : \Upsilon}{PSC : PS_{r,\gamma,\Phi}} \tag{6.7}$$

with $PS_{r,\gamma,\Phi} = \{resolve(\Pi_{p,r,\gamma,\Phi}, \Phi, \gamma, r, \boldsymbol{trust\_calculation})\}_{p \in Params_{\boldsymbol{trust\_calculation}}}$ being the outcome of an operational procedure, as alluded to in Section 6.2.1.

It is up to the implementation of the agent to define *resolve* as well as the rules in Υ (or a method for generating them).

6.4.2 Instantiating trust models

A socially-dependent goal γ is not immediately executable, unlike a goal with a plan which contains no social actions. To achieve a social action it is insufficient to define which role an agent must fulfil, but the agent must actually choose another agent to fulfil that role. For this, the trust evaluation of other agents in the system must be calculated, so that a decision can be made whom to interact with. As stipulated earlier, we require that there is a method to obtain an instantiation of a trust model that complies with a priority system. Accordingly, the agent can use such an instantiation to calculate its trust evaluations. A system implementing our model has to provide a definition of the *instantiate* function that, when given a priority system $PS_{r,\gamma,\Phi}$ and a trust function $\boldsymbol{trust\_calculation}$, outputs another trust function $\boldsymbol{trust\_calculation}_{r,\gamma,\Phi}$, such that $instantiate(PS_{r,\gamma,\Phi}, \boldsymbol{trust\_calculation})$ complies with the priority system $PS_{r,\gamma,\Phi}$. A trust model complies with a priority system if the value of each of its parameters complies with the priorities over its labels, as explained in Section 6.3.2. Both *instantiate* and how the compliance of the resulting trust function is checked depend heavily on the actual trust model used, and we will demonstrate this in an example in Section 6.5.1.

Now this $\boldsymbol{trust\_calculation}_{r,\gamma,\Phi}$ can be used to aid the agent in selecting a partner for the required role. Trust, however, may not be the only thing an agent uses to select a partner. In a BDI-based agent architecture all reasons an agent may have to select a partner are usually inferred from the belief base. Therefore, trust evaluations should be added to the belief base, allowing them to be incorporated in this reasoning process. Once a partner has been selected, a social action may be executed just as a basic action can be executed by the agent. The details of how this happens is outside the scope of a high-level specification, but we do need to specify how trust evaluations get added to the belief base. $\boldsymbol{trust\_calculation}_{r,\gamma,\Phi}$ is a functional representation of an underlying computational trust model. This trust model can calculate the trust evaluation of prospective partners for the achievement of the goal γ. These calculations need to be added in the belief base, and accordingly, we require a bridge rule for this:

$$\frac{IC : \gamma = [\alpha]\psi\langle r\rangle \qquad BC : \Phi \qquad PSC : PS_{r,\gamma,\Phi}}{BC : \varphi_{\gamma,r,a}} \tag{6.8}$$

where $\varphi_{\gamma,r,a} = \boldsymbol{trust\_calculation}_{r,\gamma,\Phi}(\Phi_{DE}, \Phi_{Comm}, \gamma, a)$ with Φ_{DE}, $\Phi_{Comm} \subset \Phi$ and $a \in Agents$.

Finally, this trust evaluation can be used in the execution of a plan, by selecting the best partner for interaction. The MCS with the two new contexts and the additional bridge rules is represented schematically in Figure 6.2.

6.5 Integrating Trust Models

With the extended BDI-framework in place we can now show how this allows an agent to reason about its trust model. We will demonstrate how to incorporate three different trust models into AdapTrust: BRS [Jøsang and Ismail, 2002], ForTrust [Lorini and Demolombe, 2008] and ReGReT [Sabater, 2003]. We show how incorporating a trust model into AdapTrust allows the agent to proactively change its trust evaluation, in addition to allowing an easy and intuitive way of allowing any model to deal with the multifaceted aspect of trust by use of the agent's goals and the roles other agents may perform. We only demonstrate the entire reasoning model for BRS, while presenting a detailed discussion of the algorithmic representation of the other models and their parameters.

6.5.1 BRS

The Beta Reputation System (BRS) is a statistical method for calculating reputation: the aggregation of other agents' trust evaluations. The approach described by Jøsang and Ismail [2002] is a centralised approach and can thus be seen as a model which does not take individuals' own direct experiences into account separately. They explicitly mention, however, that this same method can be used in a decentralised approach, and newer extensions of this model, such as TRAVOS [Teacy et al., 2006], are distributed. Nevertheless, their basis is the same statistical model as BRS uses. In BRS, an agent's own direct experiences are aggregated in just the same way as any other agent's communicated experiences — with no special preference. BRS is based on the beta-family of probability distributions of binary events. This family is expressed by the following probability density function, where p is a probability conditioned by the shape parameters α and β and Γ is the Gamma function[2]:

$$f(p|\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\cdot\Gamma(\beta)} \cdot p^{\alpha-1} \cdot (1-p)^{\beta-1}, \text{ where } 0 \leq p \leq 1,\ \alpha > 0,\ \beta > 0$$

The expected value, given such a probability density function is:

$$E(p) = \frac{\alpha}{\alpha+\beta}$$

The expected outcome is thus a real value in the range $(0, 1)$. BRS uses this expectation as the reputation, but converts it to a value in $(-1, 1)$, which they state is more intuitive to human users. The α and β are determined by the number of "good" and "bad" evaluations of interactions with a certain target. Jøsang and Ismail add further refinements by discounting opinions from agents who are uncertain and discounting experiences over time. Additionally, they provide a method for using non-binary evaluations of an interaction by treating a single interaction as a number of interactions with a mix of "good" and "bad". A numerical evaluation of a single interaction can be represented as a pair (r, s)

[2] $\Gamma(x) = \int_0^\infty t^{x-1} \cdot e^{-t}\mathrm{d}t$

where r is the value of "good" and s the value of "bad" in that single interaction. It is necessary to keep $r + s$ constant, but in this way we can add r good and s bad evaluations for each interaction, representing non-binary evaluations. By using a weight w it is further possible to give important interactions a higher value for r and s and thus more importance in the determination of α and β, although this is not explored further by Jøsang and Ismail [2002]. Algorithm 4 describes the entire process.

Algorithm 4: Centralised BRS

Input: $Evals = \{(v, a, t)\}$, a set of evaluations. Each evaluation with value $v \in [-1, 1]$, evaluator agent a and time t
Input: $j \in Agents$, the agent to be evaluated
Input: w, the weight parameter
Input: λ, a parameter for decay over time
$s := 0$
$r := 0$
foreach $(v, a, t) \in Evals$ **do**
  $r := r + \lambda^{now-t} \cdot w \cdot (1 + v)/2$
  $s := s + \lambda^{now-t} \cdot w \cdot (1 - v)/2$
end
reputation $:= \frac{r-s}{r+s+2}$
Output: $Reputation(j, \text{reputation}, now)$

Parametrisation of centralised BRS

When looking at the algorithm we see that there are two explicit parameters, w and λ. λ can be interpreted as defining the balance between new and old evidence: if $\lambda = 1$ then the time since the evidence was observed is irrelevant; if $\lambda < 1$ then historical evidence is given less importance; and the unusual situation of $\lambda > 1$ would mean that the longer ago the evidence was observed, the more importance it is given. However, w in the formula given by Jøsang and Ismail only influences the convergence rate. We feel the weight w can be put to better use: as they themselves describe, it can be used to define the *importance* of an interaction to the final calculation. This ties in directly to another issue with the algorithm: it starts off with a set of evaluations without describing how these are calculated from an agent's beliefs. Thus the algorithm above only describes the *aggregation* method used for calculating trust based on individual evaluations, but not the evaluation method itself, which forms an important part of the evaluation. This is mainly because the article describes a *centralised* method in which it uses communicated information: the computational entity that performs the aggregation is not the same one that performs the evaluation of individual interactions. We thus propose a decentralised version of BRS defined in Algorithm 5, which is a procedure for calculating the outcome of the

trust_ calculation function for an agent, using BRS as its trust and reputation model. This uses three secondary functions: **eval**, **weight** and **time_stamp** to calculate the value of a direct experience, the weight assigned to this value by the agent, and the time the direct experience took place, respectively. Because the agent's knowledge of its direct experiences and received communications are collected in its belief base, we make the link to the belief context BC explicit in the algorithm.

Algorithm 5: Decentralised BRS

Input: $j \in Agents$, the agent to be evaluated
Input: $BC : I_j$, a set of direct experiences with agent j
Input: $BC : R_j = \{received(reputation, i, j, v, t)\}$, a set of messages with sender i, communicating that agent j has reputation value v at time t
Input: λ, a parameter for decay over time
$s := 0$
$r := 0$
foreach $i \in I_j$ **do**
  $v := \textbf{eval}(i)$
  $w := \textbf{weight}(i)$
  $t := \textbf{time\_stamp}(i)$
  $r := r + \lambda^{now-t} \cdot w \cdot v$
  $s := s + \lambda^{now-t} \cdot w \cdot (1 - v)$
end
foreach $received(reputation, i, j, v, t) \in R_j$ **do**
  $Rep_i := (1 + rv)/2$, where $BC : Reputation(i, rv, now - 1)$
  $r := r + \lambda^{now-t} \cdot Rep_i \cdot v$
  $s := s + \lambda^{now-t} \cdot Rep_i \cdot (1 - v)$
end
reputation $:= \frac{r-s}{r+s+2}$
Output: $Reputation(j, \text{reputation}, now)$

This algorithm does not strictly follow the description of BRS [Jøsang and Ismail, 2002]. To make sense in a decentralised setting a distinction must be made between the own and others' experiences. This distinction need not be made in a centralised system, and therefore we propose to discount incoming communications based on the reputation of the sender. This discounting is not done in the original article, but we feel it represents an accurate extension of BRS to a decentralised model. The own experiences are evaluated using the functions **eval** and **weight**. These can easily be parametrised to be dependent on the goal the agent is attempting to achieve.

As mentioned earlier, TRAVOS [Teacy et al., 2006] is an example of a newer trust model that extends BRS, and it makes a similar distinction between own direct experiences and incoming communications as in Algorithm 5. Neverthe-

less, it extends the BRS model significantly, mainly in how it chooses what information to take into account when it calculates a trust evaluation. BRS uses all information available, although weights can influence this. TRAVOS, however, uses further calculations of the confidence in different sources to possibly disregard information.

Parametrisation of decentralised BRS

The only explicit parameter is the same one as in the centralised model, the decay factor for time. There is, however, also an implicit one: while the communicated evaluations always have a weight of at most 1, the weight of own experiences may be arbitrarily high. This leads to an implicit weighing of the own experiences as compared to the communicated evaluations with ratio $w_{direct} = max_{i \in I_j}(\textbf{weight}(i))$. This implict weight can be made explicit by adding the constant $w_{reputation}$, that is multiplied with all the communicated evaluations (in addition to the reputation of the sender). The parameter $\frac{w_{direct}}{w_{reputation}}$ describes the relative influence of direct experiences and communicated evaluations, thereby allowing the agent to modify this parameter and change this balance. The **eval** function may also be parametrised, similar to the other models. Next, we explain this parametrisation in more detail.

An e-commerce agent using decentralised BRS

We will now show how this system allows an agent to proactively change its trust model, using Algorithm 5 as the procedure for calculating trust. The implementation depends on the domain in which the agent is operating, so to keep our example compact, we use the same domain as in the example of Section 6.3.1: an agent in an e-commerce environment. The environment has two roles an agent may play, buyer or seller, corresponding to the actions an agent can perform, buying an item or selling an item. To evaluate agents' direct interactions we use evaluation functions similar to those described in Section 6.3.1, but returning the outcome in a format that BRS can handle. BRS performs the aggregation, so we just need to provide a definition for the functions **eval** and **weight**. BRS does not give any description of how these should be defined and it is up to the designer. We choose the following definitions, for the different types of interactions, because they are simple calculations with parameters that we can adjust using AdapTrust:

$$\textbf{eval}(DE_{Sale}) = round(w_{profit} \cdot eval\_profit(DE_{Sale}) + w_{time} \cdot eval\_paytime(DE_{Sale}))$$
$$\textbf{eval}(DE_{Purchase}) = round(w_{cost} \cdot eval\_cost(DE_{Purchase}) + w_{delivery} \cdot eval\_delivery(DE_{Purchase}))$$
$$\textbf{weight}(DE_{Sale}) = w_{sale}$$
$$\textbf{weight}(DE_{Purchase}) = w_{purchase}$$

which use the functions:

$$eval\_profit(Sale) = max(-1, min(1, \frac{profit(Sale) - expected\_profit(Sale)}{profit\_threshold}))$$

$$eval\_paytime(Sale) = \begin{cases} -1 & \text{if } date\_payed(Sale) > payment\_deadline(Sale) \\ 1 & \text{otherwise} \end{cases}$$

$$eval\_cost(Purchase) = max(-1, min(1, \frac{expected\_price(Purchase) - price(Purchase)}{cost\_threshold}))$$

$$eval\_delivery(Purchase) = \begin{cases} -1 & \text{if } delivery\_date(Purchase) > expected\_delivery(Purchase) \\ 1 & \text{otherwise} \end{cases}$$

Having now specified all the calculations of the ***trust_ calculation*** function, we choose the following set of parameters: $Params_{\textbf{\textit{trust\_ calculation}}} = \{\lambda, \frac{w_{profit}}{w_{time}}, \frac{w_{cost}}{w_{delivery}}, \frac{w_{sale}}{w_{purchase}}, \frac{w_{sale}}{w_{reputation}}\}$, giving a full parametrisation of BRS. The last four parameters are described in Section 6.3.1, while λ is a parameter from Algorithm 5. We demonstrate the agent's reasoning using a simple set of labels. These labels are constants that allow us to capture the meaning of the parameter. The numerical value of a parameter is thus symbolically represented as an ordering of factors that influence the parameters:

- $labels(\lambda) = \{old, new\}$
- $labels(\frac{w_{profit}}{w_{time}}) = \{payment\_time, price\}$
- $labels(\frac{w_{cost}}{w_{delivery}}) = \{delivery\_time, cost\}$
- $labels(\frac{w_{sale}}{w_{purchase}}) = \{sale\_experience, purchase\_experience\}$
- $labels(\frac{w_{sale}}{w_{reputation}}) = \{sale\_experience, reputation\}$

The ordering of these labels are defined in a priority system, which is deduced in the PRC. The following are a sample of rules an agent may use to reason in this example.

The rules for deducing the PL for parameter λ are:

- $belief\_rule(`dynamism(environment, high)', `new \succ old', 1)$
- $belief\_rule(`dynamism(environment, none)', `new = old', 1)$

These state that if the environment is very dynamic, old information should be given less importance than new, whereas if the environment is not dynamic, all information should be treated equally.

For $\frac{w_{profit}}{w_{time}}$ the rules are:

- $belief\_rule(`frugal(me)', `price \succ payment\_time', 1)$
- $belief\_rule(`[pay(rent, landlord)]have(home, next\_month) \wedge \neg have(rent, now)', `payment\_time \succ price', 5)$
- $goal\_rule(`[sell(item)]have(money, future)', `price \succ payment\_time', 1)$

With the meaning that if the agent is frugal, price is given more importance than the deadline for payment. A similar rule applies if the agent does not have a goal to sell an item urgently. On the other hand, if the agent needs money urgently, for instance to pay the rent, then it should give a high importance to the time the payment is made, rather than the profit made.

For $\frac{w_{cost}}{w_{delivery}}$:

- $belief\_rule(\text{`}frugal(me)\text{'}, \text{`}cost \succ delivery\_time\text{'}, 1)$
- $goal\_rule(\text{`}[buy(item)]have(item, tomorrow)\text{'}, \text{`}delivery\_time \succ cost\text{'}, 2)$

Similar to the rules for $\frac{w_{profit}}{w_{time}}$, a frugal agent should prioritise the cost over timeliness of the delivery. If, however, the item is needed urgently, for instance the next day, then the delivery time is more important than the cost.

For $\frac{w_{sale}}{w_{purchase}}$:

- $role\_rule(seller, \text{`}purchase\_experience \succ sale\_experience\text{'}, 2)$
- $role\_rule(buyer, \text{`}sale\_experience \succ purchase\_experience\text{'}, 2)$

These rules define the importance in the aggregation process for direct experiences of different types. If the agent is searching for a seller, then it should give more importance to direct experiences in which it was buying items than those in which it was selling. Vice versa if the agent is searching for a buyer.

Finally for $\frac{w_{sale}}{w_{reputation}}$:

- $belief\_rule(\top, \text{`}sale\_experience \succ reputation\text{'}, 1)$
- $belief\_rule(\text{`}\forall x \in Agents : good\_reputation\_source(x)\text{'}, \text{`}sale\_experience = reputation\text{'}, 2)$

The first rule states that the default is for sale experiences to be more important than reputation. Nevertheless, if all agents in the system are good reputation sources then the two types of information should be given equal importance.

These rules are triggered by the first parameter (the antecedent) being true in its respective context. The bridge rules (6.6) cause these true sentences to be added in the PRC, where the internal reasoning checks which priority rules hold true at any one time. For bridge rule (6.7) to be of any effect, we need to define a *resolve* function. An example of a *resolve* function that resolves possible conflicts in the priority system could be to perform a best first search for each parameter, which removes rules from the set of applicable rules recursively, based on their preference value (removing the lowest values first) until the set of rules results in a consistent PL-theory. The priority system to be used is the family of the PL-theories for each individual parameter.

In our example case we can use a simpler algorithm for *resolve* because each parameter is only influenced by two factors. We thus have that, for any two factors a, b a consistent set of priorities states that either $a \succ b$, $b \succ a$ or $a = b$ (or none of them, but then there is no inconsistency). We sum the preference

values for the priority rules supporting any of the three consistent cases, and the one with the highest total value wins. If all three cases are tied, then $a = b$. In the case of a tie between any two cases, we need to break the tie in some manner. For instance, we could choose at random, or use the number of rules supporting each case. Our choice is to use the priority supported by the single rule with the highest preference as a tiebreaker, with as rationale that the weights accurately reflect the importance for having the priority, and one important rule should be chosen over multiple less important ones. If it is still tied, we choose one of the tied cases at random.

To obtain a trust model complying with these parameters, we also need to define the *instantiate* function. We give an example of a straightforward manner for obtaining such a function, although this, once again, depends on parameters being influenced by only two factors. For the first parameter, λ, the value is determined simply by discerning the two cases: if $new = old$ then $\lambda = 1$, otherwise $\lambda = 0.9$ (which we use as a default decay rate).

For the other parameters, we use the preference values of the priority rules that are triggered by the agent's requirements. We demonstrate the function for the parameter $\frac{w_{profit}}{w_{time}}$. Let us assume the agent requires trust evaluations for a role r, goal γ and beliefs Φ, and that we have $\pi = resolve(\Pi_{\frac{w_{profit}}{w_{time}}}, \Phi, \gamma, r, \textbf{\textit{trust\_ calculation}})$.

We define *instantiate* in the following manner: if $\pi \equiv (price = payment\_time)$ then $\frac{w_{profit}}{w_{time}} = 1$, otherwise we depend on the preference values of the priority rules. We define S^{profit} as the sum of the preference values of the priority rules in Υ, whose conclusion is $price \succ payment\_time$ and that are triggered by role r, goal γ, or a subset of the beliefs Φ. We define S^{time} analogously, but for the priority rules with $payment\_time \succ price$ in the conclusion. We now calculate the ratio $\frac{w_{profit}}{w_{time}} = \frac{1+S^{profit}}{1+S^{time}}$, where the 1 is added in both the numerator and denominator to avoid division by 0.

There is a particular case in which this calculation does not reflect the priority π, which corresponds to when we need to use a tiebreaker in the *resolve* function: if $S^{price} = S^{profit}$, but $\pi \not\equiv (price = payment\_time)$, then we want $\frac{w_{profit}}{w_{time}}$ near 1, but still reflecting the priority π. In this case we choose an $\varepsilon > 0$ and use $\frac{w_{profit}}{w_{time}} = \frac{1+\varepsilon}{1}$ if $price \succ payment\_time$, or $\frac{w_{profit}}{w_{time}} = \frac{1}{1+\varepsilon}$ otherwise.

A similar function exists for the other weight ratios. Note that the trust model does not use the weight ratios, but needs to be instantiated with actual weights. We choose a simple algorithm for instantiating these weights, using the ratios. The algorithm starts by assigning the value 1 to each weight. Then, for each parameter, it adjusts one of the weights so that the ratio in the parameter is correct (in other words, if $\frac{w_{profit}}{w_{time}}$ has value 1/2 it could adjust w_{profit} to 0.5 or w_{time} to 2). The algorithm always starts by adjusting a weight that has value 1, before adjusting other weights. This loop continues until all parameters are satisfied.

Now we see that this allows the agent to instantiate different trust models with different weights, depending on the role the other agent plays in the interaction, if there is urgency in selling off stock, or the agent believes all other agents

to be truthful in their reputation assessments. The agent designer could create more priority rules (or implement a system in which the agent learns priority rules) to cover more cases, similar to how agent's plans are required to allow an agent more flexibility in fulfilling its desires.

We will not discuss the integration of ForTrust or ReGReT in such a detailed example, but rather show how those algorithms can be parametrised. The further steps to full integration into the extended BDI model are left to the engaged reader.

6.5.2 ForTrust

ForTrust is a logical framework for describing properties of trust: it provides a definition of trust in the target's action [Lorini and Demolombe, 2008] and states that an agent trusts a target agent to perform an action α, which will fulfil a desire φ, if the following conditions hold:

Power: the target agent can ensure φ, by doing α

Capability: the target agent is able to perform action α

Intention: the target agent intends to do α

Trust is then defined in a multimodal logic, with trust defined as a modality that is equivalent to an agent both desiring φ (in their logic this is an achievement goal) and believing that the target has the power, the capability and intention to achieve φ through action α. Their formalisation of trust thus incorporates those aspects of trust that we have mentioned earlier: an agent trusts another agent with regards to a specific goal. The role an agent plays is directly dependent on the social action that it is required to perform. As such this formalisation fits exactly with our own one. Lorini and Demolombe [2008] also present a *graded* version of the multimodal logic, which allows trust to have a strength, rather than be binary as in classical logic. This graded version of ForTrust is implemented [Hübner et al., 2009] using Jason [Bordini et al., 2007]. This implementation is very specific for the ART Testbed [Fullam et al., 2006]. In it, it is clear that certain assumptions need to be made regarding *how* the formalisation is actually calculated. In Algorithm 6 we give a more general version of the implemented algorithm, which is not limited to application in only the ART Testbed. We assume the existence of the predicates $request(a, \alpha, contract)$ and $performed(contract, a, \psi, time)$ in $\mathcal{L}_{Bel}$ to model the requests to perform action α from an agent a and the result ψ of a's performance of α, respectively.

The first thing to notice is that the formalisation of Lorini and Demolombe [2008] is very abstract. Given a direct experience in which the target agent performed α and the result was a world in which φ' held, how likely is it that next time the agent performs α, the desire φ is fulfilled? This is just one of the issues that the formalisation stays away from, but any implementation must necessarily solve. We specify this with **eval**, which compares the outcome of a previous interaction with the requirement φ and returns a numerical value. It

Algorithm 6: ForTrust

Input: $BC : \mathcal{B}$, the set of the agent's beliefs
Input: $IC : [\alpha]\varphi$, the goal to be achieved
Input: $a \in Agents$, the agent to be evaluated
Input: ϵ, a parameter defining the cut-off rate for capability of performing α
Input: δ, a parameter defining the cut-off rate for the power of achieving φ by performing α
Input: γ, a parameter defining the decay factor for evidence over time
Input: c_0, a default value for the *capability*
Input: p_0, a default value for the *power*

```
performed := 0
contracts := 0
outcome := 0
denominator := 0
foreach request(a, α, contract) ∈ B do
    contracts := contracts + 1
    if performed(contract, a, ψ, t) ∈ B then
        performed := performed + 1
        modifier := γ^(now−t)
        outcome := outcome + modifier · eval(φ, ψ)
        denominator := denominator + modifier
    end
end
if contracts = 0 then
    capability := c_0
end
else
    capability := performed / contracts
end
if capability ≤ ε then
    capability := 0
end
if performed = 0 then
    power := p_0
end
else
    power := outcome / denominator
end
if power ≤ δ then
    power := 0
end
x := min(capability, power)
```

Output: $trust(a, `[\alpha]\varphi', x)$

is easy to fill in a formula here so the same result is obtained as in Hübner et al.'s implementation for the ART Testbed [2009]. Another implementation of ForTrust is similar [Krupa et al., 2009]; however, it also adds the possibility of evaluating the trust in an agent to not perform a malicious action. While this requires a different type of logical reasoning, from a trust perspective it is rather similar. An agent simply trusts another agent regarding the fictitious action α': the action of not performing α. By considering the absence of an action as performing a different action and reasoning about that instead, Algorithm 6 is also a generalisation of Krupa et al.'s implementation of ForTrust.

A second remark is that while the ART Testbed provides the possibility of asking for reputation information from other agents, the implementation does not use this. We have therefore not included it either, but, just as the integration of *reputation* is considered as future work by Hübner et al. [2009], this algorithm can be extended to include reputation as a source of information for estimating both the power and capability of other agents.

Parametrisation of ForTrust

Looking at Algorithm 6 we see there are three parameters explicitly specified. Of these, γ is the easiest to interpret: it has the exact same effect as λ in BRS (see Section 6.5.1). The parameters δ and ϵ can be considered independently or together. Independently they describe the cut-off rates for power and capability, respectively. As such, a high ϵ means we do not wish to consider agents who do not perform α when asked, independent of the degree of success at achieving φ. We see ϵ thus directly specifies the importance of being capable, as we would expect from the algorithm. Similarly, δ specifies the importance of actually achieving φ if the agent performs α. However, the interplay between the two variables is complicated because if one of these two cut-off rates is unrealistically high, then it does not make any difference what the other value is at, and the trust will be 0, based on this unrealistic expectation. In other words, it depends on the probability distributions of power and capability, what influence δ and ϵ have. When both are above the cut-off rate, whether capability or power is the deciding factor also depends on their probability distributions: because we take the minimum, the smallest of the two is the deciding factor, given they are both larger than the cut-off rate. We thus see that capability is the deciding factor in the following two situations:

- $\mathit{capability} \leq \epsilon$ and $\mathit{power} > \delta$
- $\mathit{capability} > \epsilon$, $\mathit{power} > \delta$ and $\mathit{capability} < \mathit{power}$

The situations that power is the deciding factor are analogous. Note that we disregard the situations where both power and capability are smaller than δ and ϵ, respectively, because in such situations both are 0. We need to choose the values for δ and ϵ before calculating the agent's capability and power, and thus the best we can do is use an estimation: we can influence the probability that capability (or power) is the deciding factor. The probability that capability is

the deciding factor is thus the sum of the probabilities for either case above. This works out to the following formula:

$$\begin{aligned} Pr(cap\_decides) &= Pr_{cap}(c \leq \epsilon) \cdot Pr_{power}(p > \delta) \\ &\quad + Pr_{cap}(c > \epsilon) \cdot Pr_{power}(p > \delta) \cdot Pr_{cap}(c < p \mid p > \delta) \\ &= F_{cap}(\epsilon) \cdot (1 - F_{power}(\delta)) \\ &\quad + (1 - F_{cap}(\epsilon)) \cdot (1 - F_{power}(\delta)) \cdot \int_{\delta}^{\infty} (F_{cap}(x) - F_{cap}(\epsilon)) \cdot Pr_{power}(p = x)dx \end{aligned}$$

Where $Pr_{cap}(c > \epsilon)$ calculates the probability that c is greater than ϵ given the probability distribution of the capability of the target. Pr_{power} does the same for power, F_{cap} is the cumulative probability function for capability, and F_{power} the cumulative probability for power.

A similar equation can be found for deciding when power defines the trust value, which, because we are not discounting those situations where both $p \leq \delta$ and $c \leq \epsilon$, will not simply be the inverse probability of the above formula. All of this is needed in order to say something about how the values for δ and ϵ influence the relation between $Pr(cap\_decides)$ and $Pr(power\_decides)$. Nevertheless, as we can see, the equations are rather complex, and additionally, the probability distributions for power and capability are generally unknown, and thus the influence cannot be calculated exactly. As a rule of thumb we can assume that power and capability are distributed equally. In this case the probability that the capability is the deciding factor is greater than the probability that power is the deciding factor when ϵ is greater than δ. For this, we need to show that, if $\epsilon > \delta$, $Pr(cap\_decides) - Pr(power\_decides) \geq 0$. The proof is straightforward and hinges on two realisations: the first is that the two probability distributions for power and capability are the same, so Pr_{cap} and Pr_{power} represent the same probabilities. The second is that if $\epsilon > \delta$ the same holds for cumulative probabilities: $Pr(x \leq \epsilon) \geq Pr(x \leq \delta)$. With these two realisations, we can simplify the algebraic expression as follows:

$$Pr(cap\_decides) - Pr(power\_decides) \geq Pr(x \leq \epsilon) + Pr(x > \epsilon) \cdot Pr(x > \delta) \geq 0$$

□

The reverse is also true and $Pr(power\_decides) \geq Pr(cap\_decides)$ if $\delta > \epsilon$. We simplify this further and say, under the assumption that power and capability are equally distributed: if $\delta > \epsilon$ then power is more important than capability and vice versa if $\epsilon > \delta$.

Even if the assumption of equal distributions does not hold, it is clear that the relations between power and capability are influenced by δ and ϵ, and thus we see that all three parameters γ, δ and ϵ can be instantiated with different values, depending on what importance the agent wants to give to the different factors. As in BRS, and as we shall see, also in ReGReT, the **eval** function may also be parametric, in which case there are even more options available for the agent to adapt its trust model to the situation.

6.5.3 ReGReT

ReGReT [Sabater, 2003] attempts to define a comprehensive trust and reputation model, which takes many different types of information into account. The first two are *direct trust* and *information from other agents*, or reputation. However, it is the first model to consider the multifaceted aspect of trust by linking a trust evaluation to a behaviour (or role). An agent is not simply evaluated, but rather a trust evaluation is an evaluation of an agent performing a specific behaviour. To achieve this, ReGReT adds *ontological information* to the calculation. Finally, ReGReT considers the structure of the social network as a source of information about the relations between agents. While other models might use these different types of information, at the time ReGReT was presented it was the first system to incorporate all these different aspects into a single comprehensive trust model. As such it is one of the most influential trust and reputation models in existence.

Roles and the ontology

The ontological dimension is considered for both the calculation of direct trust and reputation. These values are calculated specifically for a single role. These roles are related through a role taxonomy. The calculation of direct trust and reputation is only done for so-called "atomic roles", which take a single aspect of an interaction into account, such as, for instance, the price of an item in an auction. These coincide with the leaves of the role taxonomy. For any interior role, the trust in an agent fulfilling that role is the weighted mean of the trust in that agent for each of the child nodes. For instance, in an electronic auction, a seller is evaluated based on the cost of an item and the delivery time. Thus, the trust in an agent based only on cost is calculated and similarly for delivery time. These are then aggregated using a weighted average to obtain the trust in that agent as a seller. The direct trust and reputation calculations below are thus the calculations for leaves of the role taxonomy.

Direct trust

ReGReT gives a clear description on how an agent can evaluate its own direct experiences with the target. It does this in terms of an *outcome*, which consists of two things: a contract between two agents and the resulting actions. It is represented as a tuple $o = \langle i, j, I, X^c, X^f, t\rangle$, where i is the evaluating agent, j is the target, t is the time when the contract was formed and I is a set of terms in an ontology that the contract is about. X^c is a vector with the agreed values of the contract for each issue in I. X^f is a similar vector, but with the actual values, after the contract is deemed "fulfilled". An agent's outcomes are stored in a part of the belief base, called the outcome database (ODB). The direct trust an agent has in the target is calculated directly from this ODB.

To perform this calculation, the outcomes need to be evaluated. For each atomic role there is a function $g_r : \mathbb{R}^n \times \mathbb{R}^n \to [-1, 1]$, where n is the length of the vectors X^c and X^f. This function is used to evaluate an outcome and

returns an impression in the range $[-1, 1]$. The impressions from all outcomes are aggregated and this is the direct trust of agent i in j concerning role r:

$$DT_{i\to j}(r) = \sum_{o \in ODB} f(now, time(o)) \cdot g_r(contract(o), fulfilment(o))$$

with $time(o) = t$, $contract(o) = X^c$ and $fulfillment(o) = X^f$ if we let $o = \langle i, j, I, X^c, X^f, t\rangle$. Moreover, $f(now, t) = \frac{f'(now,t)}{\sum_{j \in ODB} f'(now,time(j))}$, where $f' : \mathbb{R}^2 \to [0, 1]$ is a function to calculate the decay factor for outcomes over time. Examples are $f'(x, y) = 0.5^{x-y}$ or $f'(x, y) = y/x$. We see the aggregation method is a weighted average, with the weight dependent on the time an interaction took place.

This gives the *value* of the direct trust, but ReGReT also uses the *reliability* of the calculation. For direct trust this is defined by two factors: the *number of outcomes factor* and the *outcome reputation deviation*. These encode the uncertainty from possibly having too few interactions to reliably predict the other's behaviour, and the uncertainty from the variability in the outcomes, respectively. The reliability of direct trust, $DTRL_{i\to j}(r)$, is simply the multiplication of the reliability calculated from either factor.

Reputation

In addition to direct trust, information from other agents is taken into account. This is calculated in a number of manners, resulting in three different types of reputation: the *witness reputation*, a value giving the reputation according to information received from other agents, the *neighbourhood reputation*, calculated by considering the neighbours of the target in a social network and the *system reputation*: a default reputation based on the role played by the target.

Witness reputation is calculated in two steps: the first of these is to decide which witnesses' opinions to consider. ReGReT uses the topology of the social network to find the witnesses. The details of this analysis are not important to the further explanation of the model and we refer the reader to Sabater's work for these [2003]. The output of the social network analysis is a set of witnesses who are asked for their opinion. To decide how reliable a witness is, ReGReT uses two systems. The first of these is simply the individual reputation of the witness. If the reliability of this reputation is too low (according to a threshold), then another metric is used, once again based on the structure of the social network, which they call socialTrust. ReGReT has a set of conditional rules. The antecedent specifies properties that can hold in the social network and the conclusion is a statement about the reliability of an agent's opinion. If the witness' position in the social network coincides with the condition in the antecedent, the conclusion prescribes the reliability of the witness' opinion.

Neighbourhood reputation is calculated in a similar way to the socialTrust metric. System reputation is used in case neither witness reputation nor neighbourhood reputation can be used and is a type of default reputation, which is specified for any agent with certain properties, such as the role it plays in a

system or other information generally available. If no information is available at all, a default value is used.

As such, ReGReT has a list of reputation metrics, each considered less reliable than the next. Individual reputation is considered before witness reputation, which in its turn is considered before neighbourhood reputation or system reputation. This is achieved by considering the final metric reputation $R_{i\to j}(r) = \sum_{x\in W,N,S,D} \xi_x \cdot R_{i\overset{x}{\to} j}(r)$, where $R_{i\overset{x}{\to} j}(r)$ is the reputation type x, with W, N, S and D being shorthand for Witness, Neighbourhood, System and Default. The ξ_x are weights for this system which depend on the reliability of each metric, as defined by the functions $RL_{i\overset{W}{\to} j}, RL_{i\overset{N}{\to} j}$ and $RL_{i\overset{S}{\to} j}$. For the definition of these reliability functions we refer to Sabater [2003]. The weights for the types of reputation are then defined as follows:

- $\xi_W = RL_{i\overset{W}{\to} j}(r)$
- $\xi_N = RL_{i\overset{N}{\to} j}(r) \cdot (1 - \xi_W)$
- $\xi_S = RL_{i\overset{S}{\to} j}(r) \cdot (1 - \xi_W - \xi_N)$
- $\xi_D = 1 - \xi_W - \xi_N - \xi_S$

It is easy to see that if the reliability of the witness reputation is high (near 1), then the weight for the aggregation of the other reputation types is low.

Combining direct trust and reputation

The final calculation step is to combine direct trust with reputation. This is done using the following formula:

$$Trust_{i\to j}(r) = DTRL_{i\to j}(r) \cdot DT_{i\to j}(r) + (1 - DTRL_{i\to j}(r)) \cdot R_{i\to j}(r)$$

Trust is a weighted sum of direct trust and reputation, with the reliability of the direct trust defining the weights. This calculation can be performed for any of the atomic roles defined in the role taxonomy. For an internal node, representing a non-atomic role, the trust in its children must be calculated first. The trust in an agent performing a non-atomic role is then the weighted mean as described in Section 6.5.3.

We refer to Sabater's work [2003] for a full description of the algorithm.

Parametrisation of ReGReT

ReGReT is the most comprehensive trust model we consider and it has many different parts about which an agent could reason. The first and most obvious of these, are the weights used to calculate trust for non-atomic roles. ReGReT considers roles in a similar manner to the way we have incorporated them into our system. The roles thus have a double function. Firstly the role taxonomy defines the structure for aggregating atomic roles into non-atomic roles. Our reasoning

system, however, allows for rules to be set up defining the importance of these child roles in the aggregation and thus influence the weight of the aggregation, which in ReGReT is defined statically. An additional improvement is that for a specific goal, or set of beliefs, the importance of the child roles might be changed and thus the weights can be defined dynamically, dependent on the situation of the agent.

The second place the agent may incorporate reasoning is in the weight functions of direct trust. Both the time-dependent weight and the role-dependent evaluation functions are undefined in the model and left for the implementation. The time-dependent weight may be parametrised similarly to the decay factor in either BRS or ForTrust. The role-dependent evaluation functions are more interesting, though: the definition in ReGReT is in terms of a single issue of an interaction and thus a one-dimensional comparison. We have taken the liberty of extending this to an arbitrary function g_r for any atomic role r that calculates the evaluation of an outcome. If, as in the original description, g_r is a one-dimensional comparison it is obvious that this single issue is the only important factor in the calculation of trust for role r. Nevertheless, by converting this into a multi-dimensional comparison it should be possible, if the function g_r is parametric, to specify dynamically which issues of an outcome are important for role r dynamically.

Finally, in the social network analysis used for witness reputation and neighbourhood reputation, ReGReT defines a set of fuzzy conditional rules. Sabater explicitly states that these rules are hand-coded, but a better approach would be to automate the process. While our reasoning system does not allow for full automation, it does allow for a mechanism to adapt these rules.

6.6 Summary

In this chapter we have proposed AdapTrust, a method for integrating trust models into a cognitive agent. By making the parameters of a trust model explicit, an agent can proactively adapt these parameters' values and thus the trust model itself. The values of these parameters are expressions of the relative importance of different factors on the trust calculation. We introduce an explicit representation of these factors and a priority logic for representing their relative importance to each other. Priority rules link the agent's cognitive aspect (its goals and beliefs) and the social aspect (roles) with particular orderings of these factors. These, in turn, determine the value of the parameters. In this manner the trust calculation can be adapted to the cognitive and social dimensions of the agent system.

The formalisation of AdapTrust is presented, using a multi-context system representation of the BDI framework to incorporate the trust model. In addition to the formal definition of AdapTrust, we illustrate how it can be applied to particular trust models (BRS, ForTrust and ReGReT). This illustration serves two purposes: the first is to demonstrate our method and the formalisation we provide. The second is to provide a guide to perform this incorporation for other

trust models.

We intentionally left out the details of the implementation of the *resolve* and *instantiate* functions, and left the design of the trust priority rules deliberately vague. The details of their implementation depends on the specific agent architecture and trust model the agent designer uses, as well as the domain in which the agent should function. Filling in such details is an important step, but in this chapter we describe the *first* step: an abstract, declarative framework, describing a new way to integrate an agent's trust model into its reasoning system.

In the next chapter we show how the specification of the priorities can be used, not just for reasoning about trust, but rather to allow agents to argue about trust. The idea of arguing about the validity of trust evaluations was presented by Pinyol and Sabater-Mir [2010], who use BDI+Repage to generate the arguments, which are used to achieve more reliable communication about trust evaluations. The argument serves to make the trust evaluation more convincing, by linking it to an agent's knowledge about the environment. While this is an exemplary tool for the communication of trust, a similar argument needs to be constructed for every communicated trust evaluation. Furthermore, as we mentioned in Chapter 5, Pinyol's method only allows the agent to filter out unreliable witness information. We propose an alternative. By allowing agents to *adapt* their trust model, they could use argumentation to reach an agreement of what should form the support for a trust evaluation and change their trust models to coincide with this. In this way, future communicated evaluations could be accepted simply by virtue of the agents having agreed on what a trust evaluation means. Furthermore, the BDI+Repage model does not provide a method for such adaptation. The AdapTrust model we present in this chapter does, and we focus on the argumentation framework in the next chapter.

Chapter 7

Arguing about Trust

You remind me of a man!
What man?
The man with the power!
What power?
The power of hoodoo.
Who do?
You do!
What do I do?
You remind me of a man!

–Rudi Koster, quoting The Bachelor and the Bobby-Soxer (1947)

7.1 Introduction

As we have argued throughout this thesis, trust is a personal and subjective evaluation of a target for the fulfilment of a specific goal. Moreover, to compute a trust evaluation of a target, an agent needs information about that target. When no direct experiences are available, most models turn to witness information. Such information is subjective and requires some form of processing, before an agent can use it. In Part II we presented one such form of processing, Trust Alignment, but that requires the agent to have had a large number of interactions with other agents in the system. This is problematic in the case that agents have not (yet) established a large network of interactions. We are now ready to present Trust Adaptation, the second approach for processing witness' communications, which does not rely on a large number of interactions. Instead, agents need to share more information about their beliefs and goals, and how these influence their trust calculations. In this way, the agents can adapt their trust models to provide each other with *personalised trust recommendations*.

In Chapter 6 we presented AdapTrust, a BDI-framework for adapting the trust model to the agent's changing beliefs and goals. We now use AdapTrust

for two purposes: firstly, it allows an agent some introspection into the working of its trust model, and allows it to communicate its reasons for a particular trust evaluation. Secondly, it allows for adaptation: a witness can incorporate the goal of the requesting agent, in order to provide a personalised trust evaluation for it. Additionally, the agents can argue about their beliefs and reach an agreement, thereby also adapting their trust models to be more similar to each other's. These uses of AdapTrust in an argumentation dialogue allow agents to personalise trust evaluations to the requesting agent's requirements.

We build our communication model as an extension of Pinyol's framework for arguing about trust evaluations. This model already assumes a certain amount of introspection, but as discussed in Section 5.3, uses the argumentation to filter out witness' recommendations that are too dissimilar from the own point of view. We extend this to not simply make a binary decision to accept or reject, but accept more recommendations by allowing adaptation and persuasion. We start this chapter with an explanation of Pinyol's framework in Section 7.2, before describing our own extension in Section 7.3. In Section 7.4 we present a dialogue protocol for personalising trust evaluations, and in Section 7.5 we describe the empirical evaluation of our model. Section 7.6 discusses the results and presents possible scenarios for the model's application.

7.2 Pinyol's Argumentation Method

Our method for enabling personalised communication about trust is based on three capabilities an agent must have:

1. An agent must be able to adapt its trust model in order to personalise its evaluations to the other agent's needs.

2. An agent must be capable of communicating its criteria for evaluating trust, as well as the underlying beliefs and goals leading to these criteria.

3. An agent must be willing and able to change its trust model, if it is persuaded that its beliefs about the environment and thus its criteria for calculating trust are wrong.

We assume that agents are willing to adapt their model if they are convinced that it is inaccurate, thereby addressing point (3). For the ability to do this, as well as the possibility of adapting the model to another agent's needs, we use AdapTrust, as described in Chapter 6. This addresses point (1), leaving the problem that the agent must be capable of communicating its criteria for evaluating trust. The criteria for evaluating trust are given by an agent's beliefs and goals. What we need is thus a communication language that allows agents to talk about trust evaluations, the beliefs and goals these depend on, and the causal relationship between the two. We propose to extend Pinyol's framework for arguing about trust [2011], in order to allow for adaptation. Pinyol proposes an information-seeking dialogue for communicating about trust as a way for the receiver of a trust recommendation to decide whether or not to accept

the recommendation. The argumentation framework creates an argument that abstracts away from the computational process of the trust model, thereby allowing agents to discover what the original sources for evaluating a trust evaluation are. Nevertheless, it cannot answer *why* an aggregation of sources resulted in a specific evaluation. Our proposal extends Pinyol's framework and allows agents to answer such questions and we present it in Section 7.3, but first we summarise Pinyol's argumentation framework.

7.2.1 An ontology of reputation

Pinyol uses the $\mathcal{L}_{Rep}$ language to build the arguments, first introduced by Pinyol et al. [2007]. This language is based on a comprehensive ontology for discussing concepts of trust and reputation. This ontology defines a *social evaluation* with three compulsory elements: a target, a context and a value. The context is specified using a second language $\mathcal{L}_{Context}$, which is a first-order dynamic language (FODL-language, as described in Section 6.2.2) for describing the domain. The target is the agent under evaluation and the value is a quantification of the social evaluation. We will not go into details of this quantification, but the original description of the $\mathcal{L}_{Rep}$ language gives different alternatives for the representation of this quantification, encompassing most, if not all, representations used in modern computational trust and reputation models [Pinyol et al., 2007]. We define functions *target*, *context* and *value* that, when given a ground atom in $\mathcal{L}_{Rep}$, return its target, context and value, respectively. These functions will be used later in this chapter.

The taxonomy of social evaluations is given in Figure 7.1. Here we see how social evaluations are split into the different types of evaluations related to trust and reputation. This taxonomy is based on a sociological model of trust and reputation [Conte and Paolucci, 2002], which splits trust into a direct component, *image*, and a generalised concept of what the society thinks, *reputation*. These, in turn, are aggregations of direct experiences, shared voices and shared evaluations. In this way the ontology allows for the discussion of not just the final trust evaluation, but also the intermediate evaluations that are used in its calculation. The $\mathcal{L}_{Rep}$ language is a first-order language with the vocabulary from the ontology described above and operators $\wedge, \neg$ and $\rightarrow$. A special subset of sentences in $\mathcal{L}_{Rep}$ are ground atoms with either the predicate symbol $DExperience$ or $Comm$. These are the basic elements in the ontology, which specify the evaluations of either direct experiences the agent has observed or communications it has received.

7.2.2 Trust as an inferential process

Pinyol [2011] starts by modelling the trust model as an inference relation between sentences in $\mathcal{L}_{Rep}$. A trust model is considered as a computational process: given a finite set of inputs, such as beliefs about direct experiences or reputation, it calculates a trust evaluation for a target. The semantics of a computational process can be given by the application of a set of inference rules [Jones, 1997].

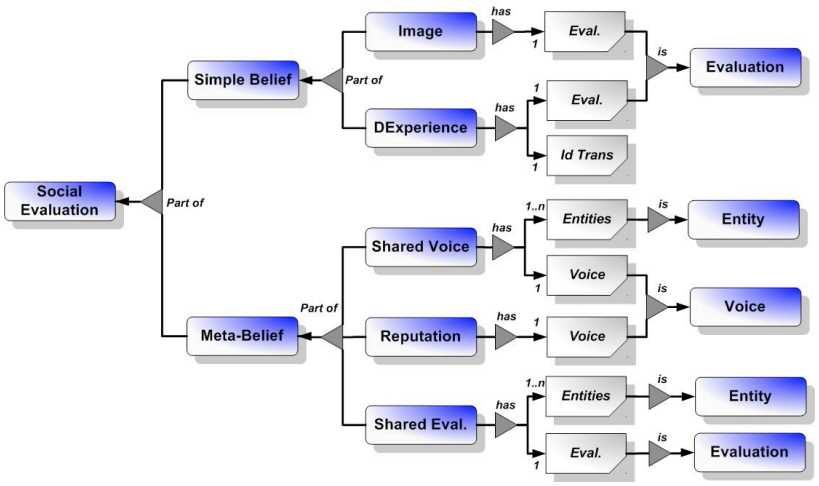


Figure 7.1: Taxonomy of social evaluations in the $\mathcal{L}_{Rep}$ ontology for talking about trust. Copied from Pinyol [2011].

In the case of trust models, we use this to specify the operational process of a trust model:

Definition 7.1 (Rule-based specification of a trust model)**.** Let $\mathcal{I}$ be a set of *inference rules*, and $\Delta \subseteq \mathcal{L}_{Rep}$ and $\delta \in \mathcal{L}_{Rep}$ sentences in $\mathcal{L}_{Rep}$. We say $\mathcal{I}$ specifies a trust model, if Δ supports a trust evaluation δ and $\Delta \vdash_{\mathcal{I}} \delta$. I.e., there exists a finite number of applications of inference rules $\iota \in \mathcal{I}$ by which we may infer δ from Δ.

Example 7.1. The inference rules themselves depend on the specifics of the computational process and thus the actual trust model being used, but for any computational trust model, such an inference relation exists. For instance, a trust model might have a rule:

$$\frac{img(T, X), rep(T, Y)}{trust(T, \frac{X+Y}{2})}$$

With img, rep and $trust$ predicate symbols in $\mathcal{L}_{Rep}$ and T, X and Y variables. For a specific target Jim, an agent knows $\{img(Jim, 3),\ rep(Jim, 5)\}$. It can thus infer $trust(Jim, 4)$ using the rule above. For a full example of representing a trust model as a set of inference rules, we refer to Pinyol and Sabater-Mir [2009b].

7.2.3 Arguing about trust

Arguments are sentences in the $\mathcal{L}_{Arg}$ language. This language is defined over another language $\mathcal{L}_{KR}$, that represents object-level knowledge. In Pinyol's framework $\mathcal{L}_{KR} = \mathcal{L}_{Rep}$, but in Section 7.3 we will supplement this language in order to extend the argumentation. A sentence in $\mathcal{L}_{Arg}$ is a formula $(\Phi : \varphi)$ with $\Phi \subseteq \mathcal{L}_{KR}$ and $\varphi \in \mathcal{L}_{KR}$. This definition is based on the framework for defeasible reasoning through argumentation, given by Chesñevar and Simari [2007]. This framework of argumentation provides a clear manner for constructing arguments from an underlying language, rather than just providing a way for resolving what set of arguments fulfil certain criteria, which is the usual role of an argumentation framework [Dung, 1995; Bench-Capon, 2003]. An alternative could be to model the trust model using a bipolar argumentation framework [Amgoud and Prade, 2009]. In most argumentation frameworks only one relation between different arguments is specified, the "attacks" relation. Bipolar arguments, however, also allow "support" relationships. Rather than using the deduction rules we describe below, we could model the trust inference using such support relationships. The advantage of this is that this allows the dialogue to be more straightforward: agents can ask each other what arguments "support" a specific argument, rather than trying to unravel the other's inference process. Nevertheless, representing the trust inference in a proof-theoretic manner in Chesñevar and Simari's framework makes more intuitive sense from the point of view of modelling the inferential process of a computational trust model.

Pinyol chooses to use Chesñevar and Simari's model, laying much of the groundwork for arguing about trust, so we choose to follow his approach. For a sentence $(\Phi : \varphi)$ in $\mathcal{L}_{Arg}$, intuitively Φ is the defeasible information required to deduce φ. Defeasible information is information that is rationally compelling, but not deductively valid. For instance, the propositional sentence "it is raining" $\rightarrow$ "it is cloudy" is an example of commonsense reasoning encoded as defeasible information: it is rationally compelling, because most of the time it is, in fact, cloudy if it is raining; however, there are rare instances when we would not call the weather cloudy, despite it raining. In $\mathcal{L}_{Arg}$ we write this as follows:

$$(\{\text{"it is raining"} \rightarrow \text{"it is cloudy"}\} : \text{ "it is raining"} \rightarrow \text{"it is cloudy"})$$

The structure of a sentence $(\Phi : \varphi)$ in $\mathcal{L}_{Arg}$ explicitly distinguishes between the defeasible information Φ and the conclusion φ that can be deduced from it, using a number of deduction rules, which we define below. If Φ is the empty set, then we say φ is non-defeasible, or strict, information.

Continuing the example above, if we know for a fact that it is raining, this could be modelled as the sentence $(\emptyset :$ "it is raining") in $\mathcal{L}_{Arg}$ and by using the defeasible information above, we could deduce $(\{$"it is raining" $\rightarrow$ "it is cloudy"$\}$: "it is cloudy") using the *Elim-IMP* rule below. We can conclude that it is cloudy, if we know it is raining and we accept the information that if it is raining then it is cloudy. We will use defeasible information to represent the trust model in $\mathcal{L}_{Arg}$, as we show below.

The argumentation language

Information is introduced into $\mathcal{L}_{Arg}$ using a set of elementary argumentative formulas. These are called *basic declarative units*.

Definition 7.2 (Basic Declarative Units)**.** A *basic declarative unit (bdu)* is a formula $(\{\varphi\} : \varphi) \in \mathcal{L}_{Arg}$. A finite set of bdus is an argumentative theory.

In the full framework of Chesñevar and Simari [2007], there are two types of bdus. The second is of the form $(\emptyset : \varphi)$, but, in our argumentation about trust, we do not use strict information, so omit it from the definition of bdus. Arguments are constructed using an argumentative theory $\Gamma \subseteq \mathcal{L}_{Arg}$ and the inference relation $\vdash_{Arg}$, characterised by the deduction rules *Intro-BDU, Intro-AND* and *Elim-IMP*.

Definition 7.3 (Deduction rules of $\mathcal{L}_{Arg}$ [Figure 1 of Chesñevar and Simari, 2007])**.**

$$\text{Intro-BDU:}\frac{}{(\{\varphi\} : \varphi)}$$

$$\text{Intro-AND:}\frac{(\Phi_1 : \varphi_1), \ldots, (\Phi_n : \varphi_n)}{(\bigcup_{i=1}^{n} \Phi_i : \varphi_1 \wedge \cdots \wedge \varphi_n)}$$

$$\text{Elim-IMP:}\frac{(\Phi_1 : \varphi_1 \wedge \cdots \wedge \varphi_n \rightarrow \beta), (\Phi_2 : \varphi_1 \wedge \cdots \wedge \varphi_n)}{(\Phi_1 \cup \Phi_2 : \beta)}$$

An argument $(\Phi : \varphi)$ is valid on the basis of argumentative theory Γ if and only if $\Gamma \vdash_{Arg} (\Phi : \varphi)$. Because the deduction rules, and thus $\vdash_{Arg}$, are the same for all agents, they can all agree on the validity of such a deduction; however, each agent builds its own argumentative theory, using its own trust model. Let $\mathcal{I}$ be the set of inference rules that specify an agent's trust model. Its bdus are generated from a set of $\mathcal{L}_{Rep}$ sentences Δ as follows:

- For any element δ in Δ, there is a corresponding bdu $(\{\delta\} : \delta)$ in $\mathcal{L}_{Arg}$.
- For all $\delta_1, \ldots, \delta_n$ such that $\Delta \vdash \delta_k$ for all $k \in [1, n]$, if there exists an application of an inference rule $\iota \in \mathcal{I}$, such that $\frac{\delta_1, \ldots, \delta_n}{\beta}$, then there is a bdu $(\{\delta_1 \wedge \cdots \wedge \delta_n \rightarrow \beta\} : \delta_1 \wedge \cdots \wedge \delta_n \rightarrow \beta)$, i.e., there is a bdu for every instantiated inference rule for the trust model specified by $\mathcal{I}$.

$\mathcal{L}_{Arg}$ is a non-monotonic logic and the consequence relation is, in the words of Chesñevar and Simari, "oriented towards a logic programming setting, where typically SLD resolution is used to model which literals follow from a given logic program" [2007]. For more details on the semantics of $\mathcal{L}_{Arg}$, we refer the interested reader to Chesñevar and Simari's work.

Example 7.2. Continuing Example 7.1, our agent might have the following argumentative theory:

$$\begin{aligned}\Gamma = \{&(\{img(Jim,3)\} :\ img(Jim,3))\\ &(\{rep(Jim,5)\} :\ rep(Jim,5)\\ &(\{img(Jim,3)\ \wedge\ rep(Jim,5) \rightarrow trust(Jim,4)\} :\\ &\quad img(Jim,3)\ \wedge\ rep(Jim,5) \rightarrow trust(Jim,4))\}\end{aligned}$$

From these bdus $(\Phi : trust(Jim,4))$ can be inferred, with Φ the union of all the defeasible information in the sentences of argumentative theory Γ. This argument contains various subarguments. An argument $(\Psi_1 : \psi_1)$ is a subargument of a second argument $(\Psi_2 : \psi_2)$ if $\Psi_1 \subseteq \Psi_2$. In our example, it is easy to see that, for instance, $(\{img(Jim,3)\} :\ img(Jim,3))$ is a subargument of $(\Phi : trust(Jim,4))$.

So far, we have discussed the bottom-up construction of arguments, but in practice it will be used in the other direction. An agent will have a trust evaluation, for instance $trust(Jim,4)$, and will work backwards to build a valid argument supporting that trust evaluation. Moreover, it can communicate this argument to another agent. This is where argumentation really comes into its own, because the other agent may have a different trust evaluation of target Jim, supported by a different argument. Because there is no consistency requirement on the argumentative theory Γ, such different arguments can be supported by the same argumentative theory, despite being inconsistent. Pinyol allows agents to communicate back and forth with arguments attacking each other, before an agent decides whether it can accept, or should reject the other agent's recommendation.

Attacks between arguments

In Chesñevar and Simari's work, attacks between arguments follow in a straightforward manner from the logic. They use the inconsistency between two sentences. Pinyol modifies this in order to deal with attacks between different trust evaluations. By adding a number of axioms to the $\mathcal{L}_{Rep}$ language, we can obtain the same result, but maintain the elegance of attacks following from logical inconsistency.

Definition 7.4 (Axioms for $\mathcal{L}_{Rep}$)**.** In order to have attacks between trust evaluations follow from the logic, we need to define when two trust evaluations are inconsistent. We define the following additional axioms for $\mathcal{L}_{Rep}$. For each predicate symbol p in the ontology of social evaluations of $\mathcal{L}_{Rep}$, we add the axiom:

$$\forall t \in Agents, v, w \in Values, c \in \mathcal{L}_{Context} : p(t, \text{`}c\text{'}, v) \wedge p(t, \text{`}c\text{'}, w) \rightarrow (v \ominus w < \tau)$$

Where $Agents$ is the set of agents in the system, $Values$ the range of values for evaluations in $\mathcal{L}_{Rep}$ and $\mathcal{L}_{Context}$ the language for specifying contexts, as

discussed in Section 7.2.1. Additionally, $\ominus$ is a metric on $Values$, and $\tau \in \mathbb{R}$ is a threshold, under which we consider trust evaluations to be similar.

With the addition of these axioms to $\mathcal{L}_{Rep}$ we include Pinyol's definition of attack [2011, page 123] as a special case of the attack relation that is defined by Chesñevar and Simari (see Definition 7.5). Specifically we obtain Pinyol's attack relationship for the threshold $\tau = 0$.

Example 7.3. In Example 7.2, the values of the trust predicates, such as *rep*, *img* and *trust*, are integers, so an example of $\ominus$ in this situation could be $x \ominus y \stackrel{def}{=} |x - y|$, the Euclidean distance between real values. With threshold $\tau = 1$, for instance, we see that $trust(Jim, 4)$ and $trust(Jim, 3)$ are consistent. However, $trust(Jim, 4)$ and $trust(Jim, 2)$ are inconsistent. Similarly, we see that $img(Jim, 1)$ and $rep(Jim, 5)$ are consistent, as are $trust(Jim, 4)$ and $trust(Dave, 0)$: the axioms in Definition 7.4 only limit those trust evaluations where the predicate symbol and target are the same. We have omitted the context of the trust evaluations in this example, and simply assume that we are talking about one single context.

In contrast to most argumentation frameworks, Chesñevar and Simari's framework does not require the explicit definition of an attack relation. This relation follows directly from the logic by defining attacks in terms of logical inconsistencies.

Definition 7.5 (Attacks between arguments [adapted from Definition 3.4 of Chesñevar and Simari, 2007])**.** Let Γ be an argumentative theory and $(\Phi_1 : \varphi_1), (\Phi_2 : \varphi_2) \in \mathcal{L}_{Arg}$ be arguments based on Γ. Then $(\Phi_1 : \varphi_1)$ *attacks* $(\Phi_2 : \varphi_2)$ if there exists a subargument $(\Phi_2' : \varphi_2')$ of $(\Phi_2 : \varphi_2)$ such that $\{\varphi_1, \varphi_2'\}$ is inconsistent in $\mathcal{L}_{KR}$.

Given an attack relationship as above, there are still many choices regarding what set of arguments is *accepted*, thus giving different semantics to the reasoning system defined by an argumentation framework. The most common semantics are "credulous" and "skeptical" acceptance of arguments, as defined by Dung [1995]. Pinyol adopts a credulous acceptability semantics for his argumentation. After all the communication has completed and all the different arguments (for and against the trust recommendation) have been presented, the agent uses the acceptability semantics to decide whether or not to accept that recommendation. We will return to the acceptability of a trust recommendation in Section 7.4, but it will not be in terms of the arguments themselves, so we omit a discussion on the acceptability of arguments.

Pinyol's framework, as described above, allows for agents to construct arguments about trust using predicates from $\mathcal{L}_{Rep}$; however, this is only part of the full language needed for being able to build arguments for personalised trust recommendations. The problem with the framework so far is that the trust model's functioning is introduced into the argumentation language in the form of bdus (see above). This means agents cannot explain why their trust model

performs a specific calculation, because this is treated as defeasible information. In the next section we present our extension to this framework, which allows agents to explain the reasons for their trust model's functioning.

7.3 Extending the Argumentation Language

In this section we present our extension of the argumentation language presented in Section 7.2. The extension allows agents to fully express the importance of criteria in their trust model. Because Pinyol's argumentation works with a trust model that does not allow for introspection into the cognitive underpinnings of its calculations, it cannot connect the trust evaluation to underlying beliefs and goals. In Chapter 6 we presented AdapTrust, an agent model that makes this connection between the trust model and the beliefs and goals explicit, but so far we have not provided a language in which agents can communicate such relations. We now extend the argumentation framework presented in Section 7.2 with concepts from AdapTrust. In AdapTrust the *reason* an agent performs this computation (and not some other one) is twofold: firstly the trust model follows an algorithmic method for aggregating the input. Secondly, the agent's beliefs and goals fix the parameters of this algorithm.

We do not propose to explain the algorithmic processes in the trust model, but the criteria, given by beliefs and goals, that define the trust model's parameters can be incorporated into the argumentative theory. For this, we need to represent the dependency of the trust model on the beliefs and goal of an agent in $\mathcal{L}_{Arg}$. In Definition 7.1 we used the observation that the semantics of a computer program can be given in terms of inference rules. The inference rules $\mathcal{I}$ specify how a trust evaluation can be deduced from a set of inputs Δ. However, in AdapTrust the algorithm has parameters that depend on the agent's beliefs and goal. The inference rules should reflect this.

Let ***trust_ calculation*** be a trust model that is compatible with AdapTrust (see Section 6.3), that is represented by the inference rules $\mathcal{I}$. Furthermore let $\Delta \subseteq \mathcal{L}_{Rep}$ and $\delta \in \mathcal{L}_{Rep}$, such that $\Delta \vdash \delta$ for ***trust_ calculation***. From Definition 7.1 we know there is a proof applying a finite number of inference rules $\iota \in \mathcal{I}$ for deducing δ from Δ. However, this deduction in AdapTrust depends on a set of parameters, $Params_{\textbf{\textit{trust\_ calculation}}}$. Therefore, the inference rules must also depend on these parameters. For each $\iota \in \mathcal{I}$, we have $Params_\iota \subseteq Params_{\textbf{\textit{trust\_ calculation}}}$, the (possibly empty) subset of parameters corresponding to the inference rule. The set of parameters corresponding to a proof $\Delta \vdash \delta$ is simply the union of all parameters of the inference rules used in the deduction. Let the beliefs Ψ, goal γ and role r determine the values for all these parameters, as described in Section 6.4.2. We denote this as $\Delta \vdash^{\Psi,\gamma,r} \delta$, which states that the trust model infers δ from Δ, given beliefs Ψ, goal γ and role r. Similarly we have $\iota^{\Psi,\gamma,r} \in \mathcal{I}^{\Psi,\gamma,r}$ to denote an inference rule with the parameters $Params_\iota$ instantiated in AdapTrust using beliefs Ψ, goal γ and role r.

This allows us to redefine the set of bdus and thus the argumentative theory so that the argumentation supporting a trust evaluation can be followed all the

way down to the agent's beliefs, the goal it is attempting to achieve and the role the target must perform. We recall from Section 7.2.1 that the *context* in $\mathcal{L}_{Rep}$ is specified using a FODL. This is expressive enough to represent the socially-dependent goals of AdapTrust, that we described in Section 6.3.3 (page 132). We defined a socially-dependent goal as $[\alpha]\psi\langle R\rangle$, which states that the agent wants to fulfil ψ with program α, requiring agents performing roles R. This can be used in $\mathcal{L}_{Context}$ as a representation of the context in which the trust evaluation is made. Nevertheless, the representation in the context of $\mathcal{L}_{Rep}$ is insufficient for arguing about how the socially-dependent goal and beliefs of an agent influence the calculation of a trust evaluation and we must extend $\mathcal{L}_{KR}$ to encompass the various languages in AdapTrust. We define $\mathcal{L}_{KR} = \mathcal{L}_{Rep} \cup \mathcal{L}_{PL} \cup \mathcal{L}_{Rules} \cup \mathcal{L}_{Bel} \cup \mathcal{L}_{Goal}$, where $\mathcal{L}_{PL}$ is the language of priorities (see Section 6.3.2), $\mathcal{L}_{Rules}$ the language describing Priority Rules (see Section 6.4.1), $\mathcal{L}_{Bel}$ the language of the agent's beliefs (see Section 6.2.3) and $\mathcal{L}_{Goal}$ that of the agent's goals (also see Section 6.2.3). Using this $\mathcal{L}_{KR}$, the argumentation language can be extended to encompass more of the agent's reasoning process, with the bdus for $\mathcal{L}_{Arg}$ defined as follows:

Definition 7.6 (Basic Declarative Units for $\mathcal{L}_{Arg}$)**.** Let $\delta \in \mathcal{L}_{Rep}$ be an agent's trust evaluation based on inference rules $\mathcal{I}^{\Psi,\gamma,r}$, such that $\Delta \vdash^{\Psi,\gamma,r} \delta$ with $\Delta \subseteq \mathcal{L}_{Rep}$, $\Psi \subseteq \mathcal{L}_{Bel}$, $\gamma \in \mathcal{L}_{Goal}$ and $r \in \mathcal{L}_{Bel}$. For each $\iota \in \mathcal{I}^{\Psi,\gamma,r}$, let $Params_\iota$ be the corresponding set of parameters. Furthermore, we recall from Definition 6.2 (on page 128) that *labels* is a function that, given a set of parameters, returns a set of constants in $\mathcal{L}_{PL}$, the language of the priority system. Finally let $\Xi \subseteq \mathcal{L}_{Rules}$ be the agent's set of trust priority rules and $\Pi \subseteq \mathcal{L}_{PL}$ be its priority system based on Ψ, γ and r, then:

1. For any sentence $\psi \in \Psi$, there is a corresponding bdu $(\{\psi\} : \psi)$ in $\mathcal{L}_{Arg}$.
2. The goal γ has a corresponding bdu $(\{\gamma\} : \gamma)$ in $\mathcal{L}_{Arg}$.
3. The role r has a corresponding bdu $(\{r\} : r)$ in $\mathcal{L}_{Arg}$.
4. For all priorities $\pi \in \Pi$ and all the rules $\xi \in \Xi$ the following bdus are generated:
   - if ξ has the form $belief\_rule(\text{`}\Phi\text{'}, \text{`}\pi\text{'}, v)$ and $\Phi \subseteq \Psi$ then $\big(\{(\bigwedge_{\varphi\in\Phi}\varphi)\rightarrow\pi\} : (\bigwedge_{\varphi\in\Phi}\varphi)\rightarrow\pi\big)$ is a bdu in $\mathcal{L}_{Arg}$
   - if ξ has the form $goal\_rule(\text{`}\gamma\text{'}, \text{`}\pi\text{'}, v)$ then $(\{\gamma\rightarrow\pi\} : \gamma\rightarrow\pi)$ is a bdu in $\mathcal{L}_{Arg}$
   - if ξ has the form $role\_rule(\text{`}r\text{'}, \text{`}\pi\text{'}, v)$ then $(\{r\rightarrow\pi\} : r\rightarrow\pi)$ is a bdu in $\mathcal{L}_{Arg}$
5. For all $\alpha_1, \ldots, \alpha_n$ such that $\Delta\vdash^{\Phi,\gamma,r}\alpha_k$ for all $k \in [1,n]$, if there exists an application of an inference rule $\iota^{\Psi,\gamma,r} \in \mathcal{I}^{\Psi,\gamma,r}$, such that $\frac{\alpha_1,\ldots,\alpha_n}{\beta}$ and $labels(Params_{\iota^{\Psi,\gamma,r}}) = L$ then $\big(\{(\bigwedge_{\pi\in\Pi_L}\pi)\rightarrow(\alpha_1 \wedge\cdots\wedge \alpha_n \rightarrow \beta)\} : (\bigwedge_{\pi\in\Pi_L}\pi)\rightarrow(\alpha_1 \wedge\cdots\wedge \alpha_n\rightarrow\beta)\big)$ is a bdu of $\mathcal{L}_{Arg}$. With $\Pi_L \subseteq \Pi$ the set of priorities corresponding to labels L (as specified in Definition 6.3).

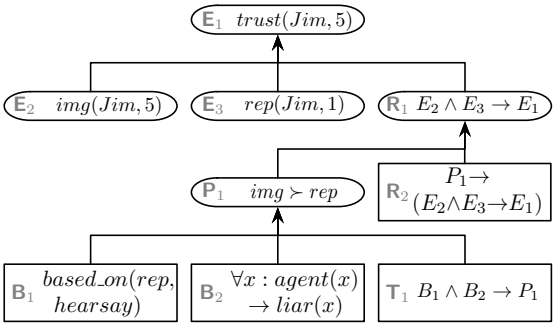


Figure 7.2: An example of an argument. The rectangular nodes are bdus.

In items 1, 2 and 3 the relevant elements of the agent's reasoning are added to the argumentation language. In items 4 and 5 the implements for reasoning about trust are added: in 4 the trust priority rules of AdapTrust, which link beliefs, goals and roles to priorities, and in 5 the rules of the trust model. The bdus added in 5 contain a double implication: they state that if an agent has the priorities in Π_L then a *trust rule* (which was a bdu in Pinyol's argumentative theory) holds. In practice what this accomplishes, is to allow the argumentation to go a level deeper: agents can now argue about *why* a trust rule, representing an application of an inference rule in the trust model, holds.

An argument for a trust evaluation can be represented in a tree. We call this an argumentation tree and give an example of one in Figure 7.2. The argumentation tree can be followed by applying the deduction rules of $\mathcal{L}_{Arg}$ at each level. In order to be succinct we have omitted the defeasible information part of the sentences, as well as the quotation marks in each node. Furthermore, we use shorthand in the tree by referring to nodes, rather than repeating the content of a node. For instance, in node R_1 we can expand $E_2 \wedge E_3 \rightarrow E_1$ to its meaning: $img(Jim, 5) \ \wedge \ rep(Jim, 1) \rightarrow trust(Jim, 5)$. An argumentation tree, such as this one, is used in a dialogue to communicate personalised trust evaluations.

We must also redefine attacks in this framework because $\mathcal{L}_{KR}$ is now a combination of many different underlying logics, and merely dealing with inconsistency in $\mathcal{L}_{Rep}$ is insufficient. Luckily, our dialogue does not allow for attacks between some of the basic arguments, and specifically conflicts of beliefs and priority rules are resolved in a different manner. Inconsistency in $\mathcal{L}_{KR}$ can thus be limited to inconsistency among sentences in one of either $\mathcal{L}_{Rep}$ or $\mathcal{L}_{PL}$. We say two sentences $\varphi, \psi \in \mathcal{L}_{KR}$ are inconsistent if and only if one of the following conditions holds:

- $\varphi, \psi \in \mathcal{L}_{PL}$ and $\{\varphi, \psi\} \vdash \bot$
- $\varphi, \psi \in \mathcal{L}_{Rep}$ and $\mathcal{A} \cup \{\varphi, \psi\} \vdash \bot$, with $\mathcal{A}$ the axioms as in Definition 7.4

This definition of inconsistency in $\mathcal{L}_{KR}$ allows us to use the original definition of attack from Definition 7.5.

7.4 Dialogue Protocol for Personalising Trust

The argumentation in the previous section can be used by an individual agent to justify its trust evaluation in a language that the other agents understand. We now specify a protocol that allows agents to argue back and forth in order for the requesting agent to receive a personalised trust recommendation from the witness. We start with the formal dialogue rules before showing informally how the protocol works and the choices an agent must make, using the example of Figure 7.2 to guide us in Section 7.4.2.

7.4.1 A formal dialogue protocol

We start our description of how the argument framework of the previous section is used by defining a formal dialogue system for communication about personalised trust recommendations in which all the agents' options mentioned above are available. The system we need is, for a large part an information-seeking dialogue system, according to the classification by Walton and Krabbe [1995], and it thus stands to reason that we use a protocol similar to the one presented by Parsons et al. [2003]. However, while our dialogue is for a large part information-seeking, it also incorporates some aspects of persuasion dialogues. We thus present the formal system in a similar structure to the dialogue system presented by Prakken [2005] for persuasion dialogues, in order to allow for some locutions in addition to the "question", "assert" and "challenge" locutions proposed by Parsons et al.

Definition 7.7 (Dialogue System for Personalised Trust [adapted from Definition 3 of Prakken, 2005])**.** A *dialogue system* for personalised trust is a tuple $\mathcal{D} = \langle \mathcal{L}_C, P, CR \rangle$ where $\mathcal{L}_C$ (the communication language) is a set of *locutions*, P is a *protocol* for $\mathcal{L}_C$, and CR is a set of *effect rules* of locutions in $\mathcal{L}_C$, specifying the effects of the locutions on the participants' commitments.

The three parts are described below, but first we must define some of the basic elements of a dialogue. The first of these is the set of participants themselves. The agent that wishes to obtain a trust evaluation of a target is seeking a recommendation from a witness, who supplies a recommendation. These agents are the participants of the dialogue and we denote them with Q for the recommendation-seeker and R for the recommendation-supplier. Both of these agents have a *commitment store*, a set of sentences in $\mathcal{L}_{Arg}$ that they have committed themselves to [Walton and Krabbe, 1995]. Commitment is a complicated concept, but we use it in a very specific way: an agent's commitment store contains beliefs it has voiced during the dialogue and is *committed* to justify and defend. Because the dialogue is essentially an information-seeking dialogue, the recommendation-supplying agent R will mainly be the one committing itself to sentences in the dialogue. As the dialogue progresses, the recommendation-supplier will justify, in increasing detail, why the initially communicated trust evaluation holds. Every justification of this kind adds to the recommendation-supplier's commitment store. The agents' commitment stores are denoted C_Q

and C_R for agents Q and R, respectively. Initially both agents' commitment stores are empty.

With these concepts in place we can move on to the definition of the locutions and protocol of a dialogue system. We start with the locutions.

Definition 7.8 (Locutions for Personalised trust)**.** The *locutions* allowed in the dialogue for personalised trust are specified by $\mathcal{L}_C$ and include the basic locutions for information-seeking, specified by Parsons et al. [2003]. The locutions are explained in Table 7.1 and the locutions **request_recommendation**, **assert** and **challenge** correspond directly to "question", "assert" and "challenge" in Parsons et al.'s system. Moreover, **justify** also corresponds to "assert" in Parsons et al.'s framework, but because they do not allow agents to backtrack, the sentence being justified is always immediately clear from the previous dialogue steps. The locutions **counter** and **argue** are not present in regular information-seeking dialogues. We add these so that agents can propose alternative priority systems for AdapTrust or attempt to persuade each other about their beliefs — thereby facilitating the adaptation of the agents' trust models.

Some of the locutions have an effect on an agent's commitment store. We usually denote the agent (either Q or R) that is sending a message, also called making a move, with s and the other agent with $\overline{s}$. We take C'_s to be the new commitment store of agent s after sending the locution, and C_s is the old commitment store prior to sending. The way the commitment store is updated for each locution is detailed in Table 7.2, which thus defines the rules CR of the dialogue.

Not all locutions can be uttered at any moment, there are rules to the dialogue. These are defined by the protocol P in terms of the moves allowed.

Definition 7.9 (Moves and dialogues [adapted from Definition 5 of Prakken, 2005])**.** The set M of *moves* in a dialogue is defined as $\mathbb{N} \times \{R, Q\} \times \mathcal{L}_C$, where the three elements of a move m are denoted by, respectively:

- $id(m)$, the numerical identifier of the move
- $player(m)$, the agent performing in the move
- $speech(m)$, the speech act performed in the move

The set of *dialogues*, denoted by $M^{\leq\infty}$, is the set of all sequences $m_1, \ldots$ from M, such that each i^{th} element in the sequence has identifier i and for any $i > 1$, $player(m_i) \neq player(m_{i-1})$[1]. The set of *finite* dialogues is denoted by $M^{<\infty}$. For any dialogue $d = m_1, \ldots, m_i, \ldots$, the sequence $m_1, \ldots, m_i$ is denoted by d_i, where d_0 denotes the empty dialogue. When d is a dialogue and m a move, then $d; m$ denotes the continuation of d with m.

[1] Note that this is a specific implementation of the turn-taking function in Prakken's dialogue system [2005].

Locution	*Use*
request_recommendation(t, γ, r)	The initial request for a recommendation, with $t \in Agents$, γ the goal and r the role that Q wants the recommendation for.
assert(p)	Assert that p is true, where $p \in \mathcal{L}_{Arg}$.
justify(p, S)	Assert that $S \subset \mathcal{L}_{Arg}$ is the (direct) support for p in $\mathcal{L}_{Arg}$.
challenge(p)	Challenge a sentence $p \in \mathcal{L}_{Arg}$ in the other agent's commitment store. An agent may challenge a sentence p if it wants the other agent to justify p.
counter(π_R, π_Q)	Propose an alternative priority π_Q to priority π_R with $\pi_Q, \pi_R \in \mathcal{L}_{PL}$. Note that this switches roles: **counter** is similar in use to **assert**, so the agent Q, that has thus far only been challenging assertions, now proposes its own priority, that R can now challenge.
argue(ψ)	Propose to enter into a separate persuasion dialogue about beliefs $\psi \subset \mathcal{L}_{Bel}$. The details of this dialogue are outside the scope of this thesis, but we propose to use the dialogue system proposed by Prakken [2009].
end	Indicate that the dialogue has concluded.

Table 7.1: Locutions in $\mathcal{L}_C$, the communication language for personalised trust recommendation dialogues

Locution	*Effect on commitment store*
request_recommendation(t, γ, r)	$C_Q = \emptyset$, $C_R = \emptyset$
assert(p)	$C'_s = C_s \cup \{p\}$
justify(p, S)	$C'_s = C_s \cup S$
challenge(p)	$C'_s = C_s$
counter(π_1, π_2)	$C'_s = C_s \cup \{\pi_2\}$
argue(ψ)	$C'_s = C_s$
end	$C'_s = C_s$

Table 7.2: The effect that the various locutions in $\mathcal{L}_C$ have on the sender's commitment store

A protocol P on a set of moves M is a set $P \subseteq M^{<\infty}$ satisfying the condition that whenever $d \in P$, so are all initial sequences of d. We define a partial function $Pr : M^{<\infty} \rightarrow \mathcal{P}(M)$ for personalised trust dialogues, that allows us to derive the protocol P. Prakken [2005] defines this in the opposite manner: with the protocol defining the function. In practice, however, it is easier to define the function than all possible sequences of legal moves.

Definition 7.10 (Protocol function for Dialogues for Recommending Trust)**.** $Pr : M^{<\infty} \rightarrow \mathcal{P}(M)$ defines the set of legal moves in a dialogue, and thus by induction defines the *protocol* P of a dialogue. We do this, by first defining the *preconditions* for each of the possible speech acts. These are listed in Table 7.3 and use the functions on $\mathcal{L}_{Rep}$ that we defined in Section 7.2.1. We define the function *pre* that, given a speech act, a player and a dialogue, returns whether the preconditions are true or false. This allows us to define a function that returns all legal moves, given the dialogue so far:

- $Pr(d_0) = \{(1, Q, \textbf{request\_recommendation}(t, \gamma, r))\}$
- $Pr(d; m_i) = \{(i{+}1, s, lm) |\ s = \overline{player(m_i)} \wedge lm \in \mathcal{L}_C \wedge pre(lm, s, d; m_i)\}$

If the persuasion dialogue about argumentation is guaranteed to terminate, then the dialogue for recommending trust is guaranteed to terminate. The proof of this is trivial, given that $\mathcal{L}_{Arg}$ contains a finite number of elements and the protocol guarantees no steps are repeated. It depends, however, on the agents' choices of the legal moves how fast it reaches a desirable outcome. A desirable outcome is furthermore dependent on the agents actually adapting their trust models when necessary. This is not treated in the actual dialogue: if either agent receives a trust priority rule as the justification for a priority, it may choose to *add* this to its own rule base. This is a choice made outside of the dialogue, and if this happens then the argumentative theories change. This means the logic for the current dialogue no longer represents the agents' stances, and therefore the agent should choose to end the current dialogue. The seeker should restart with a new request for recommendations. In the next section we discuss the choices an agent can make in more detail, whereas this section provided the formal model of the possible choices an agent may make.

7.4.2 A dialogue for recommending trust

The protocol of the previous section defines a dialogue for two agents: a recommendation-seeker and a recommendation-supplier. If we look at the rules for the **end** locution in Table 7.3, we see that either agent may, if it does not want to continue conversing, end the dialogue at any point. The ending of a dialogue gives no formal guarantees about whether the trust evaluations communicated are personalised to the seeker's criteria for calculating trust, but the more information exchanged, the higher the chance that the seeker can obtain useful information. In the rest of this section, we describe the options both participants have at each point in the dialogue and how they can decide on what to communicate. The decisions an agent can make are summarised in Figure 7.3.

Locution	*Precondition. d is the dialogue so far and s the player*
request_recommendation(t, γ, r)	A recommendation-seeker may only request a recommendation in the first move, t must be a target, γ a goal and r a role. Formally: $d = d_0$, $t \in Agents$, $\gamma \in \mathcal{L}_{Int}$ and $r \in Roles$
assert(p)	A recommendation-supplier may only assert a trust evaluation in the second move, and the context of the recommended trust evaluation must be equal to the goal for which it was requested. Formally: $d = m_1$, $player(m_1) = \overline{s}$, $p \in \mathcal{L}_{Rep}$ and $context(p) = \gamma$, with γ the goal in $speech(m_1)$.
justify(p, S)	A sentence p can be justified, if it is in the current player's commitment store and the other player challenged it in a previous move. Formally: let $d = d_{i-1}; m_i$ and $\overline{s} = player(m_i)$, then there is a move m in d, such that $player(m) = \overline{s}$ and $speech(m) =$ **challenge**(p). Furthermore $p \in C_s$, $S \vdash_{Arg} p$ and $S \not\subseteq C_s$.
challenge(p)	A sentence p can be challenged, if it is in the other player's commitment store and the current player has not previously challenged it. Formally: let $d = d_{i-1}; m_i$ and $\overline{s} = player(m_i)$, then there is no move m in d such that $player(m) = s$ and $speech(m) =$ **challenge**(p). Furthermore $p \in C_{\overline{s}}$.
counter(π_1, π_2)	A priority π_1 can be countered by priority π_2, if it is in the other player's commitment store and π_2 is not yet in the current player's commitment store. Formally: let $d = d_{i-1}; m_i$ and $\overline{s} = player(m_i)$, then $\pi_1 \in C_{\overline{s}}$, $\pi_2 \notin C_s$ and $\pi_1, \pi_2 \in \mathcal{L}_{PL}$.
argue(ψ)	The current player may propose to argue about belief ψ if ψ is in the other player's commitment store and the player has not previously proposed to argue about ψ. Formally: let $d = d_{i-1}; m_i$ and $\overline{s} = player(m_i)$, then there is no move m in d such that $speech(m) =$ **argue**(ψ). Furthermore $\psi \in C_{\overline{s}}$ and $\psi \in \mathcal{L}_{Bel}$.
end	A player may always choose to end the dialogue after the first move. Formally: $d \neq d_0$

Table 7.3: The preconditions, in terms of the dialogue, for the various locutions in $\mathcal{L}_C$

The dialogue starts with the seeker contacting the supplier to request its recommendation of a partner, performing a specific role, in order to achieve the seeker's goal. The supplier provides a recommendation, at which point the dialogue begins in earnest. The guiding principle in the dialogue is that the seeker agent is trying to decide whether the recommendation is acceptable or what further information and adaptation is required for this. Thus, in the diagram of Figure 7.3 the first decision is whether or not to accept the argument. If the argument is accepted, or rejected, then the seeker simply ends the dialogue. If the argument is not immediately accepted, or rejected, the next step is to decide which of the nodes of the argumentation tree is most likely to expedite this decision. This choice is made in the "Select node in argument" action of the diagram. In the description of the protocol below, we also describe this selection process. After selecting a node, the protocol determines what courses of action are available to the agent, based on the type of the node.

The example we use to describe the dialogue is the same as in Section 7.3, with the argumentation tree in Figure 7.2. As in the figure, we will use the identifier of the arguments in the tree as shorthand for the content of the node and write, for instance, E_1 for $trust(Jim, 5)$. The supplier does not reveal the entire argumentation tree at once. It only discloses information when the seeker asks for it.

After the seeker performs the first move, communicating the speech act **request_recommendation**(Jim, γ, r) for some goal γ and role r, the supplier provides its evaluation E_1. We thus see that the commitment store of agent R is $\{trust(Jim, 5)\}$ at this point, or to be precise $\{(\Phi : trust(Jim, 5)\}$ with Φ all the defeasible information it needs to deduce $trust(Jim, 5)$. This defeasible information, however, is not included in the communication: all the bdus can be found by exploring the argumentation tree.

The seeker can respond to the move (2, R, **assert**$(trust(Jim, 5))$ by either ending the dialogue, or challenging that single evaluation. In most cases the first step will be to challenge the initial recommendation; however, if the seeker Q can calculate its own trust evaluation of Jim it may build an argument that supports this evaluation using its own trust model. If this argument does not attack the argument E_1 then it might want to accept it outright, especially if it has previously entered trust recommendation dialogues with the same recommender; previous recommendation dialogues could already have resulted in one of the agents adapting its model.

Throughout most of the dialogue, challenging those arguments for which the agent can build an attacker is a way of quickly discovering where the trust model can be adapted; thus finding those nodes that can be attacked is an effective heuristic for the "Select node in argument" action in the decision process of Figure 7.3. Let us assume that agent Q challenges the evaluation and the move $(3, Q,$**challenge**$(E_1))$ is made in the dialogue. The supplier R then has the option again to end the dialogue, or perform the only other legal move, respond with a justification. It thus responds with the move $(4, R,$ **justify**$(E_1, \{E_2, E_3, R_1\}))$, and correspondingly its commitment store is

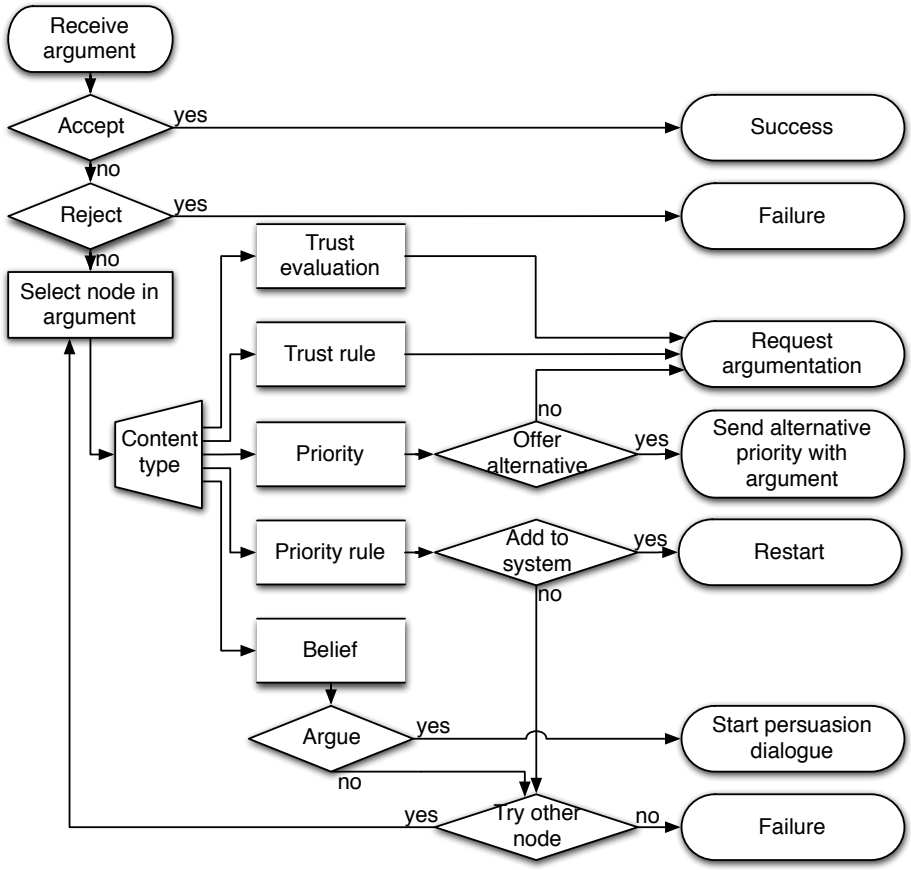


Figure 7.3: Diagram of the choices the seeker can make during a dialogue for trust recommendations

updated to contain the justification. After this dialogue move, we thus have $C_R = \{trust(Jim, 5), img(Jim, 5), rep(Jim, 1), img(Jim, 5) \wedge rep(Jim, 1) \rightarrow trust(Jim, 5)\}$ and C_Q is still the empty set. It is now the seeker's turn again and it must make a choice which of the three new sentences to challenge. It can, at a later stage, always return to challenge any of the others.

In $\mathcal{L}_{Arg}$, a trust evaluation is deduced from a trust rule and a number of inputs for the trust model using $Elim{-}IMP$ (see Definition 7.3). In the example these are trust rule R_1 and the trust evaluations E_2 and E_3. To decide whether or not to accept a trust rule, such as R_1 in our example, the seeker can compare it to the output of its own trust model, by using this with the inputs in the argument. We illustrate this first in our example before giving the general case. In our example, the seeker can find attacking arguments by assuming E_2 and E_3 hold and finding the evaluations $\mathcal{E}^* = \{E' \mid \{E_2, E_3\} \vdash E'\}$ in the seeker's trust

model. Now if, for any $E' \in \mathcal{E}^*$ we have that $\{E', E_1\}$ is inconsistent, then the agent has found an attack on R_1. Simply by considering all the elements in C_R as bdus in an argumentative theory Γ and adding $(\{E_2 \wedge E_3 \rightarrow E'\} : E_2 \wedge E_3 \rightarrow E')$, it has an argumentative theory that allows for conflicting arguments. This gives it motivation to challenge R_1.

In general, for any argument of the form $\varphi_1 \wedge \cdots \wedge \varphi_n \rightarrow \psi$ with $\varphi_1, \ldots, \varphi_n, \psi \in \mathcal{L}_{Rep}$, which we call a trust rule, the receiving agent can try to find an attacker, or counterargument, by finding the set $\Psi^* = \{\psi' \mid \{\varphi_1, \ldots, \varphi_n\} \vdash \psi'\}$ and testing whether $\psi' \in \Psi^*$ such that $\{\psi', \psi\}$ is inconsistent. If there is, then construction of the argument is trivial (and unnecessary) and the agent has a good reason to challenge that trust rule. If it cannot construct such an argument that attacks the trust rule, it can still decide to challenge it (the protocol does not limit this, after all), but the agent might be better served challenging any one of the evaluations $\varphi_1, \ldots, \varphi_n$ to find where the trust model needs adapting.

In our example we omit the expansions of nodes E_2 and E_3 because the resulting subtrees are similar to the argumentation for the root, E_1. Instead we focus on the expansion of rule R_1. The seeker challenges R_1 and the supplier responds with its justification: $\{P_1, R_2\}$. Upon receiving this argument the seeker starts the decision process in Figure 7.3 again.

The reasons for a trust rule are clearly defined in $\mathcal{L}_{Arg}$. They are priorities over the criteria and a bdu that represents the dependency of the trust rule on these priorities. In the example there is only one priority, P_1, that influences the calculation of a trust evaluation from reputation and image. The first step in the protocol is once again to decide whether to accept or reject the argumentation, this time supporting node R_1. The seeker's trust model provides a way of deciding to reject the argument: if instantiating its trust model with priority P_1 does not allow it to compute E_1 from E_2 and E_3, then the agents' underlying algorithmic methods are too dissimilar for the supplier to provide a personalised recommendation. Despite both agents using the same priorities to instantiate the parameters of their trust models, they compute different evaluations from the same input. In this case the dialogue ends in failure: the seeker should reject recommendations from the supplier and try another agent. Just as in Pinyol's framework, this is still useful information: the agents know that they disagree and that, in this situation, agreement is impossible.

If, in contrast, the seeker can emulate the supplier's trust calculation by using its priorities, then the only possible reason to not accept the trust rule outright is because the seeker disagrees with at least one of the priorities in the argumentation. The seeker can select such a priority and choose what to do. The protocol offers two options. The first is to challenge any of the priorities in the supplier's commitment store. The second is to counter and propose using its own priority instead. Note that the protocol of Section 7.4.1 allows an agent to explore both possibilities. This is also the case in the diagram of the decision process of the agent: if at a later stage in the dialogue the agent reaches "Try other node" it can try the alternative approach. The example of Figure 7.2

continues as if the seeker chooses to challenge P_1, but the approach of countering is equally valid and is described in Section 7.4.2.

Reasoning about the supplier's priorities

The dialogue continues with the supplier providing the justification for the priority that was challenged. In the example, the reasons for the supplier having priority P_1 are in the 4th level of the argumentation tree of Figure 7.2.

The reasons for prioritising one criterion over another, are given by the priority rules of AdapTrust, which are adopted as bdus in $\mathcal{L}_{Arg}$. These priority rules are supported by the agent's beliefs or goal, or the role it requires the target to perform. If the priority is supported by beliefs, as in the example, the protocol defines four possibilities:

1. The seeker chooses not to add the priority rule to its system. In this case its trust model will continue to be based on different criteria from the supplier's. It can try backing up in the dialogue and countering the supplier's priorities and proposing its own alternatives.

2. The seeker agent tries to add the priority rule to its system. This rule does not conflict with the rules it already knows. In this case it can be seen as a gap in the seeker's knowledge and it can choose to adopt this rule.

3. The seeker agent tries to add the priority rule to its system and this rule does conflict with the rules it holds. In this case the agents have found a context in which agreement is impossible: the cognitive underpinnings of their trust models are different in this situation. The seeker agent should reject recommendations from the supplier in this context.

4. The agents enter a separate persuasion dialogue in order to convince each other about the validity of their beliefs. This can be done using a state-of-the-art argumentation framework for persuasion, such as the one proposed by Prakken [2009].

Priority rules can also have a goal or role in the antecedent, which are treated similarly, although the option for a persuasion dialogue is then not present. Conflicts among priority rules are defined as follows:

Definition 7.11 (Conflict of Priority Rules)**.** Let $U \subseteq \mathcal{L}_{Rules}$ be a set of priority rules such that:

1. $\Pi = \{\pi' | (belief\_rule(`\Phi', `\pi', v) \in U\}$ is satisfiable in $\mathcal{L}_{PL}$

2. the set Φ is satisfiable in $\mathcal{L}_{Bel}$, with Φ defined as the union of all Φ', such that $belief\_rule(`\Phi', `\pi', v) \in U$

Then a priority rule $belief\_rule(`\Psi', `\pi', v)$ *conflicts directly* with U if and only if $\Pi \cup \{\pi\}$ is unsatisfiable and either $\Phi \models \Psi$ or $\Psi \models \Phi$.

A set of priority rules $\Xi \subseteq \mathcal{L}_{Rules}$ *conflicts* with a rule ξ if there is a set $U \subseteq \Xi$ that conflicts directly with ξ.

This definition states that a priority rule ξ conflicts with a set of priority rules Ξ, if ξ is in direct conflict with any subset of Ξ. For direct conflict we only consider subsets U of Ξ whose conclusions result in a satisfiable $\mathcal{L}_{PL}$-theory. Direct conflict occurs between U and ξ, if the antecedent of ξ is entailed by the antecedents of U (or vice versa) and the addition of the conclusion of ξ to the conclusions of U results in an unsatisfiable $\mathcal{L}_{PL}$-theory.

Note especially that rules do not conflict with a set of rules if their antecedents are merely consistent with those of the set, but only if the former's antecedent is entailed by the latter's, or vice versa. This is because two consistent antecedents with different conclusions might be designed to trigger in different situations, which is, after all, dependent on the beliefs and goals an agent has. In the case of two rules with conflicting conclusions triggering, AdapTrust contains a mechanism for choosing a consistent set of priorities (see Section 6.4.1 on page 135). Definition 7.11 only defines conflicts for priority rules over beliefs. For goals and roles it is the same, but then it is simply that a single goal, or role, leads to a conflicting set of priorities.

Reasoning about the seeker's priorities

If, instead of continuing the argument about the supplier's priority, the seeker proposes an alternative priority, the roles in the dialogue are switched. Now the supplier needs to discover why it should accept the seeker's priority. The same decision tree, in Figure 7.3, is used, but now the supplier performs the choices on what arguments, put forward by the seeker, to challenge. Note that there are always less options because, using our $\mathcal{L}_{Arg}$, the reason for having a priority cannot be a trust evaluation or a trust rule. Note that the supplier also has the possibility to accept a priority rule into its knowledge base, but, unlike the seeker, can do this only temporarily: it may do this with the sole purpose of calculating a personalised trust evaluation for the seeker and its goal.

If at any point in the dialogue, either agent has adapted its trust model, they should restart the dialogue in order to verify that they have reached agreement and the supplier is able to provide personalised recommendations.

7.5 Experiments

In Section 7.3 we described a new argumentation framework for discussing personalised trust evaluations and in the previous section presented a dialogue protocol for communicating such arguments. We now compare this model of communication to Pinyol's argumentation framework [Pinyol, 2011]. We have implemented AdapTrust using Jason [Bordini et al., 2007]. In order to make a fair comparison, we keep everything as similar as possible to Pinyol's experimental evaluation, so we use the trust model Repage [Sabater et al., 2006] and run the experiment in a simulated e-commerce environment, in which we evaluate the accuracy of buyers' trust evaluations of the sellers by using three methods of communication: (1) accepting other agents' trust evaluations directly (no ar-

gumentation), (2) filtering out mismatched communication with argumentation (Pinyol's system) and (3) our model for communicating personalised trust evaluations. Just as in Pinyol's experimentation, agents always respond truthfully and we do not consider scenarios in which deception is possible.

7.5.1 The simulation environment

The simulation environment initially runs 20 agents who need to buy an item from any one of the 40 sellers in the environment, as in Pinyol's simulation. The sellers in this environment offer products with a constant price, quality and delivery time. These aspects of the product are used to evaluate the trustworthiness of the seller. A buyer can be "frugal", in which case it gives more importance to the quality of the product than to the price or delivery time. A buyer can also be "stingy", in which case it evaluates price as being more important than delivery time or quality. Finally, a buyer can be "impatient", in which case the delivery time is the most important. The buyer profiles are implemented using AdapTrust's priority rules, based on the beliefs of the agent.

In addition to these basic profiles, the buyers can have different goals. We have implemented the goal to buy a bicycle, which is not associated with any priority rules, and the goal to buy milk, which must be delivered quickly and thus has an associated priority rule to prioritise delivery time over both quality and price.

These two types of priority rules and the different profiles and goals of the agents allow them to benefit from the full dialogue of Section 7.4. Agents can attempt to persuade each other to switch their basic profile. Because we rely on pre-existing persuasion dialogues for this, we have simply hard-coded the outcome. A frugal agent can persuade a stingy agent to change its profile (and thus become frugal as well): a good quality item allows one to save money in the longer term by not needing to replace it as soon. This serves both agents' underlying motivation of saving money. Furthermore, the different goals, and associated priority rules allow recommendation-suppliers to personalise their recommendation to the seeker's goal, as well as have the agents exchange priority rules for their goal.

The simulation environment runs for 40 rounds to initialise the environment. In each round the buyers buy an item from a random seller. To ensure that no single buyer can obtain full knowledge, by interacting with all the sellers, each buyer can only interact with a configurable percentage of the sellers. This percentage is thus a control on the amount of knowledge each individual buyer can obtain about the set of sellers. After buying, the buyers can communicate their trust evaluations to exactly one other buyer. Depending on the type of communication we wish to evaluate, they use no argumentation, Pinyol's argumentation, or personalised trust evaluations to perform this communication.

After this initialisation, we create a new agent, which is the one to be evaluated. This agent knows nothing about the environment. It is a frugal agent with a 50/50 chance to have either goal, to buy a bicycle or milk, the same as the other buyer agents in the system. However, this agent does not explore by

interacting with random sellers, but rather needs to discover the sellers' trustworthiness through communication with the established buyers. For this, it uses the configured communication model, no argumentation, Pinyol's model, or ours.

The results are plotted in Figure 7.4. The experiments were run with an equal distribution of sellers offering one of either good quality, price or delivery time. Similarly the buyers were equally distributed over frugal, stingy and impatient agents. The experiment agent was always frugal and had a 50/50 chance of having the goal to buy a bicycle or milk. On the x-axis is plotted the percentage of sellers each buyer can interact with directly during the initialisation. As explained above this is a measure of the knowledge each agent can obtain about the environment. With 20 buyers, 5% is the minimum to have all the sellers covered by at least one buyer. In this case, to obtain information about all sellers, information from all the buyers is needed. As the percentage of sellers each buyer can interact with increases, it becomes easier to obtain an accurate evaluation through communication, because the experiment agent needs to communicate with less of the established buyers to cover all the sellers.

The y-axis plots the average accuracy of the experiment agent's evaluation of all the sellers in the system. First it calculates the error of the experiment agent for each seller. The error is the difference between the agent's evaluation at the end of the experiment and the evaluation it would have had if it had been able to interact with the seller (and thus obtain perfect information). We convert these errors into percentages by comparing them to the "worst expected error". The accuracy of an evaluation is the percentual difference between its error and this worst expected error. The worst expected error is the expected error if both the estimated and most accurate evaluation were random. This is equal to the expected value of the difference two random variables from two standard uniform distributions, which is equal to the expected value of the standard left triangular distribution, or $\frac{1}{3}$ [Johnson and Kotz, 1999]. This error value is also assigned for any seller that the experiment agent has no information about. The accuracy of the experiment agent that is plotted on the y-axis, is the average accuracy for all the sellers $Sellers$ in the environment:

$$\frac{\sum_{s \in Sellers} 100\% \cdot \frac{1/3 - error(s)}{1/3}}{|Sellers|}$$

Each point in the graph is the average of 100 experiments with the error bar being 1.96 standard deviations (representing an expected 95% of the population).

7.5.2 Simulation results

The first thing we see when we look at Figure 7.4 is that both Pinyol's method and Argumentation + Adaptation converge to almost perfect results when the knowledge in the system is high. Without any kind of argumentation, however, the communication does not add anything to the accuracy. It does not filter out any communication, so even at 5% knowledge it is obtaining an evaluation for each of the sellers in the system. Nevertheless, because the communicating

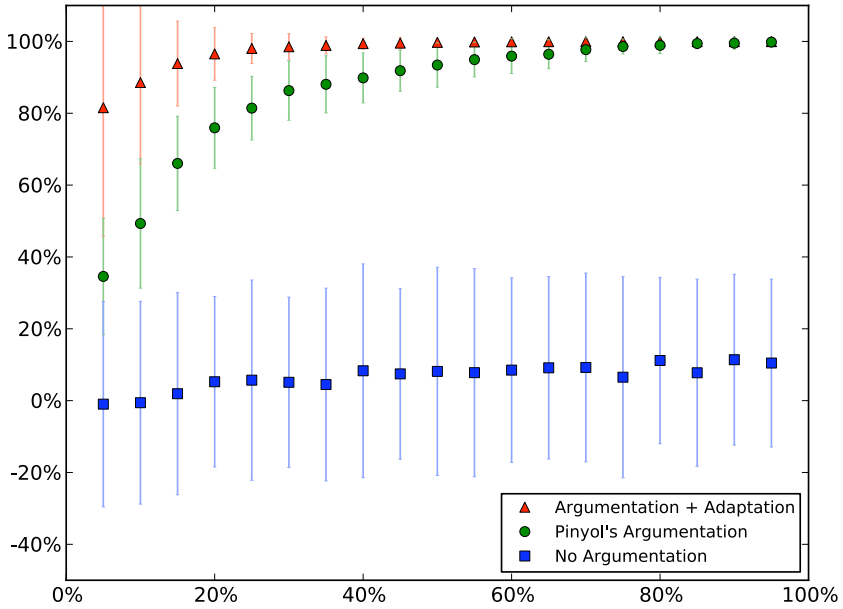


Figure 7.4: Experimental results. The x-axis represents the knowledge in the system and the y-axis the quality of the evaluation.

agent uses its own goal, and there is no opportunity to argue about underlying beliefs, these communications are inaccurate and it does not improve upon the baseline, the worst expected error.

In contrast, both Pinyol's method and Argumentation + Adaptation filter out information and obtain near-perfect results when the knowledge in the system is high both methods. When buyers, however, cannot interact with many of the sellers, and the knowledge in the system is low, then we see that Argumentation + Adaptation outperforms Pinyol's method. When a buyer can interact with 20% of the sellers, our method is still slightly over 20% more accurate than Pinyol's method in the experiment scenario.

7.6 Discussion

The experiment in the previous section is a proof-of-concept demonstration of the presented method for personalising communication about trust. Despite being a prototypical implementation, the experiment displays some interesting features of this method. Firstly, we once again confirm that some form of processing of the received communication is necessary. Without argumentation, the error introduced by communication is approximately equal to the worst expected error.

When arguing about the communication, whether through Pinyol's method or our own, the results immediately improve. Our method displays the greatest

gains over Pinyol's argumentation in scenarios where each buyer agent only has information about a few of the sellers, but Pinyol's method performs equally well as our own when buyers have information about a large number of sellers. This is to be expected, because in the latter case the experiment agent only needs to accept communications from a few of the buyers to get accurate information about all the sellers. This is very clear in the graph, where both Pinyol's method and our own converge on near-perfect accuracy.

We see, however, that our method performs better when buyers have little information, and we consider the situations when buyers can interact with less than 20% of the sellers. In these cases we feel that the increase in performance that our method offers over Pinyol's justifies its greater complexity and communication; the agents clearly benefit from being able to communicate, not just about trust evaluations, but about their goal and beliefs as well. By adapting their models, the buyers personalise their trust recommendation to the experiment agent, which accepts trust evaluations from a greater number of buyers and obtains significantly more accurate results.

Note that agents having had direct interactions with 20% of the providers of a service is already on the high side for many application scenarios for multi-agent systems, such as e-commerce, P2P or grid computing scenarios. Despite this, we do not claim that the results from this experiment carry over to other scenarios. We run the experiment with a uniform distribution of both sellers' qualities and buyers' criteria in a simplified representation of an e-commerce environment. Even in this simple environment, if we change the parameters, we see different results. Specifically, the less likely it is that the experiment agent finds agents who are like-minded, the more important it becomes for it to obtain personalised trust recommendations from agents whose evaluation would otherwise need to be discarded.

More experimentation is needed in more diverse scenarios to decide when personalised communication about trust offers useful benefits to the agents. This experiment's purpose is to demonstrate the method's functional viability and sketch the general domain in which we expect agents could use it.

Despite the experiment being based on a small and simulated scenario, it shows that even with just three parameters for the agents and two different goals, a filtering method will be left with too little information to work with, necessitating the use of a method such as the one proposed in this chapter. This simple scenario serves as a proof of concept for its application in more realistic scenarios, such as the following:

Automated e-commerce agents: the scenario we presented in the experimentation was a simplified e-commerce environment, but as the scenario is extended with more items and more properties of these items, the probability of coinciding with another agent decreases correspondingly. Therefore, despite there also being a far larger number of agents in the system, those with similar backgrounds to the own will still be sparse, necessitating a communication model such as the one we describe. If the community of sellers and buyers is relatively stable, then Trust Alignment can be a

possible approach, as we stated in Section 4.3.5, but if this is not the case then we provide Trust Adaptation as an alternative solution.

Helpful community assistance: the uHelp scenario of the CBIT project[2] proposes an automated, distributed, algorithm to help delegate human tasks, such as picking up children from school, within a community. Such tasks should only be delegated to trusted members of the community, but as in the scenarios above, the criteria on which the different members base their trust may vary greatly. In this case the argumentation does not serve as much to enable the communication about trust, but is rather a vehicle to find the criteria the community has in common for deciding whether someone is a trustworthy performer of a certain task. In this sense the argumentation about trust can be used to converge the community's disparate models of trust and strengthen the bonds within such a virtual community. We intend to explore the possibilities of convergence of trust through argumentation in future work.

7.7 Summary

In this chapter we presented an argumentation framework for acquiring personalised and more accurate communication about trust evaluations. The work extends the argumentation framework presented by Pinyol [2011] and uses AdapTrust (see Chapter 6) to build a justification for a trust evaluation. An agent asked to supply a trust recommendation builds such a justification up from its beliefs and goals, and can communicate this to the recommendation-seeker. We presented a formal dialogue protocol and discussed how to use this for communicating trust. This dialogue allows the two agents to argue about *why* their model calculates a trust evaluation, and using AdapTrust, they can adapt their models. This adaptation has two effects. The first is that both agents can accept new information that fill gaps in their knowledge. The second is that the recommendation-supplier can (temporarily) adapt its model to the recommendation-seeker's and personalise its trust recommendation. Such personalised recommendations can be used, in addition to reputation information, recommender systems, or other sources of information in the environment, to help agents choose partners in social environments.

We compare our method for personalising communication about trust to Pinyol's method. This is a state-of-the-art method for communicating about trust, but it relies on filtering out mismatched information using argumentation, rather than trying to personalise it. We demonstrate that, if the interests of the agents are diverse enough, and the number of different criteria that can be used to evaluate an agent is large enough, then, for direct communication about trust, the method presented in this chapter results in a more accurate estimate of an agent's behaviour than the filtering technique we compare it with.

[2] www.iiia.csic.es/en/project/cbit

Part IV

Comparison and Conclusions

Chapter 8

Conclusions, Comparison and Future Work

Let us think the unthinkable, let us do the undoable, let us prepare to grapple with the ineffable itself, and see if we may not eff it after all.

–Douglas Adams

8.1 Conclusions and Contributions

In the introduction, we asked ourselves how computational agents can communicate their subjective trust evaluations in a meaningful manner. In this thesis we have provided two different approaches to answer this question. The first way of communicating about trust in a meaningful manner is by performing Trust Alignment. By relating the trust evaluations back to the set of shared evidence that supports it, the receiving agent can learn the meaning of a communicated trust evaluation, in the context it was made. This allows the receiver to translate received evaluations into its own frame of reference.

The second way in which we answer the question is based on very different assumptions from the first. Whereas Trust Alignment requires a set of shared evidence, Trust Adaptation requires more reflection on the part of the agents. They need to be able to reconstruct the justification for having a trust evaluation. By communicating about these justifications, the agents can achieve personalised communication about trust.

These two different approaches answer the same question, in very different manners. In the next section we will discuss these differences in greater detail, but first we summarise the main contributions of this thesis.

8.1.1 Main Contributions

In the Trust Alignment approach, we consider the problem of communicating about trust as a problem of pragmatic heterogeneity. By formalising the problem using Channel Theory, a qualitative theory of information flow, we see that if two agents share knowledge of some evidence that supports their individual trust evaluations, they can align these evaluations. An alignment is a generalisation of so-called Specific Rules for Alignment (SRAs), which are Horn clauses containing an agent's own trust evaluation as the head, and the other agent's trust evaluation, together with a description of the shared evidence that supports both trust evaluations, in the body. To generalise from these SRAs to a predictive alignment, the description of the shared evidence should only use information that the other agent has communicated. To align, the sender can help the receiver by only comunicating relevant properties of the evidence, in a consistent manner.

We have implemented this solution, called FORTAM, and use the machine-learning algorithm Tilde, to learn an alignment. FORTAM performs well in the simulated scenario that we use to test it, and we compare it to a number of other alignment methods. A particular strength of FORTAM is that, in addition to translating subjective trust evaluations into the receiver's frame of reference, it works when the sender is intentionally deceptive. If the sender attempts to deceive, giving false trust evaluations based on the shared evidence, the alignment using FORTAM is not significantly affected. In contrast, if the description of the shared evidence is tampered with (for instance, by communicating irrelevant properties of the evidence), then FORTAM may not perform as well.

One of the drawbacks of FORTAM, or any alignment method, is that it requires a large amount of shared evidence to be able to learn. To communicate trust evaluations effectively in scenarios with little or no shared evidence, we propose Trust Adaptation. Whereas Alignment relies on external properties, the shared evidence in the environment, to talk about trust, Trust Adaptation uses the internal properties of agents. Trust Adaptation requires agents to explain how their beliefs and goals influence their trust evaluations.

To make this possible, agents must reason about their trust model: the computational trust model must be integrated into the cognitive process of the intelligent agent, and we present AdapTrust for this purpose. AdapTrust is an extension of the BDI-agent architecture, formalised in a multi-context system. We add contexts for reasoning about the trust model, using the beliefs and goals an agent has. Furthermore, we do this in a manner that allows an agent to reason about its trust evaluations and justify why it holds a specific evaluation.

We propose an argumentation language, extending the work done by Pinyol, to formalise the justifications of a trust evaluation, and to allow agents to communicate them. Specifically, the language allows agents to communicate about the priorities in their trust model, and what beliefs and goals caused them to have these priorities. We describe a formal dialogue protocol for this communication and demonstrate experimentally that if both agents use AdapTrust, and can communicate in a shared argumentation language, then a sender can per-

sonalise its communication to the receiver's needs. This, in turn, improves the accuracy of agents' trust evaluations.

8.1.2 Additional findings

Additional findings related to Trust Alignment

- By modelling the problem of Trust Alignment in Channel Theory [Barwise and Seligman, 1997], we can draw some parallels between Trust Alignment and other forms of alignment that are formalised using Channel Theory. Specifically, our model of Trust Alignment complies with the "Semantic-Alignment Hypotheses" of Schorlemmer et al. [2007]; however, the way both works describe the problem is the only similarity between them, and the actual methods of alignment proposed are different. Schorlemmer et al. show how ontology alignment can be seen as a form of refining the information channel[1], which aims at finding a shared meaning of the concepts involved. In contrast, we are not interested in a shared meaning: both agents should maintain their own, subjective, trust models, but find a *translation* between the two different models. For this we are better served using a machine-learning approach than following traditional methods for achieving semantic alignment.

  Such an approach may be useful in other situations where pragmatic heterogeneity is problematic. A specific area in which this might be useful is in learning the semantics of folksonomies. Folksonomies are socially created taxonomies, in which information and objects are tagged for one's own retrieval [Vander Wal, 2007]. According to Vander Wal, "the value in this external tagging is derived from people using their own vocabulary and adding explicit meaning, which may come from inferred understanding of the information/object." This may be so, but automating the retrieval or use of such information is problematic. Semiotics has been proposed as a way of understanding folksonomies and coupling them to semantic web ontologies [Saab, 2011]. This problem could be considered in our framework by viewing the information/objects as tokens and the tags as types. The task in this problem of alignment is to learn the meaning of the tags, with regards to their different uses in classifying information/objects.

- We introduce a complexity measure in order to decide whether an alignment can be learnt between two trust models; moreover, this measure allows us to estimate how many shared interactions will be required to learn it. This measure, essentially, quantifies the difference between two trust models based on the trust evaluations they compute. This means that we can measure the difference between two models without analysing the computational models themselves, but can simply use their output. This measure of the relative complexity of computational processes may

[1] For an explanation of refinement we refer to Schorlemmer et al.'s work, or, for a full theoretical treatment, to Barwise and Seligman [1997].

be useful in domains other than Trust Alignment, such as in matching elements in a recommender system, or in a clustering algorithm.

Additional findings related to Trust Adaptation

- In order to formulate Trust Adaptation properly we designed a new agent framework in which to integrate trust models. This framework is useful in and of itself, even if the agent does not argue about its trust evaluations. The priority rules cause the trust model to change, if the agent's beliefs about the environment change. Possibly more impactfully, the priority rules allow for a fully goal-orientated model of trust; this is necessary for a proper treatment of trust [Castelfranchi and Falcone, 2010]. As discussed in Section 5.2, the incorporation of trust into cognitive agents has not received much attention. Work on the computational trust models we discussed in Section 2.2, for instance, do not go into details on how this incorporation should happen. AdapTrust allows all kinds of computational trust models to be incorporated into a cognitive agent, in a theoretically sound manner.

- One aspect of trust that we have not focused on in this thesis, is that trust is not merely an evaluation of a target, but a *decision* to trust, or not to trust, the target. The justifications an agent has for its trust evaluations may be useful in making this decision. The primary aim of our argumentation framework for justifying trust evaluations is to allow agents to communicate their trust evaluations, but a secondary effect may be that the justification can be used in the decision process. AdapTrust provides a way for the trust evaluation to be adapted to the needs of the agent, and the justification, in a formal logic, allows the agent to verify that its needs are indeed taken into account in the trust evaluation. This allows the agent more control over the pragmatic aspect of trust. Justifying trust in AdapTrust could serve in a similar fashion as the fuzzy cognitive maps of Falcone et al.'s system [2005] or the BDI+Repage model [Pinyol et al., 2012]: it provides a way of deliberating about trust evaluations in the decision process.

8.2 Comparing Adaptation to Alignment

In this section we aim to compare the two different approaches we have described in this thesis for communicating trust. We start with a brief overview of the characteristics of the two approaches, their principal differences and their similarities.

The first, and probably foremost, difference between Trust Alignment and Trust Adaptation is that the latter is a form of pre-processing the trust evaluation, whereas the former is a form of post-processing. Both require some communication prior to the witness providing its trust evaluation of a target; however, what happens with this evaluation is entirely different in both methods.

	Trust Alignment	**Trust Adaptation**
Quantity of interactions required	High	Low/None
Quantity of communication with witness	High: agents must communicate about all shared interactions	Medium: agents must argue about all aspects of the computation of a single evaluation
Complexity of communication with witness	Low: witness simply communicates its trust evaluation and the relevant aspects of the domain	High: witness must construct an argument supporting its trust evaluation
Complexity of agent model	Low: trust model can be treated as a black box	High: requires a cognitive agent, both for introspection into the trust model and for adaptation
Complexity of processing a received recommendation	High: must learn an alignment that can translate an incoming recommendation	Low: just requires an information-seeking dialogue to confirm the adaptation has succeeded
Ability to deal with deception	Excellent, as long as deceptive agents communicate truthful domain information	Dependent on how good the witness is at inventing a consistent support for its trust evaluation
Ability to deal with dynamic environments	Poor: the agent needs to relearn an alignment any time the environment changes	Excellent: adaptation to a changing environment follows naturally from the system

Table 8.1: Comparison of the features of Trust Alignment and Trust Adaptation

In Trust Adaptation the communication serves for the recommendation-seeker to tell the witness its goal and priorities, and argue about beliefs. If this has succeeded, then upon receiving a recommendation, the seeker might enter an information-seeking dialogue to ensure the trust evaluation can be accepted as is, but this is merely a confirmation that the pre-processing has been performed correctly. On the other hand, in Trust Alignment the recommendation-seeker has to learn an alignment, based on prior communications, with regards to shared information. The alignment then serves to post-process the recommendation-supplier's trust evaluation by translating it into the seeker's frame of reference.

Other differences between the two approaches of Trust Alignment and Trust Adaptation can, for a large part, be related back to this fundamental difference. The main features of both approaches are summarised in Table 8.1. In Section 8.3.3 we discuss some ideas on how the two approaches can be combined, but first we discuss the main features of both approaches, and their differences, in detail.

8.2.1 Environmental considerations

Pre-processing a trust evaluation has different requirements from post-processing. In pre-processing, the witness must adapt its recommendation to the requirements of the recommendation-seeker, whereas in post-processing there is no such onus on the witness and the recommendation-seeker must find a way to use the "raw" information provided by the witness. There is no clear-cut way for deciding which method is better, but we can analyse the differences by considering the environment the agents must function in and the type of agents involved.

The first matter to consider is the number of shared interactions among agents. We recall that FORTAM obtained its best results in our example scenario when it had at least 500 examples. A scenario with a more expressive domain language will need more examples, while a scenario with a simpler language will require less. The reason for this is that increasing the expressivity of the domain language increases the space in which we need to find a hypothesis, and the number of samples required to learn an alignment is directly related to this [Russel and Norvig, 2010, pages 668–670]. This is also the reason why using no domain language, as, for instance, in AR&H's method, requires less than 100 interactions to reach peak performance. Nevertheless, whichever method for Trust Alignment is used will require some set of shared interactions. The more expressive the language, the finer the distinction among different types of interactions can be made, and thus the more examples a machine learning algorithm requires to learn an alignment.

However, this is not the whole story: firstly, this should be seen in the light of a probably approximately correct (PAC) alignment: the higher the complexity of the hypothesis space, the more examples are required to obtain the same probability that the alignment learnt is approximately correct (we refer to Russel and Norvig [2010] for the formalisation). Secondly, the language may be far more expressive than required to discuss relevant aspects of the trust model, in which case the part of the hypothesis space that needs to be searched can be small, despite the expressivity of the language. Finally, the relative complexity between trust models (as discussed in Section 4.2.4) has a large impact on the number of interactions required to learn an alignment.

Trust Adaptation, on the other hand requires no shared interactions at all. Rather, it places more restrictions on the communication language. Trust Adaptation requires that the agents share a language in which they can express their beliefs and goals, as well as their trust priorities, and consequently their priority rules. The language must, therefore, be more expressive than a language for exchanging objective properties of interactions. Some languages may be expressive enough to communicate some goals and beliefs, but not all, allowing for partial adaptation. Similar to the case for Alignment, this is a simplification of reality: while a sufficiently expressive language is a prerequisite, it is not always the case that a more expressive language allows agents to better express their beliefs and goals. Sometimes, even a very expressive language may not cover the specific needs of an agent, while a simple language might be sufficient for

expressing some goals and beliefs that allow for at least a partial adaptation.

We have considered the environment along only two dimensions, the expressivity of the communication language and the number of shared interactions. While this is a simplification, it allows us to distinguish, roughly, four different cases:

- If there are too few interactions for Trust Alignment to succeed and the language does not allow agents to express themselves sufficiently for Trust Adaptation, then the methods we have presented in this thesis for communicating about trust are insufficient.
- If there are enough interactions to learn an alignment, given the complexity of the language, but the language is not expressive enough for Trust Adaptation, then Trust Alignment provides a solution for talking about trust.
- In contrast, if the language is expressive enough for Adaptation, but there are too few interactions for Alignment, then Trust Adaptation can be used to talk about trust.
- Finally, in the, for us, most interesting situation, there are both enough shared interactions for enabling Trust Alignment, and the language is expressive enough to use Trust Adaptation, then other criteria, such as the specifics of the agents, can be used to decide what method to use. In Section 8.3.3 we discuss how the two methods might be combined, but first we discuss some of these other factors.

8.2.2 Complexity of the agents

Whether Trust Alignment or Trust Adaptation are viable depends for a large part on the agents involved. Especially in the case of Trust Adaptation, this is very clear: both agents involved must be able to examine their own trust models and explain them to each other. Moreover, the recommendation-supplier must be able to adapt its trust model, if necessary, to the priorities of the recommendation-seeker. Finally, both agents must be able to argue about the beliefs underlying the priorities in their trust models and adapt their models to changed beliefs. In addition to the capabilities of the agents, we assume the recommendation-supplier is not only truthful, but benevolent: it is not only able to adapt its trust model, but willing to do so.

For Alignment the dependencies are less on the agent architecture, but more on the ability to learn a translation from the alignment messages. This depends firstly on the learning algorithm used, and secondly on what the sender chooses to send. While we showed that FORTAM deals well with agents sending false evaluations, we still require the description of the domain to be accurate. For this we refer back to Section 3.4, in which we stated the requirements that the descriptions of interactions be relevant and consistent. An additional limitation of Alignment is that, in contrast to Adaptation, it focuses solely on the evaluations of single interactions.

If we consider this from the three-level approach by Staab and Engel [2008], that we discussed in Section 2.3, we see that Trust Alignment provides a method for communicating at the second level: the subjective evaluation of individual pieces of evidence. In contrast, Adaptation provides a method for communicating at the third level: trust-values based on the aggregation of the subjective evaluations. Given the increase of complexity and subjectivity between these levels of communication, it is not surprising that the requirements for Adaptation are greater than for Alignment.

Despite this, we feel that any intelligent agent that functions in a social environment must naturally have a trust model; furthermore, it must be able to explain to some extent or another why it trusts other agents. The integration of a trust model into a framework for intelligent agents is thus not so much a requirement of our method for Adaptation, but a requirement for acting intelligently with regards to trust. We recall Castelfranchi and Falcone's definition: "trust is (conscious and free, deliberated) reliance based on a judgement, on an evaluation of Y's virtues, and some explicit or entailed prediction/expectation". Such deliberation is, in our opinion, a requisite of trust, and while we can create computational trust models that give excellent estimates of the trustworthiness of agents, it is not until such estimates are used in an agent's decision process, that we can say that an agent trusts.

Such, more philosophical, considerations aside, current agent technologies are generally not equipped with the capabilities required for Trust Adaptation. This means that, at least in the short term, Trust Alignment is easier to apply than Trust Adaptation — if only because Alignment can be performed in a separate module, whereas Adaptation requires a new agent model, such as the AdapTrust model we presented in Chapter 6.

8.2.3 The cost of communication

The amount of communication between the recommendation-supplier and recommendation-seeker is the last vector along which we compare Trust Alignment and Adaptation. The amount of communication in the alignment process is dependent on two factors: the number of shared interactions and the amount of relevant information about each interaction that is communicable in the domain language. This must be communicated every time an alignment needs to be learnt, but, when this is done, further communication is simply an evaluation and a description of the interactions that support it. In practice, however, we recall that it is hard to learn an alignment from aggregated evaluations and both FORTAM and BLADE, the only two alignment methods that take the context into account, both learn at the level of individual interactions. Therefore, even once the alignment is learnt, agents must communicate their evaluations of, possibly many, individual interactions, rather than the trust evaluation that is the result of aggregating these evaluations.

For using Trust Adaptation, we expect that, on the whole, the communication load is lower, although it is less easy to predict how much communication will pass between the recommendation-seeker and recommendation-supplier. While

all argumentation dialogues are guaranteed to end, it is dependent on the agents' trust models and choices within the argumentation dialogue, how much will be communicated every time a recommendation is required. Specifically if the agents try to convince one another about their beliefs, the amount of communication may be quite large. On the whole, we need detailed information about how large the argumentation tree is, and how much of it the agents actually explore, to give an estimate of the amount of communication.

A major advantage of Adaptation over Alignment, however, is that if the amount of communication possible is restricted, or bandwidth is costly, then agents can make adjustments in their strategies in the argumentation in order to restrict communication to a minimum, or make a trade-off between expected gain from exploring part of the argumentation tree and the communication cost of that exploration. In contrast, alignment methods require a minimum number of examples to be communicated in order to learn anything, and if this communication is prohibitively expensive, then learning a good alignment is flat-out impossible.

In addition to the amount of communication, there can be risks involved with communicating. The first is primarily to the recommendation-seeker. If the recommendation-supplier is intentionally deceptive, then the seeker may be misled into thinking that a communicated evaluation is acceptable, whereas the agent is being deceived. In the case of Alignment, we have demonstrated that the process is relatively safe to deception in and of itself: if the information about the interactions is truthful, then deceptive trust evaluations are not really a problem. Generating believable lies about shared interactions is far harder than lying about trust evaluations, but, if this is also done, then the recommendation-seeker can still rely on its trust model, as in a system such as LIAR [Vercouter and Muller, 2010], to decide whether the communicated evaluation is reliable. Using Adaptation, deception is a greater problem: we require the recommendation-supplier to be benevolent and truthful, and we rely on the agent's trust model to find such a benevolent recommendation-supplier.

The second risk with communication is a risk to both agents involved. For either alignment or adaptation the exchange of possibly privacy-sensitive information is required. Agents may be unwilling to disclose such information. Here again, the trust model itself can be of some help: if the agent trusts the communication partner to protect, and not misuse, such privacy-sensitive information, then it will be more willing to disclose it. Additionally, even if agents do not trust each other with privacy-sensitive information, both Trust Alignment and Trust Adaptation allow the agents to limit the information exchanged. In Adaptation, either agent can choose to end the dialogue at any point if it does not want to disclose more information. In the worst case this leads to failure of the adaptation process, but it may very well be that the agents are willing to exchange enough information to succeed, even if they are unwilling to disclose all information.

In Alignment a similar limitation to the information disclosed is allowed in the domain description of interactions. We require the witness to provide a relev-

ant and consistent description of the interaction, but, other than that, the agent is free to decide what to communicate. In the worst case, the agent will choose to communicate no, or not enough, domain information for the alignment algorithm to improve on Abdul-Rahman and Hailes' method [2000]. Similarly, the best case, possibly resulting in a perfect alignment (dependent on the learning algorithm used), is attainable only if the sending agent communicates all relevant information consistently. Between these two cases there are many intermediate alignments that can be learnt, if the sending agent is willing to communicate some, but not all of its relevant information, and is possibly not always consistent.

In closing, it is our opinion that both Adaptation and Alignment deal adequately with deceptive agents and problems arising from privacy-sensitive data. Especially because the trust model itself can be used as a tool to decide whether to rely on the communication, if the agent is in doubt.

8.2.4 Prototype applications

In Sections 4.3.5 and 7.6 we discussed some possible applications of Trust Alignment and Trust Adaptation, respectively. Notably we mentioned e-commerce as an application domain for both methods. With this we do not mean that both methods can always be applied: there are many different types of e-commerce applications and some may be better suited to Trust Alignment and others to Adaptation. One thing, however, that e-commerce applications have in common, is that communication is generally desirable, and agents in the environment are willing to help each other. Moreover, the differences in e-commerce scenarios cover many of the points we brought up in the previous sections. We will highlight this in two different applications and briefly compare them below.

Trust Alignment: personalised reputation ratings

In the case of an open market, where thousands, if not millions, of people buy and sell their goods, Trust Alignment is more likely to succeed. Generally, the infrastructure is available for the, possibly limited, observation of other agents' interactions, and by considering such, publicly available, information as shared interactions agents could learn an alignment. Trust Alignment could be used to to tailor reputation ratings, such as provided by online marketplaces, like eBay[2] or Amazon[3], to the user. Both eBay and Amazon have enormous databases of binary or numerical feedback that has been provided by users — often with detailed textual explanations, such as the following, which was left as feedback for a seller on Amazon:

> "Item damaged in transit, so it was returned.. but the seller was very prompt about taking care of the problem. I appreciate their efforts

[2] http://www.ebay.com
[3] http://www.amazon.com

> to track the missing package down. and if have the opportunity, will order from them again."

If each user were to have a profile in which he or she could indicate the importance of certain factors to a satisfactory trade, then this could be seen as a rudimentary trust model. Such a rudimentary trust model could serve as a basis for aligning with other users. by considering the textual feedback as a description in the domain language, we could use natural language processing to generate examples for a machine learning algorithm. The learnt General Rules for Alignment could be used to translate other users' reputation ratings, and thus personalise them for the user. Note that this does not even have to be done by eBay or Amazon; the alignment could be performed by an application on the user's computer. This also means that the user does not have to disclose his preferences, even to the trading site.

Trust Adaptation: an automated financial adviser

Trust Adaptation may be more suited to situations in which agents are unwilling to share any information about their past interactions, but due to the risks involved with interacting, are willing to communicate about their goals and beliefs about the environment. A potential domain is in financial markets: additional features of such domains that match with the strengths of Trust Adaptation are that there are generally trusted agents, but the number of possible factors to be taken into account when interacting is high. While Adaptation is not as directly applicable as Alignment, we identify automated investment advice as a potential candidate. Investment advice is an expert job. The task is to help with buying or selling a financial product, such as shares, bonds or equity. The overall goal is usually to make money; however, there are different risks involved with all possible products, and correspondingly different possible rewards. Assessing the right combination of financial products is thus an interplay between a human expert, the financial adviser, who can correctly assess the risks and rewards (possibly only for a subset of financial products) and the investor, who has specific desires (such as a low-risk, long-term investment, a high-reward, short-term investment, or even investments in anything that is not related to the oil- or weapon-industry).

A personalised recommendation to buy a specific financial product is thus the result of an expert taking the specific beliefs and goals of the investor into account, and possibly, convincing the investor that his or her beliefs regarding certain financial products are wrong. In reality, experts do not always agree, because economics is a complex and chaotic subject, and an investor may benefit from talking to various different financial advisers, and weighing their justified opinions, before choosing how to invest.

An automated investment adviser, using Trust Adaptation, would be a personal agent that has knowledge of the user's beliefs and goals with regards to investments. We envision expert services providing financial advice, just as banks' websites do currently. Trust Adaptation could allow the personal agent

to argue with the expert services, allowing these to adapt their recommendations to the specific needs of the user, as represented by the personal agent. This combination of a personal agent and expert services could thus automatically give a user investment advice, obtained from multiple sources and tailored to his or her specific needs.

A brief comparison

The application of Trust Alignment in personalising reputation ratings takes advantage of the availability of very many, publicly available, interactions. While Adaptation, or even a collaborative filtering approach, as in contemporary recommender systems [Ricci et al., 2010], could be used to a similar end, such approaches would require users' profiles to be disclosed, and they do not take advantage of the large amounts of data already available. Applying Trust Adaptation to create an automated financial adviser, in contrast, would work in a scenario in which users are more likely willing to discuss their beliefs and goals, rather than details of past investments. As we mentioned in Section 8.2.2, the technology required for Adaptation is more advanced than Alignment. Thus, while the former application could be implemented using today's technological infrastructure, the latter depends on future advances.

Regardless of currently available technology, each application plays to the strengths of the approach we distinguish in Table 8.1. Personalising reputation ratings takes advantage of the high number of interactions and the textual feedback of them. Despite being written in natural language, such feedback is generally not terribly complex. Moreover, if the alignment is performed while the user is idle, then the complexity of the learning task does not bother him or her. Finally, the ability to learn an alignment could deal with deceptive feedback, such as retaliatory negative ratings, a known problem of eBay's reputation system [Klein et al., 2009].

An automated financial adviser, however, would not have access to a large number of other agent's interactions, and would make up for that by communicating with experts about why certain financial products can be trusted, given a set of beliefs and goals. Because financial experts are considered trusted, deception is not a problem, but the ability to deal with a dynamic environment is essential to function in an environment as volatile as modern financial markets.

8.3 Future Work

Communication about trust is a new and exciting field to work in. The work presented in this thesis can easily be extended in many different directions and we describe a few of these here.

8.3.1 Trust Alignment

Our work on Trust Alignment is divided into the theoretical description and the practical implementation. There is plenty to be done in both areas. Here we present a short list of topics that we feel deserve special attention.

Extending the theoretical framework

- An important extension of the theoretical framework is to allow the trust classifications and the channel to model dynamic trust models. If agents change their trust models over time to adapt to a changing environment (such as is the case for AdapTrust), or a trust model calculates evaluations based on more than just the agent's observations of interactions, then interpreting a trust model as a classification between observations and trust evaluations is insufficient. The tokens for a classification representing such a trust model must include something more than just the set of interactions (or observations thereof). This extension of the classifications and trust channel, in order to deal with dynamic trust models, can be done in the scope of Channel Theory and we propose to extend the particular instantiation of Channel Theory that we present in this thesis.

- We have identified relevance-maintaining and consistency as necessary properties for the translation between an agent's internal observation language and $\mathcal{L}_{Domain}$. If the translation is a Galois connection then these properties hold, but the reverse is not true. If we add what we tentatively call "reverse relevance"[4], then it does follow (trivially) that the translation is a Galois connection; in fact, we do not even need consistency in this case. It is interesting to discover uses for reverse relevance, or a different, minimal condition that, together with consistency and maintaining relevance, ensures a Galois connection.

Future work in the practice of Trust Alignment

- The experimental analysis of FORTAM is far from complete. We expect that the accuracy of the alignment possible with FORTAM depends, in a large way, on the domain language used. In our opinion the first step in a more thorough analysis of FORTAM should thus be to experiment with different domains and their corresponding languages. Secondly, we have shown in Section 4.2 that the trust models themselves have a large influence on the accuracy of alignments. Analysing the interplay of some different trust models, together with more — or less — descriptive domain languages for describing the context, is an important concern for future research. Perhaps, however, it is time to put FORTAM to use and analyse its functioning in application scenarios. Especially interesting would be to see how well it compares to BLADE [Regan et al., 2006] in such a scenario.

[4] using the same notation as in Definition 3.26 on page 62, reverse relevance is if $(H^{down} \circ H^{up})(\Psi) \in \rho_\theta(\Psi)$ holds.

- One problem of FORTAM is that it requires a large number of shared interactions and a lot of computation to learn every alignment. It would be very useful if we could somehow reuse alignments with agents that are similar to those we have already aligned with. Case-Based Reasoning [Aamodt and Plaza, 1994] seems like an ideal candidate to perform such reuse of alignments. It may, however, be hard to decide what agents are "similar". An approach could be to use social-network analysis, but another alternative might be to use our difference measure for trust models. If the complexity between two agents' trust models is low, the same alignment, possibly with modification using AR&H's method afterwards, can be applied.

8.3.2 Trust Adaptation

Improving AdapTrust

- AdapTrust is an abstract model and makes no commitment to the method in which the priority rules are created. In our implementation for personalising trust recommendations, we designed these priority rules manually, but ideally such rules are formed by some type of common-sense reasoning or machine learning. For instance, if the agent beliefs its environment contains lots of liars it should clearly trigger a rule not to rely much on communicated evaluations. A designer may include such common scenarios, but probably cannot think of all eventualities. The agent, therefore, will adapt in an unsatisfactory manner. A learning module that recognises that its trust model is badly adapted to the situation and tries to rectify this by adding new rules may provide a solution.

- AdapTrust made a start in reasoning about the trust model. We focused on using the trust model's parameters to adapt the model; however, it can be imagined that adapting the computational process itself might be desirable. There is no reason why the priority system could not be connected to such computational processes instead of parameters, although an appropriate representation language for such a computational process must be found. This adaptation is more complex than the adaptation we proposed in this thesis, but it might be useful; especially if agents want to personalise their trust evaluations for agents using very different trust models.

Future work in personalising trust recommendations

- The dialogue protocol that we propose does not allow agents to persuade each other about the correctness of their priority rules. For this, an agent should be able to recognise that its trust evaluations, based on the current set of priority rules, are badly adapted to the environment and goal. As opposed to above, however, the agent would not learn new priority rules by itself, but could copy them from fellow agents. A persuasion dialogue that allows agents to convince each other of their priority rules may improve the adaptability, and thus the ability to further personalise trust evaluations.

- We recognise that the method we presented for personalising trust recommendation requires more extensive experimental evaluation. We regard such an evaluation as future work and intend to use personalised trust communication in different, realistic, scenarios. We intend to compare Trust Adaptation to contemporary recommender systems or the use of reputation, to give a more precise indication of what applications will truly benefit from this model.

8.3.3 Combining Trust Alignment and Adaptation

In addition to work on improving Trust Alignment and Trust Adaptation in isolation, we feel that an important track of future research is to combine the two approaches. In the previous section we discussed the strengths and weaknesses of both approaches and we see that they could very well complement each other. We discuss some possible approaches.

- The most straightforward manner of combining Trust Alignment and Adaptation, is by simply giving an agent the possibility to use both. The agent then needs some decision support system for choosing when one approach is more appropriate than the other: for instance, by considering the number of interactions that it shares with the other agent, the expected benevolence of that agent or other criteria for both systems that we discussed in the previous section. After choosing an approach to communicating about trust, it executes it, and in the case of failure, can always evaluate whether the other approach might be viable after all.

- A more intricate manner of combining the two is in a true hybrid system. We propose two possible approaches.

  - If the agent uses FORTAM as its primary method for communicating about trust, the output includes a confidence in each case for the translation. If the agent wants to communicate with the witness regarding a target with many interactions that fall within a case of low confidence, it could ask that witness to personalise its trust recommendations and enter an argumentation dialogue. Alternatively, it could, before even asking about a target, ask to enter an argumentation dialogue to discuss these cases and see if it can adapt its model, or convince the witness to adapt its model. Such adaptation should improve the alignment, by removing the "difficult cases".
  - Alternatively, the agent can function the other way round, using adaptation as its primary method for communicating about trust. As discussed in Section 7.4, there are situations in which the adaptation dialogue simply fails; either because the agents refuse to adapt, or because their underlying computational methods are too different. In these cases, the agent might be able to fall back on Trust Alignment as a secondary method for communicating.

Bibliography

Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1): 39–59, 1994.

Alfarez Abdul-Rahman and Stephen Hailes. Supporting trust in virtual communities. *Proceedings of the 33rd Hawaii International Conference on System Sciences*, 6:4–7, 2000.

Leila Amgoud and Henri Prade. Using arguments for making and explaining decisions. *Artificial Intelligence*, 173(3–4):413–436, 2009.

Manuel Atencia and Marco Schorlemmer. An interaction-based approach to semantic alignment. *Journal of Web Semantics*, 12, 2012. doi:10.1016/j.websem.2011.12.001.

Francis Bacon. Of counsel. In Brian Vickers, editor, *The Essays – or Counsels, Civil and Moral*. Oxford University Press (1999), 1625.

Jon Barwise and Jerry Seligman. *Information Flow: The Logic of Distributed Systems*. Cambridge University Press, 1997.

Marc Bekoff and Jessica Pierce. *Wild Justice – The Moral Lives of Animals*. University of Chicago Press, 2009.

Trevor J. M. Bench-Capon. Persuasion in practical argument using value-based argumentation frameworks. *Journal of Logic and Computation*, 13(3):429–448, 2003.

Claudia Bianchi. Semantics and pragmatics: The distinction reloaded. In Claudia Bianchi, editor, *The Semantics/Pragmatics Distinction*, pages 1–12. CSLI Publications, 2004.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

Hendrik Blockeel and Luc De Raedt. Top-down induction of first-order logical decision trees. *Artificial Intelligence*, 101(1-2):285–297, 1998.

Hendrik Blockeel, Luc De Raedt, and Jan Ramon. Top-down induction of clustering trees. In J. Shavlik, editor, *Proceedings of the 15th International Conference on Machine Learning*, pages 55–63, 1998.

Hendrik Blockeel, Luc Dehaspe, Bart Demoen, Gerda Janssens, Jan Ramon, and Henk Vandecasteele. Improving the efficiency of inductive logic programming through the use of query packs. *Journal of Artificial Intelligence Research*, 16:135–166, 2002.

Hendrik Blockeel, Sašo Džeroski, Boris Kompare, Stefan Kramer, Bernard Pfahringer, and Wim van Laer. Experiments in predicting biodegradability. *Applied Artificial Intelligence*, 18(2):157–181, 2004.

Rafael Bordini, Jomi Hübner, and Michael Wooldridge. *Programming Multi-Agent Systems in AgentSpeak using Jason*. Wiley, 2007.

Dennis Basil Bromley. *Reputation, Image and Impression Management*. John Wiley & Sons, 1993.

Chris Burnett, Timothy J. Norman, and Katia Sycara. Trust decision-making in multi-agent systems. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI'11)*, pages 115–120, Barcelona, Spain, 2011. AAAI Press.

Ana Casali. *On Intentional and Social Agents with Graded Attitudes*, volume 42 of *Monografies de l'Institut d'Investigació en Intel·ligencia Artificial*. Consell Superior d'Investigacions Científiques, 2008.

Sara Casare and Jaime Sichman. Towards a functional ontology of reputation. In *Proceedings of the 4th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005)*, pages 505–511, Utrecht, The Netherlands, 2005. ACM.

Christiano Castelfranchi and Rino Falcone. *Trust Theory: A Socio-cognitive and Computational Model*. Wiley, 2010.

M. Celentani, D. Fudenberg, D. K. Levine, and W. Psendorfer. Maintaining a reputation against a long-lived opponent. *Econometrica*, 64(3):691–704, 1996.

Daniel Chandler. *Semiotics: The Basics*. Routledge, 2002.

Carlos Chesñevar and Guillermo Simari. Modelling inference in argumentation through labelled deduction: Formalization and logical properties. *Logica Universalis*, 1(1):93–124, 2007.

Rosaria Conte and Mario Paolucci. *Reputation in Artificial Societies: Social beliefs for social order*. Kluwer Academic Publishers, 2002.

Gregory W. Corder and Dale I. Foreman. *Nonparametric Statistics for Non-Statisticions: A Step-by-Step Approach*. Wiley, 2009.

Natalia Criado, Estefania Argente, and Vicent Botti. Normative deliberation in graded BDI agents. In Jürgen Dix and Cees Witteveen, editors, *Proceedings of MATES'10*, volume 6251 of *LNAI*, pages 52–63. Springer, 2010.

Partha Dasgupta. Trust as a commodity. In D. Gambetta, editor, *Trust: Making and Breaking Cooperative Relations*, pages 49–72. Blackwell, 1988/2000.

Mehdi Dastani. 2APL: A Practical Agent Programming Language. *Journal on Autonomous Agents and Multi-Agent Systems*, 16:214–248, 2008. ISSN 1387-2532.

Mehdi Dastani, Andreas Herzig, Joris Hulstijn, and Leendert van der Torre. Inferring trust. In João Alexandre Leite and Paolo Torroni, editors, *Proceedings of Fifth Workshop on Computational Logic in Multi-agent Systems (CLIMA'04)*, volume 3487 of *LNCS*, pages 144–160, 2004.

Luc De Raedt. *Logical and Relational Learning*. Springer, 2008.

Luc De Raedt, Peter Idestam-Almquist, and Gunther Sablon. θ-subsumption for structural matching. In *Proceedings of the 9th European Conference on Machine Learning (ECML'97)*, volume 1224 of *LNCS*, pages 73–84. Springer, 1997.

Chrysanthos Dellarocas. Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior. In *Proceedings of the 2nd ACM conference on Electronic commerce (EC'00)*, pages 150–157, New York, USA, 2000.

Daniel C. Dennett. *The Intentional Stance*. MIT Press, 1987.

Robin I. M. Dunbar. Coevolution of neocortical size, group size and language in humans. *Behavioral and Brain Sciences*, 16:681–735, 1993.

Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 7(2):321–358, 1995.

J. Michael Dunn and Gary M. Hardegree. *Algebraic Methods in Philosophical Logic*. Oxford University Press, 2001.

Jérôme Euzenat and Pavel Shvaiko. *Ontology matching*. Springer, 2007.

Rino Falcone, Giovanni Pezzulo, and Cristiano Castelfranchi. *A Fuzzy Approach to a Belief-based Trust Computation*, volume 3577 of *LNAI*, pages 43–58. Springer, 2005.

Ernst Fehr and Urs Fischbacher. The nature of human altruism. *Nature*, 425: 785–791, 2003.

Jose-Luis Fernandez-Marquez. *Bio-inspired Mechanisms for Self-organising Systems*. PhD thesis, Universitat Autònoma de Barcelona, 2011.

Nir Friedman, Dan Geiger, and Moiser Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2–3):131–163, 1997.

Karen K. Fullam and K. Suzanne Barber. Dynamically learning sources of trust information: experience vs. reputation. In *Proceedings of the 6th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2007)*, pages 1–8, Honolulu, Hawaii, USA, 2007. ACM.

Karen K. Fullam, Tomas B. Klos, Guillaume Muller, Jordi Sabater-Mir, K. Suzanne Barber, and Laurent Vercouter. The Agent Reputation and Trust (ART) testbed. In *Proc. of the 4th International Conference on Trust Management*, volume 3986 of *Lecture Notes in Computer Science*, pages 439–442. Springer, 2006.

Diego Gambetta. Can we trust trust. In Diego Gambetta, editor, *Trust: Making and Breaking Cooperative Relations*, pages 213–237. Basil Blackwell, 1988.

Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis : Mathematical Foundations*. Springer, 1999.

Michael R. Garey and David S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.

Fausto Giunchiglia and Luciano Serafini. Multilanguage hierarchical logics, or: how we can do without modal logics. *Artificial Intelligence*, 65:29–70, 1994.

Jennifer Golbeck. Combining provenance with trust in social networks for semantic web content filtering. In Luc Moreau and Ian Foster, editors, *Provenance and Annotation of Data (IPAW 2006)*, volume 4145 of *LNCS*, pages 101–108. Springer, 2006.

Luigi Guiso, Paolo Sapienza, and Luigi Zingales. Trusting the stock market. *Journal of Finance*, 63(6):2557–2600, 2008.

David Harel. *First-Order Dynamic Logic*, volume 68 of *Lecture Notes in Computer Science*. Springer, 1979.

Ramon Hermoso, Holger Billhardt, and Sascha Ossowski. Dynamic evolution of role taxonomies through multidimensional clustering in multiagent organizations. In Jung-Jin Yang, Makoto Yokoo, Takayuki Ito, Zhi Jin, and Paul Scerri, editors, *Principles of Practice in Multi-Agent Systems (Proceedings of PRIMA'09)*, volume 5925 of *LNCS*, pages 587–594. Springer, Nagoya, Japan, 2009.

Kevin Hoffman, David Zage, and Cristina Nita-Rotaru. A survey of attack and defense techniques for reputation systems. *ACM Computing Surveys (CSUR)*, 42(1):1–31, 2009.

Jomi F. Hübner, Emiliano Lorini, Andreas Herzig, and Laurent Vercouter. From cognitive trust theories to computational trust. In *Proc. of the Twelfth Workshop "Trust in Agent Societies" at AAMAS '09*, pages 55–67, Budapest, Hungary, 2009.

David Hume. A treatise of human nature. In David Fate Norton and Mary J. Norton, editors, *A Treatise of Human Nature (Oxford Philosophical Texts)*. Oxford University Press (2000), 1737.

Trung Dong Huynh, Nicholas R. Jennings, and Nigel R. Shadbolt. An integrated trust and reputation model for open multi-agent systems. *Journal of Autonomous Agents and Multi-Agent Systems*, 13:119–154, 2006.

Norman L. Johnson and Samuel Kotz. Non-smooth sailing or triangular distributions revisited after 50 years. *The Statistician*, 48(2):179–187, 1999.

Neil D. Jones. *Computability and Complexity: From a Programming Perspective*. Foundations of Computing. MIT Press, 1997.

Audung Jøsang and Roslan Ismail. The beta reputation system. In *Proceedings of the Fifteenth Bled Electronic Commerce Conference e-Reality: Constructing the e-Economy*, Bled, Slovenia, 2002.

Audung Jøsang, Roslan Ismail, and Colin Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618–644, 2007.

Sindhu Joseph, Carles Sierra, Marco Schorlemmer, and Pilar Dellunde. Deductive coherence and norm adoption. *Logic Journal of the IGPL*, 18(1):118–156, 2010.

Radu Jurca and Boi Faltings. Obtaining reliable feedback for sanctioning reputation mechanisms. *Journal of Artificial Intelligence Research*, 29(1):391–419, August 2007.

Yannis Kalfoglou and Marco Schorlemmer. IF-Map: An ontology-mapping method based on information-flow theory. In Stefano Spaccapietra, Sal March, and Karl Aberer, editors, *Journal on Data Semantics I*, volume 2800 of *Lecture Notes in Computer Science*, pages 98–127. Springer, 2003.

Aram Karalič and Ivan Bratko. First order regression. *Machine Learning*, 26: 147–176, 1997.

Tobias J. Klein, Christian Lambertz, Giancarlo Spagnolo, and Konrad O. Stahl. The actual structure of eBay's feedback mechanism and early evidence on the effects of recent changes. *International Journal of Electronic Business*, 7(3): 301–320, 2009.

Andrew Koster, Jordi Sabater-Mir, and Marco Schorlemmer. Inductively generated trust alignments based on shared interactions (extended abstract). In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, pages 1571–1572, Toronto, Canada, 2010a. IFAAMAS.

Andrew Koster, Jordi Sabater-Mir, and Marco Schorlemmer. Engineering trust alignment: a first approach. In *Proc. of the Thirteenth Workshop "Trust in Agent Societies" at AAMAS '10*, pages 111–122, Toronto, Canada, 2010b. IFAAMAS.

Stefan Kramer and Gerhard Widmer. Inducing classification and regression trees in first order logic. In Sašo Džeroski and Nada Lavrač, editors, *Relational Data Mining*, pages 140–156. Springer, 2001.

Yann Krupa, Laurent Vercouter, Jomi Fred Hubner, and Andreas Herzig. Trust based evaluation of wikipedia's contributors. In *Engineering Societies in the Agents World X*, volume 5881 of *Lecture Notes in Computer Science*, pages 148–161. Springer, 2009.

Ralph Levien and Alexander Aiken. Attack-resistant trust metrics for public key certification. In *Proceedings of the 7th USENIX Security Symposium*, San Antonio, Texas, USA, 1998.

Churn-Jung Liau. Belief, information acquisition, and trust in multi-agent systems – a modal logic formulation. *Artificial Intelligence*, 149(1):31–60, 2003.

John Locke. Two treatises of government. In Peter Laslett, editor, *Locke's Two Treatises of Government*. Cambridge University Press (1960), 1689.

Emiliano Lorini and Robert Demolombe. From binary trust to graded trust in information sources: a logical perspective. In Rino Falcone, Suzanne K. Barber, Jordi Sabater-Mir, and Munindar P. Singh, editors, *Trust in Agent Societies – 11th International Workshop, TRUST 2008*, volume 5396 of *LNAI*, pages 205–225. Springer, Estoril, Portugal, 2008.

M. Luck, P. McBurney, O. Shehory, and S. Willmott. *Agent Technology: Computing as Interaction (A Roadmap for Agent Based Computing)*. AgentLink, 2005.

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

Paul-Amary Matt, Maxime Morge, and Francesca Toni. Combining statistics and arguments to compute trust. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, pages 209–216, Toronto, Canada, 2010. IFAAMAS.

Charles Morris. Foundations of the theory of signs. In Rudolf Carnap Otto Neurath and Charles Morris, editors, *Foundations of the Unity of Science*, volume 1, chapter 2, pages 77–137. University of Chicago Press, 1938.

Luis G. Nardin, Anarosa A. F. Brandão, Guillaume Muller, and Jaime S. Sichman. SOARI: A service-oriented architecture to support agent reputation models interoperability. In Rino Falcone, Suzanne K. Barber, Jordi Sabater-Mir, and Munindar P. Singh, editors, *Trust in Agent Societies – 11th International Workshop, TRUST 2008*, volume 5396 of *LNAI*, pages 292–307, Estoril, Portugal, 2008. Springer.

Luis G. Nardin, Anarosa A. F. Brandão, Guillaume Muller, and Jaime S. Sichman. Effects of expressiveness and heterogeneity of reputation models in the art-testbed: Some preliminar experiments using the soari architecture. In *Proc. of the Twelfth Workshop "Trust in Agent Societies" at AAMAS '09*, Budapest, Hungary, 2009.

Shan-Hwei Nienhuys-Cheng and Ronald de Wolf. *Foundations of Inductive Logic Programming*, volume 1228 of *LNAI*. Springer, 1997.

James J. Odell, Parunak Van Dyke, and Mitchell Fleischer. The role of roles in designing effective agent organizations. In Alessandro Garcia, Carlos Lucena, Franco Zambonelli, Andrea Omicini, and Jaelson Castro, editors, *Software Engineering for Large-Scale Multi-Agent Systems*, volume 2603 of *Lecture Notes in Computer Science*, pages 27–38. Springer, 2003.

Nardine Osman, Carles Sierra, and Jordi Sabater-Mir. Propagation of opinions in structural graphs. In *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI 2010)*, Lisbon, Portugal, 2010.

Simon Parsons, Carles Sierra, and Nick R. Jennings. Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8(3):261–292, 1998.

Simon Parsons, Michael Wooldridge, and Leila Amgoud. Properties and complexity of some formal inter-agent dialogues. *Journal of Logic and Computation*, 13(3):347–376, 2003.

Simon Parsons, Yuqing Tang, Elizabeth Sklar, Peter McBurney, and Kai Cai. Argumentation-basded reasoning in agents with varying degrees of trust. In Kagan Tumer, Pinar Yolum, Liz Sonenberg, and Peter Stone, editors, *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, pages 879–886, Taipei, Taiwan, 2011. IFAAMAS.

Adrián Perreau de Pinninck Bas, Marco Schorlemmer, Carles Sierra, and Stephen Cranefield. A social-network defence against whitewashing. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, pages 1563–1564, Toronto, Canada, 2010. IFAAMAS.

Isaac Pinyol. *Milking the Reputation Cow: Argumentation, Reasoning and Cognitive Agents*, volume 44 of *Monografies de l'Institut d'Investigació en Intel·ligencia Artificial*. Consell Superior d'Investigacions Científiques, 2011.

Isaac Pinyol and Jordi Sabater-Mir. Arguing about reputation. The LRep language. In A. Artikis, G.M.P. O'Hare, K. Stathis, and G. Vouros, editors, *Engineering Societies in the Agents World VIII: 8th International Workshop, ESAW 2007*, volume 4995 of *LNAI*, pages 284–299. Springer, 2007.

Isaac Pinyol and Jordi Sabater-Mir. Pragmatic-strategic reputation-based decisions in BDI agents. In *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, pages 1001–1008, Budapest, Hungary, 2009a.

Isaac Pinyol and Jordi Sabater-Mir. Towards the definition of an argumentation framework using reputation information. In *Proc. of the Twelfth Workshop "Trust in Agent Societies" at AAMAS '09*, pages 92–103, 2009b.

Isaac Pinyol and Jordi Sabater-Mir. An argumentation-based protocol for social evaluations exchange. In *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI 2010)*, pages 997–998, Lisbon, Portugal, 2010. IOS Press.

Isaac Pinyol and Jordi Sabater-Mir. Computational trust and reputation models for open multi-agent systems: a review. *Artificial Intelligence Review*, In Press. doi:10.1007/s10462-011-9277-z.

Isaac Pinyol, Jordi Sabater-Mir, and Guifre Cuni. How to talk about reputation using a common ontology: From definition to implementation. In *Proc. of the Tenth Workshop "Trust in Agent Societies" at AAMAS '07*, pages 90–102, Honolulu, Hawaii, USA, 2007.

Isaac Pinyol, Jordi Sabater-Mir, Pilar Dellunde, and Mario Paolucci. Reputation-based decisions for logic-based cognitive agents. *Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS)*, 24(1):175–216, 2012.

Plato. The republic. In *The Republic: The Complete and Unabridged Jowett Translation*. Vintage Books (1991), 370BC. Translated by Benjamin Jowett in 1871.

Gordon D. Plotkin. A note on inductive generalization. *Machine Intelligence*, 5, 1970.

Henri Prade. A qualitative bipolar argumentative view of trust. In V.S. Subrahmnanian and Henri Prade, editors, *International Conference on Scalable Uncertainty Management (SUM 2007)*, volume 4772 of *LNAI*, pages 268–276. Springer, 2007.

Henry Prakken. Coherence and flexibility in dialogue games for argumentation. *Journal of Logic and Computation*, 15(6):1009–1040, 2005.

Henry Prakken. Models of persuasion dialogue. In Iyad Rahwan and Guillermo Simari, editors, *Argumentation in Artificial Intelligence*, chapter 14, pages 281–300. Springer, 2009.

Iyad Rahwan and Guillermo Simari. *Argumentation in Artificial Intelligence*. Springer, 2009.

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

Anand S. Rao and Michael P. Georgeff. Modeling rational agents within a BDI-architecture. In R. Fikes and E. Sandewall, editors, *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning*, pages 473–484, San Mateo, CA, USA, 1991. Morgan Kaufmann.

Anand S. Rao and Michael P. Georgeff. BDI agents: From theory to practice. In *Proceedings of the First International Congress on Multi-Agent Systems (ICMAS'95)*, pages 312–319, San Francisco, USA, 1995. AAAI.

Kevin Regan, Pascal Poupart, and Robin Cohen. Bayesian reputation modeling in e-marketplaces sensitive to subjectivity, deception and change. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)*, pages 1206–1212, Boston, MA, USA, 2006. AAAI Press.

Martin Rehák and Pěchouček. Trust modeling with context representation and generalized identities. In Mathias Klusch, Koen Hindriks, Mike P. Papazoglou, and Leon Sterling, editors, *Cooperative Information Agents XI – Proceedings of the 11th International Worshop, CIA 2007*, volume 4676 of *LNAI*, pages 298–312. Springer, Delft, The Netherlands, 2007.

Philip Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, 11:95–130, 1999.

Francesco Ricci, Rokach Lior, Bracha Shapira, and Paul B. Kantor. *Recommender Systems Handbook*. Springer, 2010.

M. Andrea Rodríguez and Max J. Egenhofer. Determining semantic similarity among entity classes from different ontologies. *IEEE Transactions of Knowledge and Data Engineering*, 15(2):442–456, 2003.

Stuart Russel and Peter Norvig. *Artificial Intelligence: A Modern Approach, 3rd edition*. Prentice Hall, 2010.

David J. Saab. An emergent culture model for discerning tag semantics in folksonomies. In *Proceedings of the 2011 iConference*, ICPS, pages 552–560, Seattle, USA, 2011. ACM.

Jordi Sabater. *Trust and Reputation for Agent Societies*, volume 20 of *Monografies de l'Institut d'Investigació en Intel·ligencia Artificial*. Consell Superior d'Investigacions Científiques, 2003.

Jordi Sabater and Carles Sierra. REGRET: A reputation model for gregarious societies. In *Proceedings of the Fourth Workshop on Deception, Fraud and Trust in Agent Societies*, pages 61–69, 2001.

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

Jordi Sabater and Carles Sierra. Review on computational trust and reputation models. *Artificial Intelligence Review*, 24(1):33–60, 2005.

Jordi Sabater, Carles Sierra, Simon Parsons, and Nicholas R. Jennings. Engineering executable agents using multi-context systems. *Journal of Logic and Computation*, 12(3):413–442, 2002.

Jordi Sabater, Mario Paolucci, and Rosaria Conte. Repage: REPutation and imAGE among limited autonomous partners. *JASSS - Journal of Artificial Societies and Social Simulation*, 9(2):3, 2006. URL http://jasss.soc.surrey.ac.uk/9/2/3.html.

Jordi Sabater-Mir and Mario Paolucci. On representation and aggregation of social evaluations in computational trust and reputation models. *International Journal of Approximate Reasoning*, 46(3):458–483, 2007.

Michael Schillo, Petra Funk, and Michael Rovatsos. Who can you trust: Dealing with deception. In *Proceedings of the 2nd Workshop on "Deception, Fraud and Trust in Agent Societies" at Autonomous Agents '99*, pages 81–94, Seattle, USA, 1999.

Michael Schläfli. Using context-dependant trust in team formation. In *Proceedings of EUMAS'11*, Maastricht, The Netherlands, 2011.

Marco Schorlemmer. Term rewriting in a logic of special relations. In *AMAST'98*, volume 1546 of *Lecture Notes in Computer Science*, pages 178–198, 1998.

Marco Schorlemmer and Yannis Kalfoglou. Institutionalising ontology-based semantic integration. *Applied Ontology*, 3(3):131–150, 2008.

Marco Schorlemmer, Yannis Kalfoglou, and Manuel Atencia. A formal foundation for ontology-alignment interaction models. *International Journal on Semantic Web and Information Systems*, 3(2):50–68, 2007.

Murat Şensoy and Pinar Yolum. Ontology-based service representation and selection. *IEEE Transactions on Knowledge and Data Engineering*, 19(8): 1102–1115, 2007.

Murat Şensoy, Jie Zhang, Pinar Yolum, and Robin Cohen. POYRAZ: Context-aware service selection under deception. *Computational Intelligence*, 25(4): 335–366, 2009.

Emilio Serrano, Michael Rovatsos, and Juan Botia. A qualitative reputation system for multiagent systems with protocol-based communication. In *Proceedings of the Eleventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS'12)*, Valencia, Spain, Forthcoming 2012. IFAAMAS.

Claude Shannon. A mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–423 and 623–656, 1948.

Carles Sierra and John Debenham. An information-based model for trust. In *Proceedings of the 4th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005)*, pages 497–504, Utrecht, The Netherlands, 2005. ACM.

Ronald D. Snee. Validation of regression models: Methods and examples. *Technometrics*, 19(4):415–428, 1977.

Eugen Staab and Thomas Engel. Combining cognitive with computational trust reasoning. In Rino Falcone, Karen Barber, Jordi Sabater-Mir, and Munindar P. Singh, editors, *Trust in Agent Societies – 11th International Workshop, TRUST 2008*, volume 5396 of *LNAI*, pages 99–111. Springer, Estoril, Portugal, 2008.

Yuqing Tang, Kai Cai, Peter McBurney, and Simon Parsons. A system of argumentation for reasoning about trust. In *Proceedings of EUMAS'10*, Paris, France, 2010.

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

Werner Uwents and Hendrik Blockeel. A comparison between neural network methods for learning aggregate functions. In *Proceedings of the 11th International Conference on Discovery Science*, volume 5255 of *LNCS*, pages 88–99, 2008.

Thomas Vander Wal. Folksonomy: Coinage and definition. retrieved February 2, 2007. URL http://vanderwal.net/folksonomy.html.

Laurent Vercouter and Guillaume Muller. L.I.A.R.: Achieving social control in open and decentralised multi-agent systems. *Applied Artificial Intelligence*, 24 (8):723–768, 2010.

Serena Villata, Guido Boella, Dov Gabbay, and Leendert van der Torre. Arguing about the trustworthiness of information sources. In Weiru Liu, editor, *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, volume 6717 of *LNCS*, pages 74–85. Springer, 2011.

George Vogiatzis, Ian MacGillivray, and Maria Chli. A probabilistic model for trust and reputation. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, pages 225–232, Toronto, Canada, 2010. IFAAMAS.

Quang Hieu Vu, Mihai Lupu, and Beng Chin Ooi. *Peer-to-Peer Computing: Principles and Applications*. Springer, 2010.

Douglas N. Walton and Erik C. W. Krabbe. *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning*. State of University of New York Press, Albany, NY, USA, 1995.

Xianchang Wang, Jia-huai You, and Li Yan Yuan. Nonmonotonic reasoning by monotonic inferences with priority constraints. In *Nonmonotonic Extensions of Logic Programming*, volume 1216 of *LNAI*, pages 91–109, 1997.

Jens Witkowski. Truthful feedback for sanctioning reputation mechanisms. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence (UAI'10)*, pages 658–665, Corvallis, Oregon, 2010. AUAI Press.

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

Bin Yu and Munindar P. Singh. An evidential model of distributed reputation management. In *Proceedings of the 1st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2002)*, pages 294–301, Bologna, Italy, 2002. ACM.

Sanming Zhou, Zhi-Qiang Liu, and Jian Ying Zhang. Fuzzy causal networks: General model, inference, and convergence fuzzy causal networks: General model, inference and convergence. *IEEE Transactions on Fuzzy Systems*, 14 (3):412–420, June 2006.