In Journal of Logic and Computation, Special Issue on Agreement Technologies, 2011.

Using argumentation to reason about trust and belief*

Yuqing Tang[1], Kai Cai[1], Peter McBurney[2], Elizabeth Sklar[1,3] and Simon Parsons[1,2,3]
[1]Department of Computer Science, Graduate Center, City University of New York, 365 Fifth Avenue, New York, NY 10016, USA, {ytang,kcai}@gc.cuny.edu
[2]Department of Informatics, Kings College London, The Strand, London, WC2R 2LS, United Kingdom, mcburney@kcl.ac.uk
[3]Department of Computer & Information Science, Brooklyn College, City University of New York, 2900 Bedford Avenue, Brooklyn, NY 11210 USA
{sklar,parsons}@sci.brooklyn.cuny.edu

April 29, 2011

Abstract

Trust is a mechanism for managing the uncertainty about autonomous entities and the information they store, and so can play an important role in any decentralized system. As a result, trust has been widely studied in multiagent systems and related fields such as the semantic web. Here we introduce a formal system of argumentation that can be used to reason using information about trust. This system is described as a set of graphs, which makes it possible to combine our approach with conventional representations of trust between individuals where the relationships between individuals are given in the form of a graph. The resulting system can easily relate the grounds of an argument to the agent that supplied the information, and can be used as the basis to compute Dungian notions of acceptability that take trust into account. We explore some of the properties of these argumentation graphs, examine the computation of trust and belief in the graphs, and illustrate the capabilities of the system on an example from the trust literature.

1 Introduction

Trust is a mechanism for managing the uncertainty about autonomous entities and the information they deal with. As a result, trust can play an important role in any decentralized system. As computer systems have become increasingly distributed, and control in those systems has become more decentralized, trust has been an increasingly important concept in computer science [6, 35].

As a number of authors have pointed out, trust is a concept that is both complex and rather difficult to pin down precisely and as a result, there are a number of different definitions in the literature. Thus, to pick a few specific examples, Sztompka [85] suggests that:

> Trust is a bet about the future contingent actions of others.

while Mcknight and Chervany [61], drawing on a range of existing definitions, define:

*This paper is a revised and expanded version of [86].

> Trust is the extent to which one party is willing to depend on something or somebody in a given situation with a feeling of relative security, even though negative consequences are possible.

and Gambetta [28] states:

> Trust is the subjective probability by which an individual, A, expects that another individual, B, performs a given action on which its welfare depends.

In the multiagent systems community, an influential model of trust is that of Falcone and Castelfranchi [25] who argue that trust arises when the trusting agent only trusts another with respect to some goal, and thus with respect to some action that the second agent can perform which will bring about that goal. Trust then relates to the beliefs that the trusting agent holds, which include a belief that the second, trusted agent is capable of bringing about the goal, will carry out the action to bring about the goal, and that this action is necessary to bring about the goal.

It is also pointed out in a number of places that there are different kinds of trust, what [44] calls "trust scopes". For example, [35] identifies the following types of trust:

1. *Provision trust:* the trust that exists between the user of a service or resource, and the provider of that resource.
2. *Access trust:* the trust that exists between the owner of a resource and those that are accessing those resources.
3. *Delegation trust:* the trust that exists between an individual who delegates responsibility for some action or decision and the individual to whom that action or decision is delegated.
4. *Identity trust:* trust that an individual is who they claim to be.
5. *Context trust:* trust that an individual has in the existence of sufficient infrastructure to support whatever activities that individual is engaged in.

While these definitions of trust vary, there are clearly some common elements. First, there is a degree of uncertainty associated with trust. As [11] point out, trust is a function of the subjective certainty of the trusting agent's beliefs. This subjective certainty might, as argued by Gambetta, be expressed as a subjective probability, or as Sztompka prefers to think of it as a bet[1]. The subjective certainty can be expressed as a "feeling of security" [61], or as Castelfranchi and Falcone [11] prefer, in terms of reasons based on perceived risk. Second, trust is tied up with the relationships between individuals. Trust is related to the actions of individuals and how those actions affect others.

As Demolombe and Lorini [15, 57] have shown, building a logical model of even part of a detailed trust model is a complex undertaking (they construct a modal logic model of one part of Falcone and Castelfranchi's [25] concept of trust). Our focus is rather different. The aim of this paper is to begin to combine two lines of work — work on modelling trust, and work on reasoning using argumentation. While, as we discuss in the next section, there is a large amount of work on trust, we are most concerned with work on handling trust in social networks which deals with a simpler concept of trust than those we have discussed so far. Much of the work on trust in social networks, for example [37], has concentrated on identifying ways to take a social network in which people rate the trustworthiness of their acquaintances, and infer new ratings between pairs of individuals. Many models have been proposed, and their predictions compared with the ratings that people themselves generate. However, little work has been done on reasoning using these ratings.

Argumentation [73] on the other hand, is a field that is almost entirely concerned with reasoning and has a couple of features that we believe make it appropriate for dealing with trust. First, argumentation has been used to capture features of interaction between agents [67, 68], a situation in which trust needs to be considered, so it seems to be a natural step to think about how argumentation-based interactions might be extended with a model of trust. The first step in doing this is to develop a system of argumentation that can make use of trust information and that is what we do here. Second, argumentation provides a reason-based model of trust — it constructs arguments (reasons) for and against adopting beliefs and actions which explicitly record the agents that need to be trusted in the adoption. Thus our work contributes towards a model of trust that is reason-based as [11] argues trust should be. (The system we describe here is a system that constructs arguments that incorporate information about trust, not arguments about what

[1] Bets can, of course, be expressed in terms of subjective probabilities [41].

information or agents to trust.) As we have suggested before [69], the fact that argumentation records the steps used in reaching conclusions makes it appropriate for reasoning where the provenance of information is important, as is widely acknowledged in handling trust [30, 32].

In this paper, we develop a general system of argumentation that can represent trust information, and be used in combination with a trust network that captures relationships between agents. This system makes it possible for an agent to construct arguments where belief in the conclusions is related to the degree of trust in the agents who supplied the information that is used as premises in the arguments. We present several instantiations of the general system, and demonstrate that one of them gives intuitively appealing results on an illustrative example. Part of the novelty of the work is its fusion of trust and argumentation, a topic that has not received much attention (though we describe a couple of related papers in Section 9). Another part of the novelty comes from the fact that the model we develop is graphical. Though arguments, and the relationships between them, are often drawn as graphs, the underlying systems are not typically formulated graph-theoretically. Since one of our long-term aims is to use the graphical display of arguments as a way to help people make decisions in situations where trust — especially trust that stems from the provenance of information — is important, a graphical formulation seems natural. As part of this, we give a graphical formulation of the usual notions of acceptability [22], though with some slight variations from the notions [22].

The rest of the paper is structured as follows. Section 3 introduces models of trust and argumentation, and then Section 4 combines them. The combined model is an abstract model, and to make the discussion more concrete, Section 5 give some instantiations. Section 6 gives a large example, before Sections 7 and 8 consider the computational aspects of building the graphical models and propagating trust and belief values through them. Finally Section 9 reviews the contribution of the paper, particularly with respect to other work on trust and argumentation, and Section 10 concludes.

2 Related work

As we pointed out above, there has been much work in recent years that is concerned with trust, both within computer science and outside computer science. Much of the work on trust in computer science has concentrated on dealing with specific scenarios in which trust has to be established or handled in some fashion. Thus, for example, we see work on trust in peer-to-peer networks, including the EigenTrust algorithm [48] — a variant of PageRank [65] where downloads from a source play the same role as outgoing hyperlinks and which is effective in excluding peers who want to disrupt the network. [1] then builds on this, developing a mechanism that prevents peers manipulating their trust values to get preferential downloads. [96] is concerned with slightly different issues in mobile ad-hoc networks, looking to prevent nodes from getting others to transmit their messages while refusing to transmit the messages of others.

The internet, as the largest distributed system of all, is naturally a target of much of the research on trust. There have, for example, been studies on the development of trust in ecommerce through the use of reputation systems [78] and studies on how such systems perform [77, 87] and how such systems can be manipulated [54]. One interesting recent development is the idea of having individuals indemnify each other by placing some form of financial guarantee on transactions that others enter into [13, 14], thus providing a reputation mechanism that is strategy-proof. Another area of concern has to do with the reliability of sources of information on the web. [92], for example, investigates mechanisms to determine which sources to trust when faced with multiple conflicting sources, while [17] looks at the related question of how to resolve conflicting information, and [2] extends this idea to rate the individuals who provide information by looking at the history of the information they have provided. Issues related to trust in the social web have also attracted much attention [34, 62, 87, 91].

Trust is an especially important issue from the perspective of autonomous agents and multiagent systems. The premise behind the multiagent systems field is that of developing software agents that will work in the interests of their owners, carrying out their owners' wishes while interacting with other entities. In such interactions, agents will have to reason about the amount that they should trust those other entities, whether they are trusting those entities to carry out some task, or whether they are trusting those entities to not misuse crucial information. As a result we find much work on trust in agent-based systems [81].

In the previous section, we mentioned the definition of trust suggested by Falcone and Castelfranchi [25], which relates trust explicitly to the goals of an agent, and considers trust to be concerned with whether another agent can and

will perform an action which will enable the first agent to achieve its goals. Much has been written on this model, and, from our perspective, [11] and [27] are the most important. [11] argues that while trust has an element of "subjective certainty" to it, it is also reason-based, and this is a position that we completely agree with — as we will see below our handling of trust uses some numerical elements to capture aspects of that certainty or uncertainty while relying on argumentation to capture the reasons[2]. [27] discusses the issue of transitivity, which is often taken for granted in work on trust, but which always involves assumptions, assumptions which are analyzed in detail in [27].

Give the logical basis of the argumentation system that we develop, we should point out other work on logical approaches to trust. As we mentioned above, [15, 57] have developed a logic that captures important elements of the definition of trust in [25]. This model has also been extended [9] to deal with situations in which the trusting agent has to rely on another agent for more than a single action. The focus on action, which is of course key in [25]'s concept of trust, distinguishes these models from other logical accounts of trust, such as [56] where the focus is on trust in information sources. This latter is much closer to our concern, though our notions of trust and the beliefs that are affected by them are much simpler than either those of [15] and [56].

In work on trust in multiagent systems and the social web, it is common to assume that individuals maintain a *trust network* of their acquaintances, which includes ratings of how much those acquaintances are trusted, and how much those acquaintances trust their acquaintances, and so on. One natural question to ask in this context is what inference is reasonable in such networks, and the propagation of trust — both through the transitivity of trust relations [50, 53, 79, 90] and more complex relationships like "co-citation" [36] — have been studied. Our approach builds on this work by considering how agents can make use of information that comes from acquaintances for which a trust value is derived using this kind of computation. In this regard, the closest progenitor of the work in this paper is [50], which combines information from different individuals in a trust to reason about what one of these agents should do. ([50] is also the paper from which we draw our ongoing example.)

3 An abstract model of trust and argumentation

In this section we present the formal models of trust and argumentation that we use in this paper.

3.1 Some graph theory

Since graphs of various kinds will crop up throughout this paper, we start with a little graph theory, based on the introductory chapters of [16]. A *graph* is a pair $G = \langle V, E \rangle$ of sets. V is the set of *vertices* (or nodes). E is the set of *edges* (or arcs), and is a set of subsets of V. Each element $e \in E$ is the set of nodes joined by that edge. For much of this paper we will be concerned with graphs that include *hyperedges*, that is edges which join three or more vertices. Thus if $V = \{v_1, v_2, v_3, v_4, v_5\}$, E might be $\{\{v_1, v_2\}, \{v_3, v_4\}, \{v_1, v_3, v_5\}\}$. A graph that includes hyperedges is called a *hypergraph*. If $V' \subseteq V$ and $E' \subseteq E$ then $G' = \langle V', E' \rangle$ is a *subgraph* of G, which we write as $G' \subseteq G$.

Many of the graphs we use will be *directed* graphs. A directed graph is a graph in which each edge has an initial vertex (or set of vertices) and a terminal vertex (or set of vertices). Thus for:

$$E = \{\{v_1, v_2\}, \{v_3, v_4\}, \{v_1, v_3, v_5\}\},$$

v_1 might be the initial vertex and v_2 the terminal vertex of the first edge, and $\{v_1, v_3\}$ might be the initial vertices of the third (hyper) edge and v_5 the terminal vertex. (The third edge might also have v_1 as the initial vertex and $\{v_3, v_5\}$ as the terminal vertices.)

A *path* in an undirected graph G is just a subgraph of G for which $V = \{v_0, v_1, v_2, \ldots, v_k\}$ and $E = \{\{v_0, v_1\}, \{v_1, v_2\}, \{v_{k-1}, v_k\}\}$. Where this is unambiguous, we will write the path as $\langle v_0, v_1, v_2, \ldots, v_k \rangle$. Clearly E has to be such that there is an edge joining every subsequent pair of vertices in V. When the graph is directed, we distinguish between an *undirected path*, as defined above, and a *directed path* which obeys an additional constraint on *path*, that for every v_i and v_{i+1} in V, v_i is the initial vertex of the edge that joins v_i and v_{i+1}, and v_{i+1} is the terminal vertex. When we consider paths in hypergraphs, the constraint is that for every pair of vertices in the sequence, there is an edge in E that contains both

[2]The arguments that we consider here are admittedly not arguments for and against trusting another agent, but that aspect can be captured by the related system we describe in [70]

vertices, and if that hypergraph is directed, the first edge in the pair must contain one of the initial vertices, and the second must contain one of the terminal vertices of the edge in question.

A *connected* graph is one for which there is a path between every pair of vertices. If the graph is directed, it is connected if there is an undirected path between every pair of vertices. A *cycle* is an (undirected) path $v_0, \ldots, v_k, v_o$. A graph that doesn't contain any cycles is called a *forest*, and a forest that is connected is called a *tree*. The usual graph-theoretic terminology calls any vertex which is a member of only one edge a *leaf*. In the directed graphs we use, we will place an additional constraint on leaves. A leaf is a vertex that is part of only one edge, and which is the terminal vertex of that edge. In contrast, a *root* is a vertex that is part of only one edge, and which is the initial vertex of that edge.

3.2 Trust Networks

Given this graph theory, we can now describe the first important set of concepts that we will be dealing with. We are interested in a set of Agents Ags and the relationships between them, relationships that we assume are captured in a structure similar to a social network. In particular, we are interested in how the agents trust one another, and to do this we use a simple model of trust that is similar to that used in other work on social networks [33, 50, 55]. Following the usual presentation, for example [36, 50, 53, 79, 90], we start with a *trust relation*:

$$\tau \subseteq Ags \times Ags$$

which we can think of as identifying which agents trust one another. If $\tau(Ag_i, Ag_j)$ for $Ag_i, Ag_j \in Ags$, then Ag_i trusts Ag_j. Note that this is not a symmetric relation, so it is not necessarily the case that $\tau(Ag_i, Ag_j) \Rightarrow \tau(Ag_j, Ag_i)$.

It is natural to represent this trust relation as a directed graph, and we have:

Definition 1 *A* trust network *for a set of agents Ags is a pair*

$$\mathcal{T} = \langle Ags, \{\tau\} \rangle$$

where $\{\tau\}$ is the set of pairwise trust relations over the agents in Ags so that if $\tau(Ag_i, Ag_j)$ is in $\{\tau\}$ then $\{Ag_i, Ag_j\}$ is a directed arc in $\mathcal{T}$.

In this graph, the set of agents is the set of vertices, and the trust relations define the arcs. We are typically interested in *minimal* trust networks, which we define to be trust networks that are connected. These thus capture the relationship between a set of agents all of whom, in one way or another, have something to say about trust in other members of the group. A directed path between agents in the trust network indicates that one agent directly or indirectly trusts others to which it is connected directly or indirectly. For example:

$$\langle Ag_1, Ag_2, \ldots Ag_n \rangle$$

is a path from agent Ag_1 to Ag_n, which requires that:

$$\tau(Ag_1, Ag_2), \tau(Ag_2, Ag_3), \ldots, \tau(Ag_{n-1}, Ag_n)$$

and looking at the path gives us a means to compute the trust that Ag_1 has in Ag_n given the trust that each agent along the path has in the next agent along the path. The usual assumption here is that we can place some measure of opinion on the trust that one agent has in another, so we have:

$$tr : Ags \times Ags \mapsto \mathcal{O}$$

where $\mathcal{O}$ denotes the set of possible values of those opinions. A popular instantiation of opinion in the literature is the set of real numbers $\Re$:

$$tr^{\Re} : Ags \times Ags \mapsto \Re$$

where tr gives a suitable trust value. For example, we might take a value of 0 to indicate that there is no trust between the agents in question and a value of 1 to indicate the fullest possible degree of trust between the agents. Other instantiations of $\mathcal{O}$ might contain more structure, such as the tuple of numbers used in subjective logic [45, 64, 90].

We assume that tr and τ match, so that:

$$\begin{aligned} tr(Ag_i, Ag_j) \neq \bot &\Leftrightarrow (Ag_i, Ag_j) \in \tau \\ tr(Ag_i, Ag_j) = \bot &\Leftrightarrow (Ag_i, Ag_j) \notin \tau \end{aligned}$$

where $\bot$ is the value from $\Re$ that denotes no trust.

Now, this just deals with the direct trust relations encoded in τ. It is usual in work on trust to consider performing inference about trust by assuming that trust relations are transitive. This is easily captured in the notion of a trust network:

Definition 2 *If Ag_i is connected to Ag_j by a set of directed paths $\{\langle Ag_i, Ag_{i+1}, \ldots Ag_j\rangle\}$ in the trust network $\mathcal{T}$, then Ag_i* trusts *Ag_j* according to *$\mathcal{T}$.*

In other words, here we take trust to be a transitive notion. We note that while [27] argue that transitivity does not always hold — and we believe that it is absolutely correct that transitivity only holds under certain conditions — transitivity is an assumption that is commonly made in the literature. For example, the notion of trust embodied here is exactly the "indirect trust" or "derived trust" of [46], and the process of inferring this indirect trust is what [36] calls "direct propagation". Our purpose here is to model those situations in which it is reasonable to consider trust to be transitive, while acknowledging that this is not always the case[3].

If we have a trust function tr, then we can compute:

$$\begin{aligned} tr(Ag_i, Ag_j) &= tr(Ag_i, Ag_{i+1}, Ag_{i+2}, \ldots, Ag_{j-1}, Ag_j) \\ &= tr(Ag_i, Ag_{i+1}) \otimes^{tr} tr(Ag_{i+1}, Ag_{i+2}) \otimes^{tr} \ldots \otimes^{tr} tr(Ag_{j-1}, Ag_j) \end{aligned}$$

for some function

$$\otimes^{tr} : \mathcal{O} \times \mathcal{O} \mapsto \mathcal{O}$$

Here we follow [90] in using the symbol $\otimes^{tr}$ to stand for this operation[4], while allowing it in practice to be one of a number of possible operations as we will discuss below. Sometimes it is the case that there are two or more paths through the trust network between Ag_i and Ag_j indicating that Ag_i has several opinions about the trustworthiness of Ag_j. If we have two paths:

$$\begin{aligned} &\langle Ag_i, Ag'_{i+1}, \ldots Ag'_{j-1}, Ag_j\rangle \\ &\langle Ag_i, Ag''_{i+1}, \ldots Ag''_{j-1}, Ag_j\rangle \end{aligned}$$

and

$$\begin{aligned} tr(Ag_i, Ag_j)' &= tr(Ag_i, Ag'_{i+1}) \otimes^{tr} tr(Ag'_{i+1}, Ag'_{i+2}) \otimes^{tr} \ldots \otimes^{tr} tr(Ag'_{j-1}, Ag_j) \\ tr(Ag_i, Ag_j)'' &= tr(Ag_i, Ag''_{i+1}) \otimes^{tr} tr(Ag''_{i+1}, Ag''_{i+2}) \otimes^{tr} \ldots \otimes^{tr} tr(Ag''_{j-1}, Ag_j) \end{aligned}$$

then the overall degree of trust that Ag_i has in Ag_j is:

$$tr(Ag_i, Ag_j) = tr(Ag_i, Ag_j)' \oplus^{tr} tr(Ag_i, Ag_j)''$$

again using the standard notation

$$\oplus^{tr} : \mathcal{O} \times \mathcal{O} \mapsto \mathcal{O}$$

for a function that combines trust measures along two distinct paths between the same two vertices [90]. This approach can clearly be extended to deal with three or more paths. The literature contains several instantiations of $\otimes^{tr}$ and $\oplus^{tr}$. For example, [79] discusses using multiplication or minimum for $\otimes^{tr}$ and using maximum for $\oplus^{tr}$, while [50] uses a weighted average that in essence adopts multiplication for $\otimes^{tr}$ and addition for $\oplus^{tr}$. [46] and [90] use operators derived

[3] Indeed, in other work [70], we discuss explicitly modelling the ways in which trust is propagated, which makes it possible to perform metareasoning about which forms are applicable in different situations.

[4] [90] uses the unadorned symbol $\otimes$ and we add the subscript to distinguish it from the similar operation for belief values we introduce below.

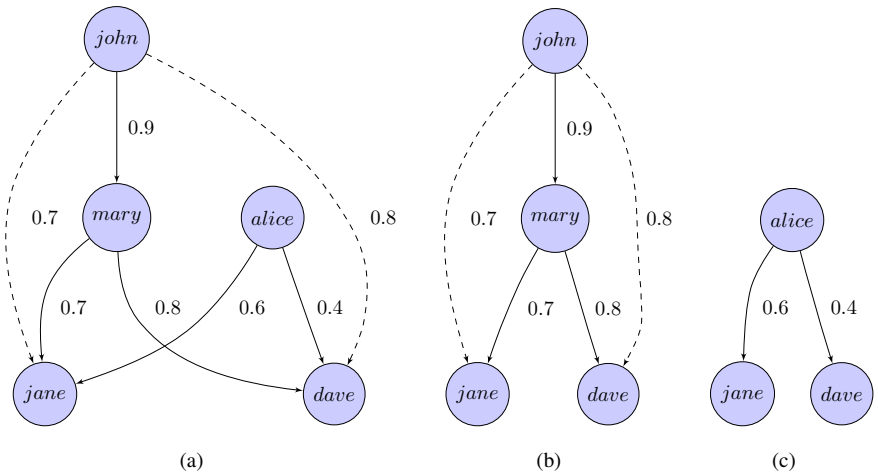

Figure 1: Example trust graphs. (a) shows a complete graph, (b) shows an agent-centric version from John's point of view, and (c) shows an agent-centric view from Alice's point of view.

from Dempster-Shafer theory, and as [8] discuss, we could instantiate operators like these in many other ways. We will discuss some of these options more below.

As an example of a trust graph, consider Figure 1(a) which shows the trust relationship between John, Mary, Alice, Jane and Dave. This is adapted from the example in [50] normalizing the values to lie between 0 and 1. The solid lines are direct trust relationships, the dotted lines are indirect links derived from the direct links. Thus John trusts Jane and Dave because he trusts Mary and Mary trusts Jane and Dave. However, John does not, even indirectly, trust Alice.

Since we will often be considering the viewpoint of a given agent, it is useful for us to define the concept of an *agent-centric* trust graph:

Definition 3 *An* agent-centric *trust graph $\mathcal{T}$ is a trust network with a single root.*

The agent-centric trust graph with Ag at its root is said to be the network from Ag's point of view or the "Ag-centric network". Thus Figure 1(b) is the John-centric version of the graph in Figure 1(a) and Figure 1(c) is the Alice-centric version. We use this terminology because:

Proposition 1 *For an agent-centric trust network $\mathcal{T}$ with Ag at the root, Ag trusts every agent in the network according to $\mathcal{T}$.*

Proof: *An agent-centric trust network has just one root and so it is a tree. Every tree is connected, and in a directed tree such as $\mathcal{T}$, there is a directed path from the single root to every node (if there was not, there would be more than one root). Thus a directed path connects Ag to every other agent in $\mathcal{T}$ and by Definition 2, Ag trusts every agent according to $\mathcal{T}$.* □

Thus an agent-centric trust network identifies the agents that are, direct and indirectly, trusted by the agent corresponding to the root, and every agent in the agent-centric trust network is trusted by the agent at the root. In addition, we can immediately see that:

Proposition 2 *For any trust network $\mathcal{T} = \langle Ags, \{\tau\}\rangle$, there exists a distinct agent-centric trust network $\mathcal{T}'$ for each node in $\mathcal{T}$, and every $\mathcal{T}'$ will be a sub-graph of $\mathcal{T}$.*

Proof: *For the first part, we can construct a trust network $\mathcal{T}' = \langle Ags', \{\tau\}'\rangle$ for every agent $Ag_i \in Ags$ by recursively identifying the agents Ag_j it is linked to by directed arcs from $\mathcal{T}$, and every agent that those Ag_j are linked to, and so on. (If there are no outgoing arcs from an Ag_i, the graph will just include that one agent). Since these graphs all have*

different root nodes, they are distinct. For the second part, it is clear that during this construction process, $Ags' \subseteq Ags$ and $\{\tau\}' \subseteq \{\tau\}$, so $\mathcal{T}' \subseteq \mathcal{T}$. □

Proposition 3 *Given a trust network $\mathcal{T}$ that contains Ag_i and Ag_j and an Ag_i-centric trust network $\mathcal{T}'$ such that $\mathcal{T}' \subseteq \mathcal{T}$, Ag_i trusts Ag_j according to $\mathcal{T}'$ iff Ag_i trusts Ag_j according to $\mathcal{T}$.*

Proof: *From Proposition 2, $\mathcal{T}' \subseteq \mathcal{T}$. As a result, every arc in $\mathcal{T}'$ is also an arc in $\mathcal{T}$, and so every path in $\mathcal{T}'$ is a path in $\mathcal{T}$. Thus Ag_i trusts Ag_j according to $\mathcal{T}'$ only if Ag_i trusts Ag_j according to $\mathcal{T}$. For the "if" part, again consider the recursive construction of $\mathcal{T}'$ sketched in the proof of Proposition 2. If there is a directed path from the root of $\mathcal{T}'$ to some agent in $\mathcal{T}$, then that path will be in $\mathcal{T}'$ and so Ag_i trusts Ag_j according to $\mathcal{T}'$ for every Ag_j that it trusts according to $\mathcal{T}$.* □

Thus an Ag_i-centric trust network that is derived from a trust network $\mathcal{T}$ exactly identifies the agents that, according to $\mathcal{T}$, Ag_i trusts. As a result, when we consider what Ag_i reasons about, we lose nothing by ignoring the parts of $\mathcal{T}$ that aren't in the Ag_i-centric network — they only contain agents that Ag_i can safely ignore because they aren't trusted.

3.3 Argumentation

Now we turn our attention to the structure of the agents. An agent Ag_i has a knowledge base $\Sigma = P \cup \Delta$. P is a set of *premises*, each of which is a logical statement in a language $\mathcal{L}$. Δ is a set of inference rules, which we individually denote by δ, each of which is of the form:

$$\delta = \frac{\{p_1, \ldots p_n\}}{c}$$

where every p_i (denoted by $p_i(\delta)$) and c (denoted by $c(\delta)$) are members of $\mathcal{L}$. In other words the inference rules link some set of premises p_i to a conclusion c. We will also write these rules as $\langle \delta, c \rangle$. In this paper, since here we draw heavily on the work of [50], these rules are much like the normal default rules used in that work. For example, a domain specific inference rule on not watching a comedy film can be represented:

$$\delta = \frac{Comedy(x)}{\neg Watch(x)}$$

wedded to the use of these default-like rule. Δ can be any set of inference rules, for example we might use the natural deduction style rules of [59].

We can represent inference using the rules in δ, whatever form they take, as another graph as follows:

Definition 4 *A* rule network *$\mathcal{R}$ is a directed hypergraph $\langle V^r, E^r \rangle$ where:*

1. *the set of vertices V^r are elements of $\mathcal{L}$;*
2. *the set of edges E^r are inference rules δ;*
3. *the initial vertices of an edge $e \in E^r$ are the premises of the corresponding rule δ; and*
4. *the terminal node of that edge is the corresponding conclusion c.*

Thus a rule network simply connects premises and conclusions of rules. For a simple rule network, see Figure 2 (again this is taken from [50]).

Under certain circumstances, a rule network captures a proof made using the rules and premises in a particular Σ:

Definition 5 *For a given knowledge base $\Sigma = P \cup \Delta$, a rule network $\mathcal{R} = \langle V^r, E^r \rangle$ is a* proof network *if and only if every premise of each $\delta \in E^r$ is either a member of P or the conclusion of some $\delta' \in E^r$.*

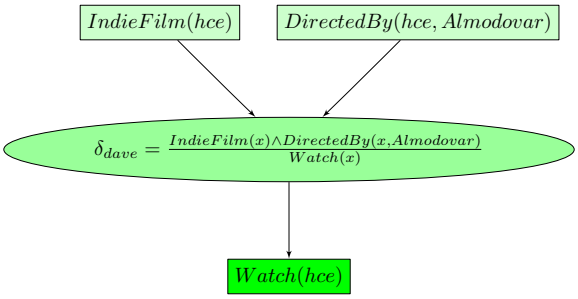

Figure 2: A rule network. The rectangular nodes denote premises and the oval, which represent a hyperedge, denotes an inference rule.

To be a proof network, the rule network has to be constructed from the contents of some knowledge-base — a rule can't be in the proof network unless its premises are either in the knowledge base or are derived by applying rules to premises that are in the knowledge base.

We say that a proof network is *for a conclusion c* if *c* is a leaf of the network. For example, if:

$$\begin{aligned} \Sigma &= \{IndieFilm(hce), DirectedBy(hce, Almodovar)\} \\ &\cup \left\{ \frac{IndieFilm(x) \wedge DirectedBy(x, Almodovar)}{Watch(x)} \right\} \end{aligned}$$

then Figure 2 is a proof network for $Watch(hce)$.

Some proof networks correspond to *arguments*:

Definition 6 *An argument A from a knowledge base $\Sigma = P \cup \Delta$ is a pair $\langle h, H \rangle$ where $H = \langle V^r, E^r \rangle$ is a proof network for h, and h is the only leaf of H.*

H is the *support* of the argument, and h is the *conclusion*. $C(H)$ is the set of *intermediate conclusions* of H, the set of all the conclusions of the $\delta \in E^r$ other than h. $P(H)$ is the set of *pure premises* of H, the premises of the $\delta \in E^r$ that aren't intermediate conclusions of H.

Thus for us an argument is a graphical representation of Prakken's idea of an argument [72], where the argument contains a full trace of the steps that were used to reach the conclusion (rather than just containing the set of rules that were used as in [29]).

Given these definitions, we can easily show that:

Proposition 4 *If $\langle h, H \rangle$ is an argument, then H is a connected graph.*

Proof: *The proof network of an argument is allowed only one leaf. Since it is a proof network, the leaf has to be a conclusion of a rule whose premises are either part of Σ, and thus roots of the graph, or the conclusions of other rules. Working backwards from the leaf/conclusion, it is clear that there can be no components of the graph that aren't connected to the leaf, and so the graph must be connected.* □

Proposition 5 *If $\langle h, H \rangle$ is an argument, then there is only one hyperedge $\langle \delta_i, h \rangle$ in H.*

Proof: *This follows directly from the fact that h is a leaf of H. If there were more than one rule in H with h as its conclusion, h would not be a leaf, which contradicts Definition 6.* □

Thus an argument does not have any duplicated reasoning that supports its conclusion, and an argument is a proof network that is minimal in the sense that it sanctions no inferences other than its conclusion and intermediate steps that become premises of rules required to generate the conclusion.

Since our agents are often going to deal with information that is uncertain, we will assume that each agent Ag_i has a function which assigns a degree of belief — which might be established as described in [58] — to elements of the logical language it uses for premises and rules:

$$bel_i : \mathcal{L} \mapsto \mathcal{B}$$

where $\mathcal{B}$ denotes the set of possible values of the degree of beliefs. A common instantiation of $\mathcal{B}$ is the set of real numbers $\Re$:

$$bel_i : \mathcal{L} \mapsto \Re.$$

Furthermore, as with trust, we will often use a function that returns degrees of belief between 0 and 1. As we build arguments, we need to combine the degrees of belief assigned to premises and rules. For a rule:

$$\delta = \frac{\{p_1, \ldots, p_m\}}{c}$$

each agent Ag_i associates a degree of belief $bel_i(\delta)$ with the rule along with a belief combination function:

$$\delta_i^{\otimes_{bel}}(bel_i(p_1), ..., bel_i(p_m), bel_i(\delta))$$

of the appropriate arity to combine the beliefs in all the premises with the belief in the rule itself to establish the appropriate belief in the conclusion. combination function $\delta_i^{\otimes}$ can be tailored to reflect different belief combinations for different inference schemes (such as the modus ponens, modus tollens, and hypothetical syllogism [52]).

With a belief function bel_i for each element of the grounds of an argument, we can assign belief to an argument as a whole:

$$bel_i(\langle h, H\rangle) = combine_i(\{bel_i(h_k | h_k \in H)\})$$

where $combine_i$ is a function to combine the belief on individual component of H into the belief on the argument as a whole. Ignoring the internal structure of an argument, a simplified implementation of *combine* is

$$combine_i(\{h_1, h_2, ..., h_n\}) = h_1 \otimes^{bel} h_2 \otimes^{bel} ... \otimes^{bel} h_n.$$

where:

$$\otimes^{bel} : \mathcal{B} \times \mathcal{B} \mapsto \mathcal{B}$$

is a function for the pairwise combination of beliefs. Depending on the requirement of the application of our argumentation system, we can also implement the combination function by taking into account the inference structure of the arguments and applying the combination function $\delta_i^{\otimes}$ of every inference rule used in the arguments.

A key notion in argumentation is that arguments *defeat* one another. That is, one argument casts doubt on another by, for example, contradicting the conclusion of the second argument. We distinguish a number of ways that a defeat may occur as follows:

Definition 7 *An argument* $\langle h_1, H_1\rangle$ defeats *an argument* $\langle h_2, H_2\rangle$ *if it rebuts, premise-undercuts, intermediate-undercuts, or inference-undercuts it, where:*

- *An argument* $\langle h_1, H_1\rangle$ rebuts *another argument* $\langle h_2, H_2\rangle$ *iff* $h_1 \equiv \neg h_2$.
- *An argument* $\langle h_1, H_1\rangle$ premise-undercuts *another argument* $\langle h_2, H_2\rangle$ *iff there is a premise* $p \in P(H_2)$ *such that* $h_1 \equiv \neg p$.
- *An argument* $\langle h_1, H_1\rangle$ intermediate-undercuts *another argument* $\langle h_2, H_2\rangle$ *iff there is an intermediate conclusion* $c \in C(H_2)$ *such that* $c \neq h_2$ *and* $h_1 \equiv \neg c$.
- *An argument* $\langle h_1, H_1\rangle$ inference-undercuts *another argument* $\langle h_2, H_2\rangle$ *iff there is an inference rule* $\delta \in \Delta(H_2)$ *such that* $\delta = \frac{p_1, \ldots, p_n}{c}$ *and* $h_1 \equiv \neg(p_1 \wedge \ldots \wedge p_n \rightarrow c)$.

In any case in which $\langle h_1, H_1\rangle$ *defeats* $\langle h_2, H_2\rangle$, $\langle h_1, H_1\rangle$ *is said to be a* defeater *of* $\langle h_2, H_2\rangle$, *and* $\langle h_2, H_2\rangle$ *is said to be the* defeatee. *The relation* defeat *collects all pairs* $(\langle h_1, H_1\rangle, \langle h_2, H_2\rangle)$ *such that* $\langle h_1, H_1\rangle$ *defeats* $\langle h_2, H_2\rangle$.

From Dung [23], we have the following component definitions, all of which hold for the system of argumentation we have described.

Definition 8 *An* argumentation framework *is a pair, Args* $= \langle \mathcal{A}, \mathcal{R} \rangle$*, where* $\mathcal{A}$ *is a set of arguments, and* $\mathcal{R}$ *is the binary relation* defeat *over the arguments.*

Definition 9 *Let* $\langle \mathcal{A}, \mathcal{R} \rangle$ *be an argumentation framework, and* $S \subseteq \mathcal{A}$*. An argument A is defended by S iff* $\forall B \in \mathcal{A}$ *if* $(B, A) \in \mathcal{R}$ *then* $\exists C \in S$ *such that* $(C, B) \in \mathcal{R}$*.*

Definition 10 $S \subseteq \mathcal{A}$*.* $\mathcal{F}_{\mathcal{R}}(S) = \{A \in \mathcal{A} | A$ *is defended by S with respect to* $\mathcal{R}\}$*.*

Now, for a function $F : D \rightarrow D$ where D is the domain and the range of the function, a fixed point of F is an $x \in D$ such that $x = F(x)$. When the D is associated with an ordering P — for example, P can be set inclusion over the power set D of arguments — x is a *least fixpoint* of F if x is a least element of D with respect to P and x is a fixed point.

Definition 11 *Let* $\langle \mathcal{A}, \mathcal{R} \rangle$ *be an argumentation framework. The set of acceptable arguments, denoted by* $Acc^{F}_{\mathcal{R}}$*, is the least fixpoint of the function* $\mathcal{F}_{\mathcal{R}}$ *with respect to set inclusion.*

The least fixpoint semantics can be viewed as a mathematical translation of the principle that an argument survives if it can defend itself and be defended by a set of arguments which can also survive all the attacks made upon its members. Other argumentation semantics from the literature, such as [7, 12, 23] can also be employed.

4 Arguments and Trust

So far we have described how trust is propagated between agents, and how each agent builds arguments. We now combine the two.

4.1 Trust and belief

As we discussed above, we make use of a very simple model of belief, since for the purposes of this paper we only need to capture the fact that an agent's knowledge can be uncertain. We capture this by allowing each agent Ag_i to have a measure of belief $bel_i(\phi)$ for each ϕ in its knowledge base Σ_i. However, an agent Ag_i is not only interested in using information from its own knowledge base, but also information from other agents — for example let's imagine that Ag_i is using ϕ which Ag_j told Ag_i was true. [56] handles this situation by saying that if Ag_i trusts Ag_j. and Ag_j says ϕ is true, then Ag_i believes ϕ. We follow this principle but adapt it for our model where both trust and belief admit degrees.

To do this, we say that Ag_i needs to take into account its degree of trust in Ag_j in formulating its degree of belief in information it gets from Ag_j. We assume that this is done this by using the trust value that Ag_i can compute for Ag_j through the trust network that joins them, as outlined in Section 3.2. Having computed this value, which we will call $tr(Ag_i, Ag_j)$, we further assume that Ag_i can convert this value into a degree of belief that it can use in argumentation, thus assuming a function *trust-to-belief*:

$$ttb : \Re \mapsto \Re$$

that can take any trust value and map it to the correct degree of belief. Depending on the semantics of the degrees of trust and belief, this function may be the identity — for example if the trust values Ag_i has for every Ag_j is simply Ag_i's subjective degree of belief that what Ag_j says is true, which is the notion of trust in [28, 63] — though different semantics for trust and belief would require more complex transformations. ([95] suggests translations between some pairs of belief measures which might be appropriate.) In any case, for Ag_i, its belief in ϕ may be computed:

$$bel_i(\phi) = ttb(tr(Ag_i, Ag_j))$$

In other words the belief that Ag_i has in ϕ is a function of the trust that Ag_i has in Ag_j. What we have so far assumes that Ag_j expresses the opinion that ϕ is true. If Ag_j expresses some degree of belief in ϕ, then $bel_j(\phi)$, Ag_i can compute its degree of belief in ϕ:

$$bel_i(\phi) = ttb(tr(Ag_i, Ag_j)) \otimes^{bel} bel_j(\phi)$$

In other words, the belief that Ag_i has in ϕ is a combination of Ag_j's belief in ϕ and the trust that Ag_i has in Ag_j.

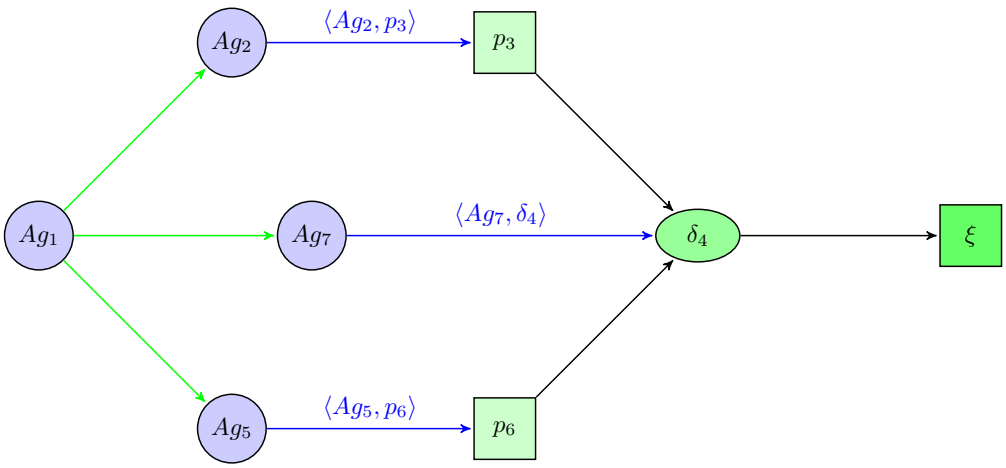

Figure 3: A trust-extended proof network

4.2 Trust-extended argumentation

Since we believe that it is useful to keep track of where different pieces of information came from, for example in case the trust values in our trust network change, we find the concept of the *trust-extended proof network* to be useful. A trust-extended proof network marries trust networks and argument graphs. This concept is formally defined below, and an example is given in Figure 3.

Definition 12 *A* trust-extended proof network $\mathcal{R}^t$ *is a pair* $\langle \mathcal{T}, \mathcal{R} \rangle$ *of an agent-centric trust network* $\mathcal{T}$ *and a proof network* $\mathcal{R}$ *such that every rule* δ_i *and every leaf* p_j *of* $\mathcal{R}$ *are connected to a node* Ag_k *in* $\mathcal{T}$ *by an arc* $\langle Ag_k, \delta_i \rangle$ *or* $\langle Ag_k, p_j \rangle$ *that denote, respectively, that* $\delta_i \in \Delta_k$ *or* $p_j \in \Sigma_k$ *respectively.*

An example of a trust-extended proof network is given in Figure 3. This shows how the acquaintances of Ag_1 each provide the elements of an argument. Ag_2 and Ag_5 each provide a premise of the rule δ_4, while Ag_7 provides the rule itself, and together these support the conclusion ξ. The idea behind the trust-extended proof network is that it relates the premises of an argument to their sources. Such a network therefore captures the reasoning of the agent at the root of the trust network, including which pieces of information it has used from which agents it trusts. It is simple to show that:

Proposition 6 *A trust-extended proof network has one root and possibly many leaves, but only one leaf that is a conclusion of the proof network.*

Proof: *A trust-extended proof network is proof-network where every root in the proof network is linked to a node in an agent-centric trust network. Since is there is, by definition, only one root in the trust network, there is only one root in the trust-extended proof network. Equally, although there may be many leaves of the trust network that are not linked to elements of the proof network, the proof network by definition only has one conclusion, and this will therefore be the only conclusion of the trust-extended network.* □

Thus a trust-extended proof network relates a single agent to a single conclusion, and we can easily extend the notion of an argument — which as we recall from Definition 6 is a pair consisting of a proof network and its conclusion — with the trust information of a trust-extended proof network:

Definition 13 *A trust-extended argument* A^t *from the union of a set of knowledge bases* $\{\Sigma_1, \ldots \Sigma_n\}$ *belonging to a set of agents* $Ags = \{Ag_1, \ldots Ag_n\}$*, all of which are in* $\mathcal{T}$*, is a pair* $\langle h, \mathcal{R}^t \rangle$ *where* $\mathcal{R}^t$ *is a trust-extended proof network for h, and h is the only leaf of* $\mathcal{R}^t$*.*

In the same way that an argument is relative to a knowledge base, so a trust-extended argument is relative to a set of agents, and, in particular, to the set of knowledge bases of those agents. Furthermore, the conclusions of a trust-extended argument are relative to a specific agent. Given a trust-extended argument, the only agent that is sanctioned to infer the conclusion of the argument is the one at the root of the trust graph. Thus, like the graph, the conclusions are agent-centric. Figure 4 presents two trust extended arguments. The leftmost, in Figure 4(a), contains the same information as a previous example, but now — in keeping with the spirit of trust-extended arguments — identifies the origin of the information. In this case it is John whose reasoning is being captured, and this agent has access to the information:

$$\Sigma_{john} = \{IndieFilm(hce), SpanishFilm(hce), DirectedBy(hce, Almodovar)\}$$

while John's acquaintance Dave says that:

$$\delta_{dave} = \frac{IndieFilm(x) \wedge DirectedBy(x, Almodovar)}{Watch(x)}$$

allowing John to construct an argument for $Watch(hce)$.

The second argument, in Figure 4(b), makes use of information from another acquaintance, Jane, who holds that:

$$\delta_{jane} = \frac{IndieFilm(x) \wedge SpanishFilm(x)}{\neg Watch(x)}$$

from which John can construct an argument for $\neg Watch(hce)$.

The last element of our model is the trust-extended argument graph, in which we show the relations between a set of trust-extended arguments. Informally, this is a set of trust-extended arguments with the defeat relationships between the arguments denoted by labelled edge. There are four kinds of defeat edge, one for each of the kinds of link identified in Definition 7.

Definition 14 *A* trust-extended argument graph AG^t *from the union of a set of knowledge bases* $\{\Sigma_1, \ldots \Sigma_n\}$ *belonging to a set of agents* $Ags = \{Ag_i, \ldots Ag_n\}$*, all of which are in* $\mathcal{T}$ *is the union of a set of trust-extended arguments* A_i^t *with an additional set D of edges which we call* defeat edges*.*

There are four kinds of defeat edge: a rebut *edge, a* premise-undercut *edge, a* intermediate-undercut *edge and an* inference-undercut *edge. Each defeat edge* $d \in D$ *links two arguments* $A_i^t = \langle h_i, \mathcal{R}_i^t \rangle$ *and* $A_j^t = \langle h_j, \mathcal{R}_j^t \rangle$*.*

- A_i^t *and* A_j^t *will be joined by a* rebut *edge iff* $h_i \equiv \neg h_j$*.*
- A_i^t *and* A_j^t *will be joined by a* premise-undercut *edge iff there is some rule* δ *in* $\mathcal{R}_j^t$ *with a premise p and* $h_i \equiv \neg p$*.*
- A_i^t *and* A_j^t *will be joined by a* intermediate-undercut *edge iff there is some rule* δ *in* $\mathcal{R}_j^t$ *with conclusion c and* $h_i \equiv \neg c$*.*
- A_i^t *and* A_j^t *will be joined by a* inference-undercut *edge iff there is some rule* δ *in* $\mathcal{R}_j^t$ *with premises* $p_1, \ldots, p_n$ *and conclusion c, and* $h_i \equiv \neg(p_1 \wedge \ldots \wedge p_n \rightarrow c)$*.*

In any case in which A_1^t *defeats* A_2^t*,* A_1^t *is said to be a* defeater *of* A_2^t*, and* A_2^t *is said to be the* defeatee*.*

This definition clearly mirrors Definition 7, and with good reason. A trust-extended argument graph is a graphical counterpart of the argumentation framework of Definition 8, and we can identify sets of acceptable arguments from the graph just as we can from an argumentation framework.

Given the equivalence of the graphical framework we introduce here and previous non-graphical frameworks, a reasonable question is why we bother with the graphical framework. There are two answers. One is that the graphical approach links neatly with trust networks, and gives us a uniform representation to use as the basis for computations involving argumentation and trust — we describe how this may be done in Sections 7 and 8. The second answer is that our long-term goal is to use the framework we describe as the basis for a decision support tool that helps human users to understand the impact of trust on their decisions. Previous work on graphical representation of arguments, for example [10, 49, 66, 76, 80, 88, 89], suggests that a graphical representation will be a good way to do this, and here we have a graphical representation that can exactly mirror the computation of acceptability of arguments.

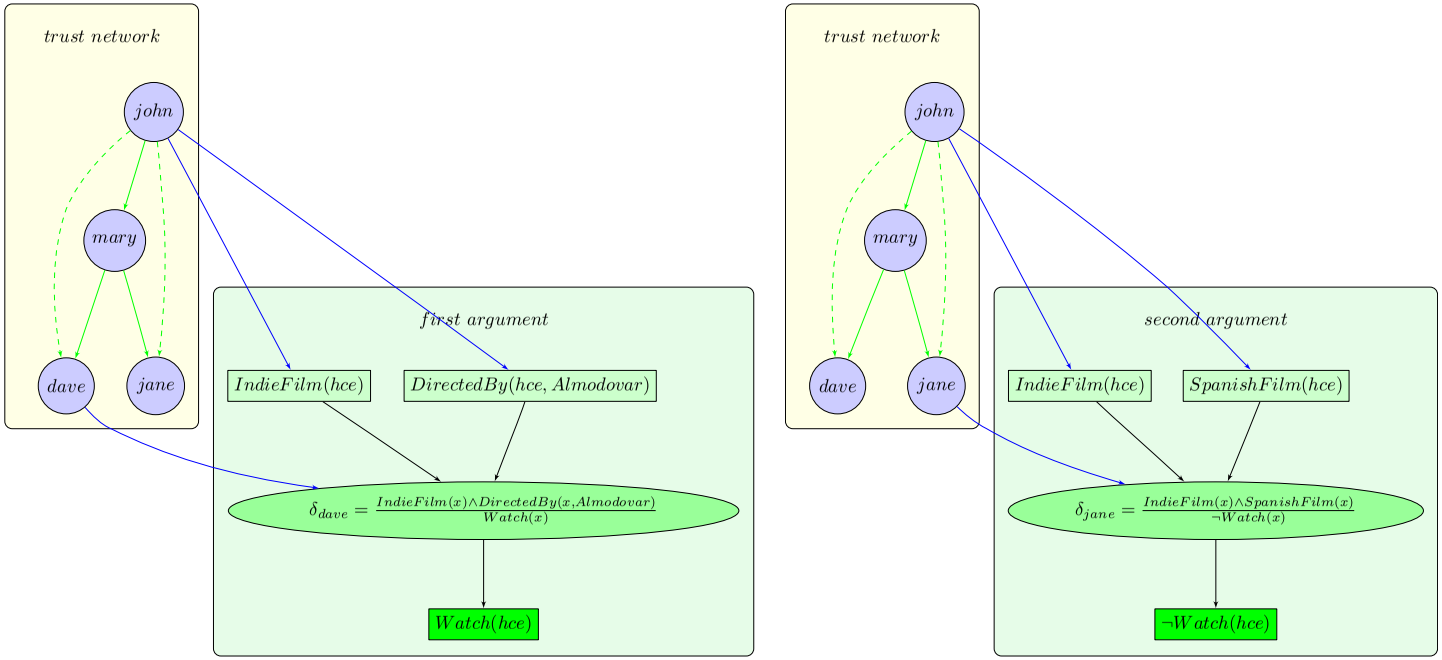

Figure 4: Two trust-extended arguments that John can construct

An example of a trust-extended argument graph is shown in Figure 5. This particular graph, which corresponds to the example we have discussed throughout this paper (and which we will continue to work with) shows the two arguments that John can develop (previously seen as separate trust-extended arguments in Figure 4) as one trust-extended argument graph. As before, the section in the middle of the graph is a trust network which shows the relationship between John, Mary, Dave and Jane. The two sections on either side of the trust network show arguments that John can develop. Each of these uses information from John's knowledge (the rectangular nodes linked to the *john* node in the trust graph) and an inference rule from one of John's acquaintances (the oval nodes linked to the *dave* and *jane* nodes in the trust graph). The conclusions of these two arguments (the rectangular nodes at the bottom of the graph) are joined by two rebut edges since the two arguments rebut each other, disagreeing with each other's conclusions.

5 Instantiations of the abstract model

So far, our discussion of the trust models, and the way that the trust information is used in argumentation, has been rather abstract. In this section, we make the discussion more concrete by describing some instantiations of the model. In particular, in this section we discuss operations that might be used for $\oplus^{tr}$ and $\otimes^{tr}$ in combining trust values.

5.1 Possibility theory

As suggested by [79], possibility theory [18, 19, 20] is a natural way to interpret trust values. In possibility theory, itself derived from the notion of fuzzy sets [93, 94], numbers are attached to propositions to quantify the degree of uncertainty in the propositions, but the values are not probabilities. Instead they are taken to represent the degree of *possibility* that the proposition is true. The possibility of a proposition p is written as $\Pi(p)$.

Informally, the degree of possibility can be taken to suggest how surprised one would be to discover that the proposition is true. A proposition with a degree of possibility of 1 is a proposition that is thought to be completely possible, so one that would occasion no surprise were it found to be the case. A proposition with a lower degree of possibility would create some surprise were it found to be true.

There are several families of functions that may be used to combine possibility values, but the most commonly used are the following. The possibility of a conjunction $p \wedge q$ is taken to be greater than or equal to the minimum of the possibility of p and the possibility of q:

$$\Pi(p \wedge q) \geq \min(\Pi(p), \Pi(q))$$

and so if we capture trust using the lower bound of the possibility values, then $\otimes^{tr}$ is $\min$. Similarly the possibility of a disjunction $p \vee q$ is typically taken to be the maximum of the possibilities of p and q:

$$\Pi(p \vee q) = \max(\Pi(p), \Pi(q))$$

and so using possibility theory to model trust it is natural to take $\oplus^{tr}$ to be $\max$.

5.2 Subjective logic

The Dempster-Shafer theory [82, 83] is a generalization of probability theory (and, indeed, possibility theory [21]) in which probabilities can be assigned not just to individual propositions, but also to sets of propositions. This, it is argued, allows for the expression of a degree of ambiguity — if we have evidence that suggests either p or q is the case we can assign probability to the set $\{p, q\}$ and wait to see if additional evidence allows us to refine this opinion. A typical way to make use of this ability is to assign probability to some proposition p, its negation $\neg p$ and the set $\{p, \neg p\}$, with the latter representing the belief that one cannot, for lack of evidence, assign to either p or its negation. This can be interpreted as the degree to which one cannot tell whether p is true or not.

Subjective logic [42, 43, 64] uses Dempster-Shafer theory in exactly this way, modelling the trust in every proposition as a triple, and this is the approach taken by [90]. In this latter work, the trust links are labeled by a belief measurement:

$$tr(Ag_i, Ag_j) = M_{i,j}$$

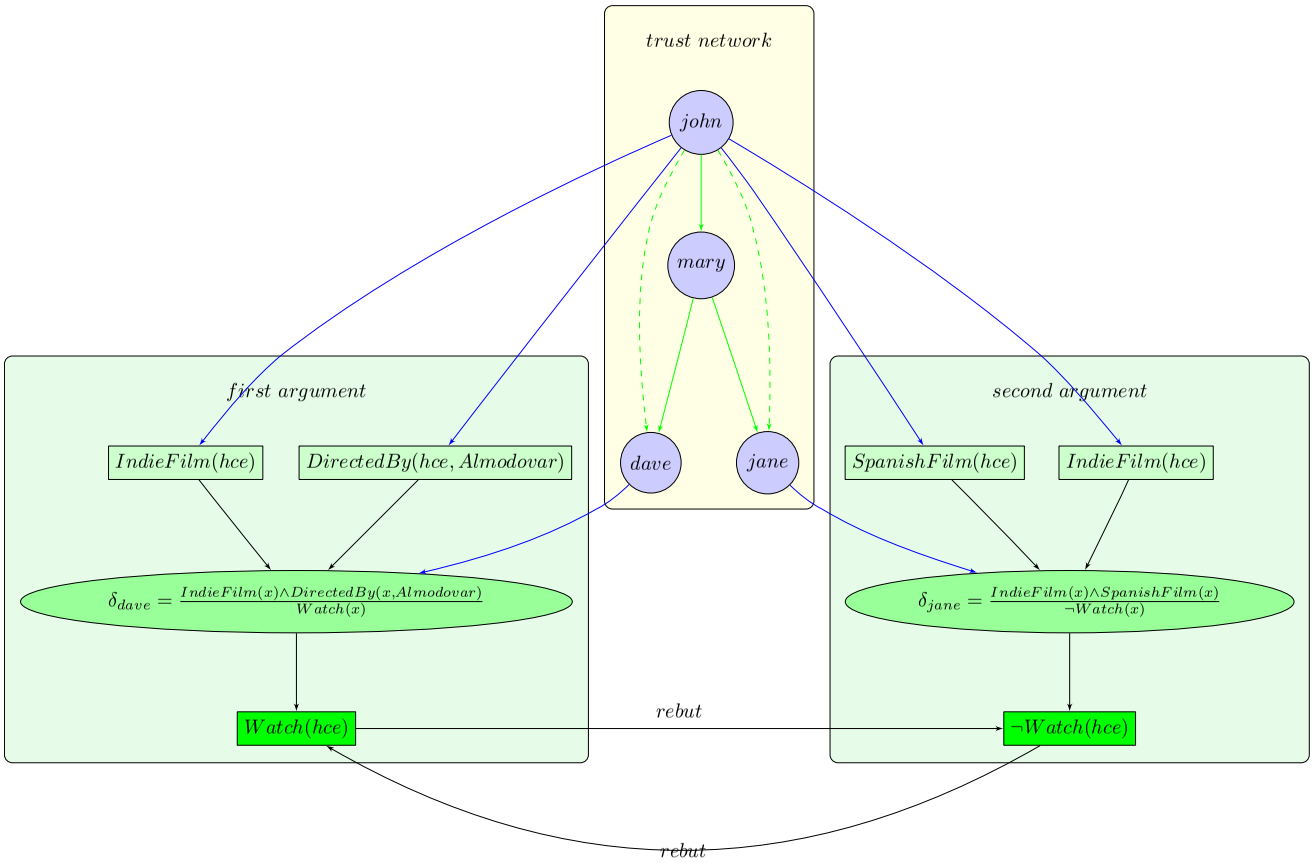

Figure 5: A trust-extended argumentation graph capturing John's view of the film example

where $M_{i,j} = \langle b, d, u \rangle$ and b, denoted by $b(m_{i,j})$, is the probability that what Ag_j tells Ag_i will be true is actually true; d, denoted by $d(m_{i,j})$, is the probability that what Ag_j tells Ag_i is true is actually false; and u, denoted by $u(m_{i,j})$, is the probability of uncertain outcomes (that Ag_i, for example, will not know if what it is told is true or not)[5]. Subjective logic, with its triple $M_{i,j}$ representing trust, is thus an example of a trust measure where opinions $\mathcal{O}$ are more complex than just a single number.

Wang and Singh [90] describe how these values can be linked to an agent's experience so that the values $b(m_{i,j})$, $d(m_{i,j})$ and $u(m_{i,j})$ are determined by how often what agent j tells agent i turns out to be the case. In particular Wang and Singh consider that agents keep track of the number of *positive outcomes* (where something that they are told is true turns out to be true, or something they are told is false turns out to be false) and *negative outcomes* (where something that they are told is true turns out to be false, or vice versa) for each agent they interact with. Then:

Definition 15 *The* evidence space *is defined as*

$$E = \{(r, s) | r > 0, s > 0\}$$

where r is the number of positive outcomes and s is the number of the negative outcomes, and the belief space *is defined as:*

$$B = \{(b, d, u) | b > 0, d > 0, u > 0, b + d + u = 1\}$$

This just says that for each agent that Ag_i interacts with, there are positive and negative instances of interaction, and, separately, there is the triple measure (b, d, u) that allows the outcome of future interactions to be estimated. Clearly we want to derive the latter from the former, and we do this as follows.

Definition 16 *Let $Z = (B, D, U)$ be a transformation from E to B such that $Z(r, s) = (B(r, s), D(r, s), U(r, s))$ where:*

$$\begin{aligned} B(r, s) &= c(r, s)\frac{r + 1}{r + s + 2} \\ D(r, s) &= c(r, s)\frac{s + 1}{r + s + 2} \\ U(r, s) &= 1 - c(r, s) \end{aligned}$$

and where $c(r, s)$ is the certainty level *computed by a statistical model of certainty given the evidence of the trust links.*

Thus the belief space is computed rather straightforwardly from the history of interactions provided that we have a model of certainty. One model of certainty is to let

$$c(r, s) = \frac{1}{2}\int_0^1 |f_{r,s}(x) - 1| dx$$

where

$$f_{r,s}(x) = \frac{x^r(1 - x)^s}{\int_0^1 x^r(1 - x)^s dx}$$

$f_{r,s}(x)$ is the posterior (cumulative) probability of the positive outcomes (of a binary event with x as its prior) after observing r positive outcomes and s negative outcomes. $c(r, s)$ is then defined as mean absolute deviation.

Given this model, the operators $\otimes^{tr}$ and $\oplus^{tr}$ can be defined as the following.

Definition 17 *Suppose $M_1 = (b_1, d_1, u_1)$ and $M_2 = (b_2, d_2, u_2)$, then $M = M_1 \otimes M_2 = (b, d, u)$ where*

$$\begin{aligned} b &= b_1 b_2 \\ d &= b_1 d_2 \\ u &= 1 - b_1 b_2 - b_1 d_2 \end{aligned}$$

[5]Conceptually, at least, this uncertainty can later be resolved, and the probability or true or false increased accordingly.

Definition 18 *Suppose $M_1 = (b_1, d_1, u_1)$ and $M_2 = (b_2, d_2, u_2)$, and let (r_1, s_1) and (r_2, s_2) be the elements in the event space of M_1 and M_2 respectively, then $M = M_1 \oplus M_2 = (b, d, u)$ where*

$$\begin{aligned} b &= B(r_1 + r_2, s_1 + s_2) \\ d &= D(r_1 + r_2, s_1 + s_2) \\ u &= U(r_1 + r_2, s_1 + s_2) \end{aligned}$$

5.3 TidalTrust

The two models we have discussed so far are taken from the literature of reasoning under uncertainty. In other words, using these models to represent and compute trust values is assuming, as suggested by [28, 63], that trust in an individual is the subjective belief that the individual tells the truth. An alternative view of trust is discussed in [50], where the authors' aim is to construct a model that agrees with the way that people compute with trust values. (The values themselves also differ — while possibility theory and subjective logic deal with values in the range $[0, 1]$, [50] uses values between 0 and 10.) The result is the TidalTrust model, which describes how to propagate values through the kind of trust network we described earlier.

In particular, in the context of a extended trust network $\mathcal{T} = \langle Ags, \{\tau\}\rangle$, we can use the TidalTrust model to propagate trust along paths through the network by recursively computing:

$$tr(Ag_i, Ag_j) = \frac{\oplus^{tr}_{(Ag_i,Ag_k)\in\mathcal{T} \text{ and } tr(Ag_i,Ag_k)\geq\theta} tr(Ag_i, Ag_k) \otimes^{tr} tr(Ag_k, Ag_j)}{\oplus^{tr}_{(Ag_i,Ag_k)\in\mathcal{T} \text{ and } tr(Ag_i,Ag_k)>\theta} tr(Ag_i, Ag_k)}$$

where θ is a parameter to ensure the model only considers trust values above a certain threshold thus ignoring small trust values, $\oplus^{tr}$ is implemented as the arithmetic addition "$+$" and $\otimes^{tr}$ is implemented as the arithmetic multiplication "$\times$".

The above formula carries out a breath-first search through the trust network. To evaluate an agent Ag_i's trust on another agent Ag_j, the algorithm has each neighbor agent Ag_k of Ag_i obtain their evaluations of the trust to the agent Ag_j (in a same way), and then Ag_i computes the weighted average of its neighbors' trust values on Ag_i with the weights being set to its trust values on each neighbor Ag_k.

6 Examples

In this section we show how the formal system we have introduced can capture two versions of the the example from [50].

6.1 The original example

This example, from which we have been drawing throughout the paper, considers reasoning in the FilmTrust [26, 34] database. This is an online database where people offer ratings for films they have seen and search for recommendations for films to watch. FilmTrust members also rate each other, indicating who they trust to give good advice on films. Although we have seen various pieces of the example already, we will start with a recap.

In the example, we are concerned with a certain agent John who is invited to watch a film by one of his friends. John is part of the trust network from Figure 1(a), and furthermore has the following information about the film in question [6]:

$$\{IndieFilm(hce), SpanishFilm(hce), DirectedBy(hce, almodovar)\}$$

In the original example, John also has the rule:

$$\delta_{john} = \frac{Comedy(x)}{\neg Watch(x)}$$

[6] "Almodovar" here is Pedro Almodovar, and "hce" is an abbreviation for his 2002 film *Hable con ella* (Talk to her). It is, of course, arguable whether "hce" is an independent film, but since the original example considered it to be one, so will we.

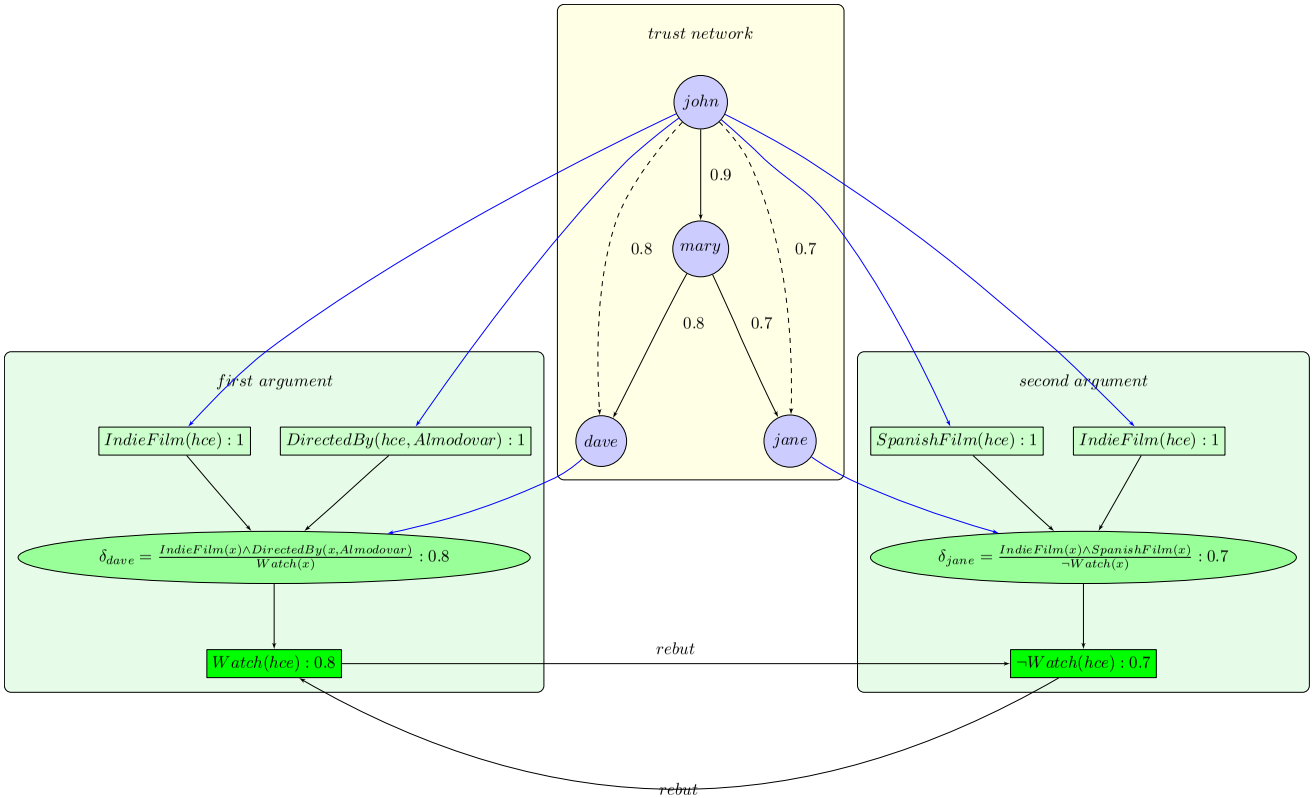

Figure 6: A trust-extended argumentation graph capturing John's view of the original film example, labelled with trust values

which indicates that he doesn't like to watch comedies.

Since this information doesn't help him to decide whether to watch the film, John asks people in his social network (for all of whom he can compute a trust rating) for their opinions and learns:

$$\delta_{jane} = \frac{IndieFilm(x) \wedge SpanishFilm(x)}{\neg Watch(x)}$$
$$\delta_{dave} = \frac{IndieFilm(x) \wedge DirectedBy(x, almodovar)}{Watch(x)}$$

From this information, John can, as we have already discussed, construct two simple arguments which are depicted in Figure 6, and which we can translate as:

> I should watch *hce* because it is an independent film and it is directed by Almodovar, and Dave says that any independent film directed by Almodovar is watchable.

and

> I shouldn't watch *hce* because it is a Spanish independent film, and Jane says that Spanish independent films are unwatchable.

These arguments rebut each other and John can use information about his friends in order to further refine his view of the arguments. However, to do that he has to choose a specific instantiation of the trust values. Let us assume he models trust using possibility theory. Then, as discussed in the previous section, $\otimes^{tr}$ is $\min$. Considering the trust graph in Figure 1(a) it is easy for John to compute that:

$$tr(john, dave) = 0.8$$
$$tr(john, jane) = 0.7$$

and these are the values we see in Figure 6.

Now, these values need to be turned into belief values. Since the trust values are expressed using possibility theory it is natural to handle belief in the same way — this means that we handle the notion of strength of arguments in a way that is close to that adopted by [4] though the authors of that paper do not explicitly mention the use of possibility theory. Using the same values for trust and belief means that we interpret the degree of trust that one agent, Ag_i, has in another, Ag_j, to be the degree of belief that Ag_i has that what Ag_j says is true (exactly as in [28, 63]).

In other words the degree of belief that John has in a statement from Dave is exactly John's degree of trust in Dave. As a result:

$$bel_{john}(\delta_{dave}) = 0.8$$
$$bel_{john}(\delta_{jane}) = 0.7$$

In order to reach conclusions about watching *hce*, these rules need to be combined with John's initial knowledge, all of which he believes to be completely possible so that:

$$bel_{john}(IndieFilm(hce)) = 1$$
$$bel_{john}(SpanishFilm(hce)) = 1$$
$$bel_{john}(DirectedBy(hce, almodovar)) = 1$$

Since these are possibility values, we use minimum for $\otimes^{bel}$, and so:

$$bel_{john}(Watch(hce)) = 0.8$$
$$bel_{john}(\neg Watch(hce)) = 0.7$$

We have not discussed how to use these belief values in conjunction with argumentation (we will come back to this topic below), but since as we mentioned, we are handling belief rather like Amgoud and Cayrol [4], we can take their approach to resolving defeat using belief. In other words, one argument defeats another if it defeats it in the sense of Definition 14 *and* its conclusion has a higher degree of belief. In this case, John will decide to watch *hce*.

While this is a simple example, it shows that our approach handles the combination of trust and argumentation in an intuitively appealing way, as well as agreeing with the analysis in [50].

6.2 The modified example

The previous example was included to show how our system can capture the reasoning from [50]. However, argumentation is typically less interested in the kind of symmetrical "rebut" conflict between the two arguments seen in that example than it is in the asymmetrical "undercut" conflict. The modified example will illustrate undercutting. We consider that John has a slightly different set of information:

$$\{SpanishFilm(hce), DirectedBy(hce, almodovar)\}$$

which does not include the information about *hce* being an indie film (as one reviewer pointed out, *hce* is not really an indie film). Rather he has the rule:

$$\delta_{john} \quad = \quad \frac{DirectedBy(x, almodovar)}{\neg IndieFilm(x)}$$

indicating that he believes that no Almodovar film is an indie film. We will further assume that, this time around, when John polls his social network about watching the film, he gets no reply from Dave, but does hear from Mary (who thinks *hce* is an independent film), and from Jane who replies as in the previous example. Thus John has the information that:

$$\begin{aligned} \delta_{jane} &= \frac{IndieFilm(x) \wedge SpanishFilm(x)}{\neg Watch(x)} \\ p_{mary} &= IndieFilm(hce) \end{aligned}$$

This time John has the arguments of Figure 7. Handling trust values as before — again using the approach from [4] — John will find that his argument against *hce* being an independent film defeats the argument for not watching *hce*. (Of course this leaves John with no information about whether to watch the film, he just knows that he doesn't have an acceptable argument to not watch it).

7 Constructing trust-extended argumentation graphs

So far in this paper we have introduced a graphical representation of a variant of Dung's argumentation framework which also captures information about which agents hold which pieces of information that are used to construct an argument, and what the trust relationship between the agents is. In this section we describe how to construct these trust-extended argumentation graphs. Here we separate the construction procedure (Algorithm 2) from the trust and belief propagation for clarity. However, in practice, we might wish to combine the two procedures together to avoid constructing unnecessary arguments.

From the algorithmic point of view, instead of constructing each proof network and the corresponding argument explicitly, we mark each proof network by its unique ID, and each arguments by its unique ID. The graph construction is based on the construction of proof networks in Algorithm 1 (below).

Proposition 7 *Given a query q, Algorithm 1 creates all possible proofs of q. Each proof of q is marked with a unique ID, $id \in ID(G, node(q))$, and each proof with id is associated with a sub-proof network $SubIDs(G, id)$.*

Proof: Algorithm 1 goes through each inference rule $\delta \in \Delta$ whose conclusion can be unified with q, and tries to construct a proof network for each premise of δ. If all premises have proof networks, the algorithm then connects $node(q)$ to $node(\delta)$, and $node(\delta)$ to the nodes corresponding to the premises of δ. At the same time, the algorithm records each combination of the proofs for δ's premises as a proof network for q and marks it down with a unique ID. This unique ID is recorded into the set of proofs for q: $ID(G, node(q))$ □

In Algorithm 1, the ID set for a node $node(q)$, denoted by $ID(G, node(q))$, records a set of arguments for q. Given an ID $id \in ID(G, node(q))$, the $SubIDs(G, id) = \langle id_1, id_2, ..., id_k \rangle$ tracks the proof sub-networks identified by $\langle id_1, id_2, ..., id_k \rangle$. With this ID construction, we can then recover each proof network of Definition 12 by traversing

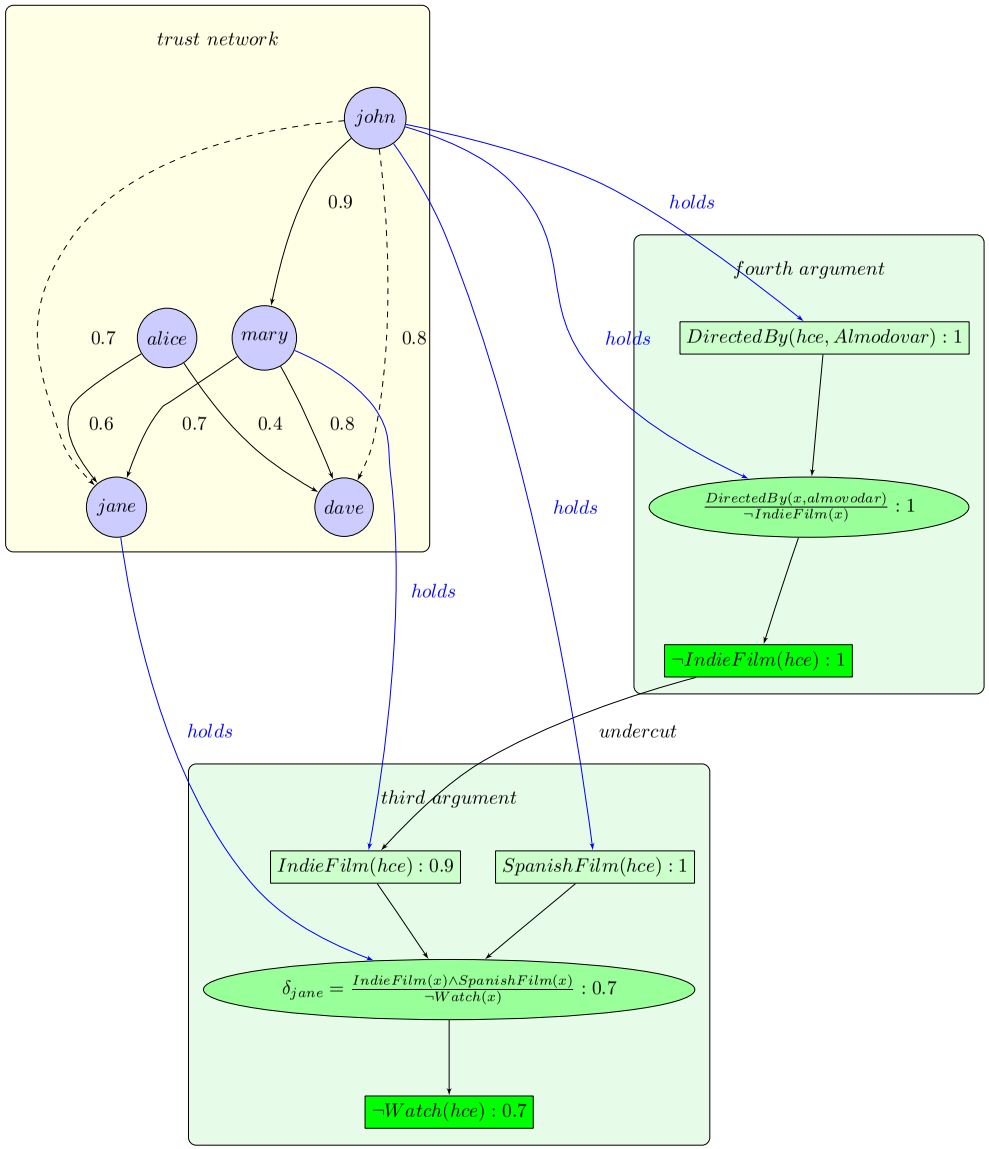

Figure 7: A trust-extended argumentation graph capturing John's view of the modified film example, labelled with trust values

$ID(G, node(q))$ and $SubIDs(G, ID(G, node(q))$ recursively until a knowledge base Σ_i and a set of inference rules Δ_j are reached. At this point the components of an argument are connected to the trust network. With these expanded proof networks, we can then recover the arguments of Definition 13. With Algorithm 1 as a sub-routine, Algorithm 2 can create an argumentation graph for q efficiently.

Proposition 8 *Given a query q, Algorithm 2 creates an argumentation graph for q.*

- *For each node in the argument graph, if there are proof networks for it, all the proof networks are included in the argument graph; all the proof networks included are valid.*
- *For each node in the argument graph, if there are defeaters, the defeaters are included in the argument graph.*
- *If there is a defeat between two arguments, it is marked in* $ID(defeater, defeatee)$.

Proof: Algorithm 2 calls Algorithm 1 which constructs all the valid proof networks for each node. It also goes through all the nodes, and tries to construct proof networks and defeaters recursively with Algorithm 1 for all these nodes. □

In a similar manner to argument construction, Algorithm 2 records the IDs of arguments where one argument defeats another argument associated with two nodes nd_1 and nd_2 (which are either premise nodes, conclusion nodes or inference nodes) in the argument graph using $ID_{defeat}(\langle nd_1, nd_2\rangle)$. The set of argument ID pairs which are recorded in ID_{defeat} (for all nodes recorded) recovers the defeat relation in Definition 8. We can then use the defeat relation recorded in ID_{defeat} to compute various argumentation semantics, such as the one in Definition 11. Following a similar approach to [3] the same defeat relation recorded in ID_{defeat} can be refined with the trust and belief values, and in turn the argumentation semantics can be refined.

As Algorithm 1 and Algorithm 2 work in a dynamic programming manner, the complexity of these procedures is determined by the size of the syntactic structure of the knowledge base and the inference rule base, and the number of all possible arguments in the output.

Proposition 9 *Assume that that rule base* Δ *only contains first order schemes (i.e. there are domain variables in the rules but no predicate variables). Let* $P_\Delta = \cup_{\delta\in\Delta} Premises(\delta)$, $C_\Delta = \cup_{\delta\in\Delta}\{c(\delta)\}$ *and the K be the maximum arity of predicates and M be the maximum number of values can be assigned to the variables of predicates. The number of reasoning links in the argumentation graph for a query q created by Algorithm 2 is bounded by* $O(K^2 \times M^2 \times (1 + |P_\Delta|) \times (|\Sigma| + C_\Delta))$. *The number of defeat links is bounded by* $O(K^2 \times M^2 \times (1 + |P_\Delta| + |\Sigma| + C_\Delta)) \times (|\Sigma| + C_\Delta))$.

Proof: The number of all possible conclusions and intermediate conclusions is $(|\Sigma| + |C_\Delta|) \times K \times M$, and the number of all possible premises is $(1 + |P_\Delta|) \times K \times M$. All possible reasoning links are between the set of all possible (intermediate) conclusions and the set of all possible premises, therefore the number of reasoning links is bounded by $O(K^2 \times M^2 \times (1 + |P_\Delta|) \times (|\Sigma| + C_\Delta))$. On the other hand, all possible defeat links are between the set of all possible conclusions and the union of all the possible premises and all the possible (intermediate) conclusions, therefore the number of defeat links is bounded by $O(K^2 \times M^2 \times (1 + |P_\Delta| + |\Sigma| + C_\Delta)) \times (|\Sigma| + C_\Delta))$. □

Note that, although we have an argumentation graph with size bounded polynomially in the number of premises the and number of inference rules, the number of arguments can still be exponential in the umber of premises the and number of inference rules. We deliberately record this exponential number of arguments and their interactions in the form of IDs in Algorithm 1 and Algorithm 2. Part of our future research focuses on how to utilize this compact graphical structure for nearly tractable algorithms for trust and belief propagation with trust extended argumentation. In this work, we only consider first order reasoning rules with no function symbols. For second order rules, such as the natural deduction rules of [59] with predicate variables over the logical connectives, we will need an additional reasoning mechanism to efficiently instantiate the deduction rules to connect the unified formulae. This extension is also part of our planned future research.

A trust extended argumentation graph that can be created by Algorithm 2 from our running example is given in Figure 8. With IDs expanded into the corresponding arguments and centered on John's view, we get the same trust extended argumentation graph is as in Figure 5.

Algorithm 1: *constructProofNet*$(G, q, \mathcal{T}, \Sigma, \Delta)$: Construct a proof network for q

Input: G: a shared graph structure; q: query; $\mathcal{T}$: trust network; Σ: knowledge base; Δ: rule base
if $node(q) \in G$ **then**
  return $node(q)$;
end
if $q \in \Sigma$ **then**
  Create a new node $node(q)$ in G;
  for *each agent* Ag_i *such that* q *is in agent* Ag_i*'s knowledge base* Σ_i **do**
    Add a holding link $\langle node^{\mathcal{T}}(Ag_i), node(q)\rangle$ to G;
  end
  Allocate a new id;
  $ID(G, node(q)) \leftarrow \{id\}$;
  return $node(q)$;
end
if *No inference rule* $\delta \in \Delta$ *with conclusion* $c(\delta)$ *unifiable with* q **then**
  return $\emptyset$;
end
Create a new node $node(q)$ into G;
$ID(G, node(q)) \leftarrow \emptyset$;
for *each inference rule* $\delta \in \Delta$ *with conclusion* $c(\delta)$ *unifiable with* q **do**
  Compute a unifier $\theta \leftarrow unify(q, c(\delta))$;
  for *each rule premise* $p_i(\delta[\sigma])$ **do**
    $result[p_i] \leftarrow constructProofNet(G, p_i, \mathcal{T}, \Sigma, \Delta)$;
  end
  if *all* $result[p_i]$*s are not empty* **then**
    Refine the unifier θ with the one produced for proof net $result[p_i]$s;
    Create a new node $node(\delta[\theta])$ in G if $node(\delta[\theta]) \notin G$;
    Add $\langle \delta[\theta], node(q)\rangle$ to G;
    Add $\langle p_i, r[\theta]\rangle$ to the G for each p_i;
    for *each agent* Ag_i *such that* δ *is in* Ag_i*'s rule base* **do**
      Add a holding link $\langle node^{\mathcal{T}}(Ag_i), \delta[\theta]\rangle$ to G;
    end
    /* Every combination of the proofs for premises is a new unique proof for q */
    for *each combination* $\langle id_1, id_2, ..., id_k\rangle \in \Pi_i ID(G, node(p_i))$ **do**
      Allocate a new id;
      $ID(G, node(q)) \leftarrow ID(G, node(q)) \cup \{id\}$;
      $SubIDs(G, id) \leftarrow \langle id_1, id_2, ..., id_k\rangle$;
    end
  end
end
return $node(p)$;

Algorithm 2: *constructArgGraph*$(G, q, \mathcal{T}, \Sigma, \Delta)$: Construct trust extended argumentation graph

Input: G: a shared graph structure; q: query; $\mathcal{T}$: trust network; Σ: knowledge base; Δ: rule base
$G \leftarrow \emptyset$;
qNode $\leftarrow$ *constructProofNet*$(G, q, \mathcal{T}, \Sigma, \Delta)$;
if *qNode* $\neq \emptyset$ **then**
  if *qNode is not visited by argumentation construction yet* **then**
    defeater $\leftarrow$ *constructArgGraph*$(G, \neg n, \mathcal{T}, \Sigma, \Delta)$;
    if *defeater* $\neq \emptyset$ **then**
      Add a rebut-defeat link $\langle$*defeater*$, n\rangle$ to G if it doesn't exist;
      Initialize $ID_{defeat}(\langle$*defeater*$, n\rangle)$ if it is not initialized;
      for *each combination* $\langle id, id'\rangle \in ID($*defeater*$) \times ID($*qNode*$)$ **do**
        Add $\langle id, id'\rangle$ into $ID_{defeat}(\langle$*defeater*$, n\rangle)$;
      end
    end
  end
  for *each node n in G connecting to q through reasoning links* **do**
    if $n \in \Sigma$ *is premise node and it is not visited yet* **then**
      defeater $\leftarrow$ *constructArgGraph*$(G, \neg n, \mathcal{T}, \Sigma, \Delta)$;
      if *defeater* $\neq \emptyset$ **then**
        Add a premise-undercut link $\langle$*defeater*$, n\rangle$ to G if it doesn't exist;
        Initialize $ID_{defeat}(\langle$*defeater*$, n\rangle)$ if it is not initialized;
        for *each combination* $\langle id, id'\rangle \in ID($*defeater*$) \times ID($*qNode*$)$ **do**
          Add $\langle id, id'\rangle$ into $ID_{defeat}(\langle$*defeater*$, n\rangle)$;
        end
      end
    end
    if *n is an intermediate conclusions and it is not visited yet* **then**
      defeater $\leftarrow$ *constructArgGraph*$(G, \neg n, \mathcal{T}, \Sigma, \Delta)$;
      if *defeater* $\neq \emptyset$ **then**
        Add a intermediate-undercut link $\langle$*defeater*$, n\rangle$ to G if it doesn't exist;
        Initialize $ID_{defeat}(\langle$*defeater*$, n\rangle)$ if it is not initialized;
        for *each combination* $\langle id, id'\rangle \in ID($*defeater*$) \times ID($*qNode*$)$ **do**
          Add $\langle id, id'\rangle$ into $ID_{defeat}(\langle$*defeater*$, n\rangle)$;
        end
      end
    end
    if $n \in \Delta$ *is an inference rule node and it is not visited yet* **then**
      defeater $\leftarrow$ *constructArgGraph*$(G, \neg(\bigwedge_{p_i \in Premises(r)}\{p_i\} \rightarrow conclusion(c)), \mathcal{T}, \Sigma, \Delta)$;
      if *defeater* $\neq \emptyset$ **then**
        Add an inference-undercut link $\langle$*defeater*$, n\rangle$ if it doesn't exist;
        Initialize $ID(\langle$*defeater*$, n\rangle)$ if it is not initialized;
        for *each combination* $\langle id, id'\rangle \in ID($*defeater*$) \times ID($*qNode*$)$ **do**
          Add $\langle id, id'\rangle$ into $ID_{defeat}(\langle$*defeater*$, n\rangle)$;
        end
      end
    end
  end
end
return *qNode*;

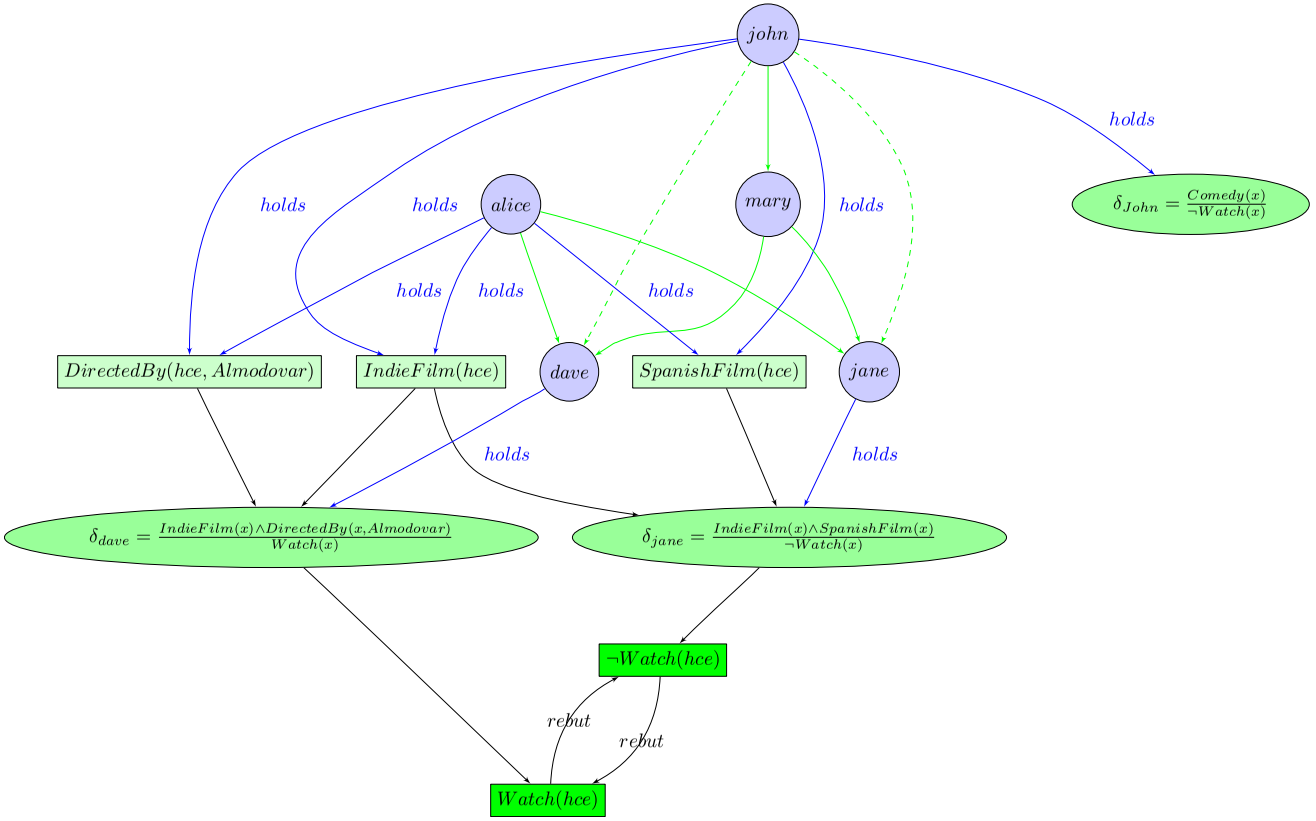

Figure 8: A compact trust-extended argumentation graph

8 Trust propagation

Having shown how to construct trust-extended argumentation graphs, in this section we turn to the question of how to propagate information about trust in those graphs — after all we introduced these graphs exactly to be able to combine trust data with argumentation. Before we can introduce the trust propagation algorithms, we first define the data structure *trustBel* that we use to capture an agent Ag_i's trust or belief in a node n:

$$trustBel[Ag_i, n] = \begin{cases} tr(Ag_i, n) & \text{if } n \text{ is an agent,} \\ bel_i(n) & \text{if } n \in \Sigma_i \text{ or } n \in \Delta_i. \end{cases}$$

To compute the trust-based belief of an agent over an argument, we employ a set of algorithms: Algorithm 3, Algorithm 4, Algorithm 5 (or Algorithm 6 if we take into account the reasoning structure of the argument when computing the beliefs), Algorithm 8 and Algorithm 9 (computing trust in a depth-first manner) or Algorithm 10 (if we need to compute the trust in a breadth-first manner), with two customized functions *combine* and *fuse*.

Algorithm 3 initializes the data structure *trustBel*, which will hold an agent's trust in another agent or an agent's beliefs over premises and inference rules, to *NIL*. Algorithm 9 computes the trust of one agent in the other agents in a depth-first manner using the customized *combine* and *fuse* functions as discussed below. If the trust model requires a breadth-first trust computation (as TidalTrust does), we can then employ Algorithm 10. With the trust computed, we can then use Algorithm 5 to compute beliefs over arguments as a whole. An alternative way to compute beliefs over arguments is to use Algorithm 6 and Algorithm 7 if the underlying belief propagation model requires that we take into account the structure of the argument rather than just the set of premises. Finally, Algorithm 4 invokes the computation of all the agents' trust and beliefs over the other agents and on the knowledge, inference rules, and the conclusions that can be derived from them.

To deal with trust propagation, a typical implementation for *combine* is:

$$combine(\{nd_1, nd_2, ..., nd_n\}) = nd_1 \otimes^{tr} nd_2 \otimes^{tr} ... \otimes^{tr} nd_n.$$

A typical implementation for *fuse* is:

$$fuse(\{nd_1, nd_2, ..., nd_n\}) = nd_1 \oplus^{tr} nd_2 \oplus^{tr} ... \oplus^{tr} nd_n.$$

Other implementations of *combine* and *fuse* can be constructed depending on the underlying trust model or the belief model being used. For example, in TidalTrust, we will have a breadth-first style implementation and use Algorithm 10 for trust computation:

$$\begin{aligned} & fuse(\{tr(Ag_i, Ag_k) | \langle Ag_i, Ag_k \rangle \in \mathcal{T}\} \cup \{tr(Ag_k, ag_j) | \langle Ag_i, Ag_k \rangle \in \mathcal{T}\}) \\ = \quad & \frac{\Sigma_{(Ag_i, Ag_k) \in \mathcal{T} \text{ and } tr(Ag_i, Ag_k) \geq \theta} tr(Ag_i, Ag_k) tr(Ag_k, Ag_j)}{\Sigma_{(Ag_i, Ag_k) \in \mathcal{T} \text{ and } tr(Ag_i, Ag_k) > \theta} tr(Ag_i, Ag_k)}. \end{aligned}$$

For belief propagation, a typical implementation for *combine* is:

$$combine(\{nd_1, nd_2, ..., nd_n\}) = nd_1 \otimes^{bel} nd_2 \otimes^{bel} ... \otimes^{bel} nd_n.$$

and a typical implementation for *fuse* is:

$$fuse(\{nd_1, nd_2, ..., nd_n\}) = nd_1 \oplus^{bel} nd_2 \oplus^{bel} ... \oplus^{bel} nd_n.$$

Other implementations of *combine* and *fuse* can be constructed depending on the underlying model of belief.

Proposition 10 *Let W be the number of arguments in the argumentation graph G. Algorithm 4, trustBelPropagate computes the beliefs of an agent Ag_i on all arguments in the size of G and the number of arguments, namely $O(W \cdot |G|)$.*

Proof: With the help of data structure $trustBel[Ag_i, n]$, we have a dynamic programming-style computation of trust and belief. The algorithm just needs to investigate each edge in G once, and compute the relevant trust or belief on

demand. Once the value has been computed, it can be looked up in the data structure *trustBel*. If we implement this with an appropriate hash table, we can guarantee the look-up for the computed trust and belief value in $O(1)$, resulting in a total complexity in terms of $O(W \cdot |G|)$. □

As G is polynomial in the size of the set of premises and inference rules, we can compute the trust in all arguments with this complexity. As before number of arguments can still be exponential in the number of premises and rules, but the way we implement the trust and belief computation through the compact argumentation graph suggests various possibility to reduce the complexity in our future work. Note that with Algorithm 4, we compute trust values and belief values of the nodes in G with complexity in the size of G. In this way, we may be able to avoid constructing undefendable arguments in terms of the belief levels of the components of arguments. This is another line of future research.

Algorithm 3: *initTrustMatrix*(G): Initialize the trust matrix

Input: G: a shared graph structure; $\mathcal{T}$: trust network; Σ: knowledge base; Δ: rule base
for *each agent* Ag_i **do**
  for *each argument* $arg \in G$ **do**
    $trustBel[Ag_i, arg] \leftarrow NIL$;
  end
  for *each premise and inference node* $nd \in G$ **do**
    $trustBel[Ag_i, nd] \leftarrow NIL$;
  end
end

Algorithm 4: trustBelPropagate(G, computeBel, combine, fuse): Trust and belief propagation in argumentation

Input: G: a shared graph structure; *computeBel*: a customized function to compute trust and belief; *combine*: a customized function to combine (AND-input) multiple sources of trust and belief together; *fuse*: a customized function to fuse (OR-input) multiple sources of trust and belief together
InitTrustMatrix(G) ;
for *each agent* Ag_i **do**
  for *each argument* $arg \in G$ **do**
    $trustBel[Ag_i, arg] \leftarrow computeBel(G, Ag_i, arg, combine, fuse)$;
  end
end

Algorithm 5: *computeBel*$(G, Ag_i, arg, combine, fuse)$: Compute the trust or belief of Ag_i over *node* in argument *arg*

Input: G: a shared graph structure; Ag_i: an agent; *combine*: a customized function to combine (AND-input) multiple sources of trust and belief together; *fuse*: a customized function to fuse (OR-input) multiple sources of trust and belief together
for *each node* $nd \in arg$ **do**
  $bel[nd] \leftarrow computeTrustBel(G, Ag_i, nd, combine, fuse)$;
end
$trustBel[Ag_i, arg] \leftarrow combine(\{bel[nd_i] | nd_i \in arg\})$;
return $trustBel[Ag_i, arg]$;

Algorithm 6: *computeBel*$(G, Ag_i, arg, combine, fuse)$: Compute the trust or belief of Ag_i over *node* in argument *arg* taking into account the argument structure

Input: G: a shared graph structure; Ag_i: an agent; *combine*: a customized function to combine (AND-input) multiple sources of trust and belief together; *fuse*: a customized function to fuse (OR-input) multiple sources of trust and belief together

$trustBel[Ag_i, arg] \leftarrow computeTrustBel(G, Ag_i, arg, conclusion(arg), combine, fuse)$;

return $trustBel[Ag_i, arg]$;

Algorithm 7: *computeTrustBel*$(G, Ag_i, arg, node, combine, fuse)$: Compute the trust or belief of Ag_i over *node* in argument *arg* taking into account the argument structure

Input: G: a shared graph structure; Ag_i: an agent; *combine*: a customized function to combine (AND-input) multiple sources of trust and belief together; *fuse*: a customized function to fuse (OR-input) multiple sources of trust and belief together

if *node is a premise node or inference node* **then**
    return $computeTrustBel(G, Ag_i, node, combine, fuse)$;
end
if *node is a conclusion node* **then**
    $\gamma \leftarrow$ inference rule connecting to *node* in *arg*;
    $result[\gamma] \leftarrow computeTrustBel(G, Ag_i, \gamma, cominbe, fuse)$;
    for *each premise* pm_k *connecting to* γ *in arg* **do**
        $result[pm_k] \leftarrow comptueBel(G, Ag_i, arg, pm_k, combine, fuse)$;
    end
    return $combine(\{result[pm_k]\} \cup \{result[\gamma]\})$;
end

Algorithm 8: *computeTrustBel*$(G, Ag_i, node, combine, fuse)$: Compute the trust or belief of Ag_i over *node* in argument *arg*

Input: G: a shared graph structure; Ag_i: an agent; *combine*: a customized function to combine (AND-input) multiple sources of trust and belief together; *fuse*: a customized function to fuse (OR-input) multiple sources of trust and belief together

if Ag_i *holds node* **then**
    return $Bel_i(node)$;
end
if $trustBel[Ag_i, node] \neq NIL$ **then**
    return $trustBel[Ag_i, node]$;
end
for *each agent* Ag_j *who holds node* **do**
    $res \leftarrow computeTrust(Ag_i, Ag_j)$;
    if $res \neq NIL$ **then**
        $bel[Ag_j] \leftarrow combine(res, Bel_j(node))$;
    end
end
$trustBel[Ag_i, node] \leftarrow fuse(\{bel[Ag_j] | Ag_j \text{ holds } node\})$;
return $trustBel[Ag_i, node]$;

Algorithm 9: *computeTrust*$(G, Ag_i, Ag_j, combine, fuse)$: Compute the trust or belief of Ag_i over *node* in argument *arg* in a Depth-First manner

Input: G: a shared graph structure; Ag_i: an agent; *combine*: a customized function to combine (AND-input) multiple sources of trust and belief together; *fuse*: a customized function to fuse (OR-input) multiple sources of trust and belief together

if $trust[Ag_i, Ag_j] \neq NIL$ **then**
  return $trust[Ag_i, Ag_j]$;
end
if $\langle ag_i, Ag_j \rangle \in \mathcal{T}$ **then**
  return $tr(ag_i, ag_j)$;
else if *there is no* Ag_k *such that* $\langle Ag_i, Ag_k \rangle \in \mathcal{T}$ **then**
  return *NIL*;
else
  for *each* Ag_k *such that* $\langle Ag_i, Ag_k \rangle \in \mathcal{T}$ **do**
    $res \leftarrow computeTrust(Ag_k, Ag_j)$;
    if $res \neq NIL$ **then**
      $result[ag_k] \leftarrow combine(tr(ag_i, Ag_k), res)$;
    end
  end
  $trust[Ag_i, Ag_j] \leftarrow fuse(\{result[ag_k] | \langle Ag_i, Ag_k \rangle \in \mathcal{T}\})$;
  return $trust[Ag_i, Ag_j]$;
end

Algorithm 10: *computeTrust*$(G, Ag_i, Ag_j, combine, fuse)$: Compute the trust or belief of Ag_i over *node* in argument *arg* in a Breadth-first manner

Input: G: a shared graph structure; Ag_i: an agent; *combine*: a customized function to combine (AND-input) multiple sources of trust and belief together; *fuse*: a customized function to fuse (OR-input) multiple sources of trust and belief together

if $trust[Ag_i, Ag_j] \neq NIL$ **then**
  return $trust[Ag_i, Ag_j]$;
end
if $\langle ag_i, Ag_j \rangle \in \mathcal{T}$ **then**
  return $tr(ag_i, ag_j)$;
else if *there is no* Ag_k *such that* $\langle Ag_i, Ag_k \rangle \in \mathcal{T}$ **then**
  return *NIL*;
else
  for *each* Ag_k *such that* $\langle Ag_i, Ag_k \rangle \in \mathcal{T}$ **do**
    $result[Ag_k, Ag_k] \leftarrow computeTrust(Ag_k, Ag_j)$;
  end
  $trust[Ag_i, Ag_j] \leftarrow fuse(\{tr(Ag_i, Ag_k) | \langle Ag_i, Ag_k \rangle \in \mathcal{T}\} \cup \{result[Ag_k, ag_j] | \langle Ag_i, Ag_k \rangle \in \mathcal{T}\})$;
  return $trust[Ag_i, Ag_j]$;
end

9 Discussion

As we mentioned in the introduction, there is some consensus in the trust literature that provenance, the relating of information to its source, plays an important role in reasoning about trust. This connection is made explicitly in [30] and [32], but can be found implicitly in, for example, work like [51] which identifies internet authorities by their links with other nodes, and [2] which constructs reputations for individuals from their contributions to online media. Since argumentation explicitly captures the data used in reasoning, it seems a natural tool for dealing with provenance, and the work we describe here develops a system of argumentation that not only records what data is used but also ties this data back to the individuals who provided it.

The provision of such a model — a graphical model which goes beyond existing graphical accounts of Dung-like argumentation — is one part of the contribution of this work. The other part is the provision of algorithms for constructing the graphical model, which identifies the arguments, and then propagates trust and belief values through the resulting network, quantifying the conclusions of the arguments. Though this particular combination of argumentation and trust is novel, the idea of combining trust and argumentation is not. Four lines of work on trust and argumentation that are complementary to ours include those of Harwood [38, 39], Matt *at al.*, Hunter [40], [58], and Stranders [84]. In addition, argumentation has been used in the past to reason about risk [47, 60], a subject closely related to trust though the cited work looked at risk of carcinogenicity given chemical structure rather than risk due to untrustworthiness.

Harwood [38, 39] takes a strong position against the use of numerical estimates of trust, instead constructing an elegant and purely symbolic approach. In his work, arguments are constructed about the reasonableness or otherwise of trusting individuals. Given such a set of arguments, it is then possible to impose a semantics reminiscent of Dung's which identifies not which arguments are acceptable, but which are to be trusted and distrusted. Hunter [40], meanwhile, is not interested in explicitly modelling trust, but is concerned with reasoning about the proponents of arguments. He does this through *meta-level* reasoning about arguments, that is, he constructs arguments about how good arguments are, where the grounds of these meta-arguments include information about the agents who put forward the object-level arguments.

Both Harwood and Hunter take strictly symbolic approaches that are not concerned with numerical degrees of trust. In contrast, Matt *at al.* [58] are, like us, interested in combining numerical data on trustworthiness with arguments. However, where we take the trust ratings for granted — or, alternatively, assume that we will employ well-founded means of establishing trust ratings from data — Matt *at al.* describe how such trust ratings may be constructed. The closest of these four lines of work to ours is the work of Stranders [84] who use the argumentation system from [5] to handle trust. This is a less general approach than ours, being tied to a possibilistic notion of trust, but they extend [5] to include vague propositions. These are propositions that, unlike those we deal with, are not just true or false, but have fuzzy degrees of truth, giving the system greater representational power.

All of the systems we have just mentioned suggest future directions for our work, and we have already mentioned areas of future work in connection with the computation of arguments and the propagation of values. Another important area of future work is to establish how to use the trust values we compute within argumentation. In Section 6, we discussed using trust values to provide strengths for arguments, following [4] in modifying defeat so that the defeater has to be stronger (in this case more trusted) than the defeatee. There are other ways to use trust values. One natural use for trust values is to allow agents to specify a trust threshold — all information from agents trusted less than the threshold is discarded. A less obvious use, but one we find intriguing, is the use of a trust budget (akin to the inconsistency budget of [24]). This approach would allow an agent to decide on a maximum amount of distrust it was prepared to tolerate across all the agents who provided it with information pertaining to a single argument, and obtain different sets of conclusions for different budget levels. Such an approach would allow the agent to search for particular conclusions it was interested in (plans to achieve particular goals for example) and determine exactly how trusting it had to be to get the conclusions it wanted.

Finally, we need to acknowledge that both the model of trust that we have used, and the model of agent beliefs, are very simple. Trust is represented as a simple numerical measure, in most cases a single number, and a 3-tuple in the case of subjective logic. An agent's beliefs are modelled by attaching similar numerical measures to formulae. While most work on argumentation takes a similarly simple view of belief, this does not mean that argumentation can only work with such a simple model. As an example, in [71] we described a system of argumentation that used a multi-model logic, in particular the belief/desire/intention logic of Rao and Georgeff [74, 75]. The model makes use

of multi-context systems [31] to capture the different modalities, and we can use the same techniques to capture the modal notions of trust and belief from [15, 57].

10 Conclusions

In this paper, we have introduced a general approach to combining argumentation with information about trust in a way that allows us to consider information about the degree to which other agents are trusted when reasoning with information obtained from them. The approach we introduce makes no commitment to a specific mathematical approach to computing trust — it can be instantiated with any of a number of numerical systems for propagating trust information that have been proposed in the literature. In particular, we discussed how to use possibility values, as suggested by [79], subjective logic, as suggested by [90], and the TidalTrust approach [50]. In addition to introducing this system, we explored a number of the basic properties of the system, and illustrated its use on an example from [50], showing how our system obtains the same solution as [50]. We also discussed in detail some of the computation required by our approach. In particular, we described how to construct arguments, and how to propagate trust and belief values through the graphical representation of arguments that we use.

Future directions, as mentioned above, include detailed examination and comparison of different approaches to propagating and combining qualitative and quantitative values for trust and belief. Long term work concerns development of an interactive tool to support users making the types of decision faced by John in our extended example.

Acknowledgement

Research was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

We would like to thanks the reviewers. Their many helpful suggestions have greatly improved this paper.

References

[1] Z. Abrams, R. McGrew, and S. Plotkin. Keeping peers honest in EigenTrust. In *Proceedings of the 2nd Workshop on the Economics of Peer-to-Peer Systems*, 2004.

[2] B. T. Adler and L. de Alfaro. A content-driven reputation system for the Wikipedia. In *Proceedings of the 16th International World Wide Web Conference*, pages 261–270, Banff, Alberta, May 2007.

[3] L. Amgoud and C. Cayrol. On the acceptability of arguments in preference-based argumentation. In *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 1–7, San Francisco, CA, 1998. Morgan Kaufmann.

[4] L. Amgoud and C. Cayrol. A reasoning model based on the production of acceptable arguments. *Annals of Mathematics and Artificial Intelligence*, 34(1-3):197–215, 2002.

[5] L. Amgoud and H. Prade. Using arguments for making decisions. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pages 10–17, Banff, Canada, July 2004.

[6] D. Artz and Y. Gil. A survey of trust in computer science and the semantic web. *Journal of Web Semantics*, 5(2):58–71, June 2007.

[7] T. J. M. Bench-Capon. Value based argumentation frameworks. *CoRR*, cs.AI/0207059, 2002.

[8] S. Bistarelli and F. Santini. A common computational framework for semiring-based argumentation systems. In *Proceedings of the 19th European Conference on Artificial Intelligence*, Lisbon, Portugal, August 2010.

[9] J. Bourdon, G. Feuillade, A. Herzig, and E. Lorini. Trust in complex actions. In D. M. Gabbay and L. van der Torre, editors, *Logics in Security*, Copenhagen, Denmark, 2010.

[10] C. S. Carr. Computer-supported collaborative argumentation: Supporting problem-based learning in legal education. In *Proceedings of the 2nd European Conference on Computer-Supported Collaborative Learning*, Bergen, Norway, 2003.

[11] C. Castelfranchi and R. Falcone. Trust is much more than subjective probability: Mental components and sources of trust. In *Proceedings of the 33rd Hawaii International Conference on System Science*, Maui, Hawai'i, January 2000. IEEE Computer Society.

[12] C. Cayrol and M.-C. Lagasquie-Schiex. Graduality in argumentation. *Journal of Artificial Intelligence Research*, 23:245–297, 2005.

[13] P. Dandekar, A. Goel, R. Govindan, and I. Post. Liquidity in credit networks: A little trust goes a long way. Technical report, Department of Management Science and Engineering, Stanford University, 2010.

[14] D. B. DeFigueiredo and E. T. Barr. TrustDavis: A non-exploitable online reputation system. In *Proceedings of the 7th IEEE International Conference on E-Commerce Technology*, pages 274–283, 2005.

[15] R. Demolombe and E. Lorini. A logical account of trust in information sources. In *Proceedings of the 11th International Workshop on Trust in Agent Societies*, Estoril, Portugal, may 2008.

[16] R. Diestel. *Graph Theory*. Springer-Verlag, Heidelberg, 4th edition, 2010.

[17] X. L. Dong, L. Berti-Equille, and D. Srivastava. Integrating conflicting data: The role of source dependence. In *Proceedings of the 35th International Conference on Very Large Databases*, Lyon, France, August 2009.

[18] D. Dubois and H. Prade. Default reasoning and possibility theory. *Artificial Intelligence*, 35:243–257, 1988.

[19] D. Dubois and H. Prade. An introduction to possibilistic and fuzzy logics. In P. Smets, E. H. Mamdani, D. Dubois, and H. Prade, editors, *Non-Standard Logics for Automated Reasoning*, pages 287–313. Academic Press, London, UK, 1988.

[20] D. Dubois and H. Prade. *Possibility Theory: An Approach to Computerized Processing of Uncertainty*. Plenum Press, New York, NY, 1988.

[21] D. Dubois and H. Prade. Consonant approximations of belief functions. *International Journal of Approximate Reasoning*, 4:419–449, 1990.

[22] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and *n*-person games. *Artificial Intelligence*, 77:321–357, 1995.

[23] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–358, 1995.

[24] P. E. Dunne, A. Hunter, P. McBurney, S. Parsons, and M. Wooldridge. Weighted argument systems: Basic definitions, algorithms, and complexity results. *Artificial Intelligence*, (in press).

[25] R. Falcone and C. Castelfranchi. Social trust: A cognitive approach. In C. Castelfranchi and Y. Tan, editors, *Trust and Deception in Virtual Societies*, pages 55–99. Kluwer Academic Publishers, 2001.

[26] `http://trust.mindswap.org/FilmTrust/`.

[27] R. Francone and C. Castelfranchi. Transitivity in trust: A discussed property. In *Proceedings of the Undicesimo Workshop Nazionale "Dagli Oggetti agli Agenti"*, Rimini, September 2010.

[28] D. Gambetta. Can we trust them? In D. Gambetta, editor, *Trust: Making and breaking cooperative relations*, pages 213–238. Blackwell, Oxford, UK, 1990.

[29] A. J. Garcia and G. R. Simari. Defeasible logic programming: an argumentative approach. *Theory and Practice of Logic Programming*, 4(2):95–138, 2004.

[30] F. Geerts, A. Kementsiedtsidis, and D. Milano. Mondrian: Annotating and querying databases through colors and blocks. In *Proceedings of the 22nd International Conference on Data Engineering*, pages 82–82, Atlanta, April 2006.

[31] F. Giunchiglia and L. Serafini. Multilanguage hierarchical logics (or: How we can do without modal logics). *Artificial Intelligence*, 65:29–70, 1994.

[32] J. Golbeck. Combining provenance with trust in social networks for semantic web content filtering. In *Proceedings of the International Provenance and Annotation Workshop*, Chicago, Illinois, May 2006.

[33] J. Golbeck. Generating predictive movie recommendations from trust in social networks. In *Proceedings of the Fourth International Conference on Trust Management*, Pisa, Italy, May 2006.

[34] J. Golbeck and J. Hendler. Filmtrust: Movie recommendations using trust in web-based social networks. In *Proceedings of the IEEE Consumer Communications and Networking Conference*, 2006.

[35] T. Grandison and M. Sloman. A survey of trust in internet applications. *IEEE Communications Surveys and Tutorials*, 4(4):2–16, 2000.

[36] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *Proceedings of the 13th International Conference on the World Wide Web*, pages 403–412, 2004.

[37] C-W Hang, Y. Wang, and M. P. Singh. Operators for propagating trust and their evaluation in social networks. In *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems*, Budapest, Hungary, 2009.

[38] W. T. Harwood, J. A.Clark, and J. L. Jacob. Networks of trust and distrust: Towards logical reputation systems. In *Logics in Security*, 2010.

[39] W. T. Harwood, J. A.Clark, and J. L. Jacob. A perspective on trust, security and autonomous systems. In *Proceedings of the New Security Paradigms Workshop*, Concord, MA, 2010.

[40] A. Hunter. Reasoning about the appropriateness of propoents for arguments. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, Chicago, Illinois, July 2008.

[41] E. T. Jaynes. *Probability Theory: The Logic of Science*. Cambridge University Press, Cambridge, UK, 2003.

[42] A. Jøsang. Trust-based decision making for electronic transactions. In *Proceedings of the Fourth Nordic Workshop on Secure Computer Systems (NORDSEC'99)*, 1999.

[43] A. Jøsang. A logic for uncertain probabilities. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 9:279–311, June 2001.

[44] A. Jøsang, E. Gray, and M. Kinateder. Simplification and analysis of transitive trust networks. *Web Intelligence and Agent Systems*, 4(2):139–161, 2006.

[45] A. Jøsang, R. Hayward, and S. Pope. Trust network analysis with subjective logic. In *Proceedings of the 29th Australasian Computer Society Conference*, Hobart, January 2006.

[46] A. Jøsang, C. Keser, and T. Dimitrakos. Can we manage trust? In *Proceedings of the 3rd International Conference on Trust Management*, Paris, May 2005.

[47] P. N. Judson, J. Fox, and P. J. Krause. Using new reasoning technology in chemical information systems. *Journal of Chemical Information and Computer Sciences*, 36:621–624, 1996.

[48] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The EigenTrust algorithm for reputation management in P2P networks. In *Proceedings of the 12th World Wide Web Conference*, May 2004.

[49] N. Karacapilidis and D. Papadias. Computer-supported argumentation and collaborative decision making: The Hermes system. *Information Systems*, 26(4):259–277, June 2001.

[50] Y. Katz and J. Golbeck. Social network-based trust in prioritzed default logic. In *Proceedings of the 21st National Conference on Artificial Intelligence*, 2006.

[51] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, pages 604–632, 1999.

[52] G. J. Klir and B. Yuan. *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Prentice Hall PTR, NJ, USA, 1st edition, May 1995.

[53] Y. Kuter and J. Golbeck. SUNNY: A new algorithm for trust inference in social networks using probabilistic confidence models. In *Proceedings of the 22nd National Conference on Artificial Intelligence*, 2007.

[54] J. Lang, M. Spear, and S. F. Wu. Social manipulation of online recommender systems. In *Proceedings of the 2nd International Conference on Social Informatics*, Laxenburg, Austria, 2010.

[55] G. Li, Y. Wang, and M. A. Orgun. Optimal social trust path selection in complex social networks. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, pages 1391–1398, Atlanta, GA., 2010.

[56] C-J. Liau. Belief, information acquisition, and trust in multi-agent systems — a modal logic formulation. *Artificial Intelligence*, 149:31–60, 2003.

[57] E. Lorini and R. Demolombe. from binary trust to graded trust in information sources: a logical perspective. In R. Falcone, S. K. Barber, J. Sabater-Mir, and M. P. Singh, editors, *Trust in Agent Societies, 11th International Workshop, Revised Selected and Invited Papers*, number 5396 in Lecture Notes in Computer Science, pages 205–225. Springer Verlag, Berlin, Germany, 2008.

[58] P-A. Matt, M. Morge, and F. Toni. Combining statistics and arguments to compute trust. In W. van der Hoek, G. Kaminka, Y Lespérance, M. Luck, and S. Sen, editors, *Proceedings of the 9th International Conference on Autonomous Agents and Multiagents Systems*, pages 209–216, Toronto, Canada, May 2010.

[59] P. McBurney and S. Parsons. Tenacious tortoises: A formalism for argument over rules of inference. In *Proceedings of the ECAI Workshop on Computational Dialectics*, Berlin, 2000.

[60] P. McBurney and S. Parsons. Dialectical argumentation for reasoning about chemical carcinogenicity. *Logic Journal of the IGPL*, 9(2):191–203, 2001.

[61] D. H. McKnight and N. L. Chervany. The meanings of trust. Working Paper 96-04, Carlson School of Management, University of Minnesota, 1996.

[62] Analysis of social voting patterns on Digg. K. lerman and a. galstyan. In *Proceedings of the 1st Workshop on Online Social Networks*, Seattle, August 2008.

[63] D. Olmedilla, O. Rana, B. Matthews, and W. Nejdl. Security and trust issues in semantic grids. In *Proceedings of the Dagstuhl Seminar, Semantic Grid: The converegance of technologies*, volume 05271, 2005.

[64] N. Oren, T. Norman, and A. Preece. Subjective logic and arguing with evidence. *Artificial Intelligence*, 171(10–15):838–854, 2007.

[65] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the Web. Technical Report 1999-66, Stanford InfoLab, 1999.

[66] M. Paolucci, D. Suthers, and A. Weiner. Belvedere: Stimulating students' critical discussion. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*, 1995.

[67] S. Parsons and P. McBurney. Argumentation-based communication between agents. In M.-P. Huget, editor, *Agent Communication Languages*. Springer Verlag, Berlin, 2003.

[68] S. Parsons and P. McBurney. Argumentation-based dialogues for agent coordination. *Group Decision and Negotiation*, 12(5):415–439, 2003.

[69] S. Parsons, P. McBurney, and E. Sklar. Reasoning about trust using argumentation: A position paper. In *Proceedings of the 7th Workshop on Argumentation in Multiagent Systems*, Toronto, Canada, May 2010.

[70] S. Parsons, P. McBurney, and E. Sklar. Using argumentation to reason with and about trust. In *Proceedings of the 8th Workshop on Argumentation in Multiagent Systems*, Taipei, Taiwan, May 2011.

[71] S. Parsons, C. Sierra, and N. R. Jennings. Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8(3):261—292, 1998.

[72] H. Prakken. Coherence and flexibility in dialogue games for argumentation. *Journal of Logic and Computation*, 15(6):1009–1040, 2005.

[73] I. Rahwan and G. R. Simari, editors. *Argumentation in Artificial Intelligence*. Springer Verlag, Berlin, Germany, 2009.

[74] A. Rao and M. Georgeff. Asymmetry thesis and side-effect problems in linear time and branching time intention logics. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, pages 498–504, 1991.

[75] A. S. Rao and M. P. Georgeff. Formal Models and Decision Procedures for Multi-Agent Systems. Technical Note 61, Australian Artificial Intelligence Institute, 1995.

[76] C. Reed, D. Walton, and F. Macagno. Argument diagramming in logic, law and artificial intelligence. *Knowledge Engineering Review*, 22(1):87–109, 2007.

[77] P. Resnick and R. Zeckhauser. Trust among strangers in internet transactions: Empirical analysis of eBay's reputation system. In M. R. Baye, editor, *The Economics of the Internet and E-Commerce*, pages 127–157. Elsevier Science, Amsterdam, 2002.

[78] P. Resnick, R. Zeckhauser, E. Friedman, and K. Kuwabara. Reputation systems: Facilitating trust in internet interactions. *Communications of the ACM*, 43:45–48, 2000.

[79] M. Richardson, R. Agrawal, and P. Domingos. Trust management for the semantic web. In *Proceedings of the 2nd International Semantic Web Conference*, pages 351–368, 2003.

[80] G. Rowe, F. Macagno, C. Reed, and D. Walton. Araucaria as a tool for diagramming arguments in teaching and studying philsophy. *Teaching Philosophy*, pages 111–124, 2006.

[81] J. Sabater and C. Sierra. Review on computational trustand reputation models. *AI Review*, 23(1):33–60, September 2005.

[82] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, NJ, 1976.

[83] P. Smets. Belief functions. In P. Smets, E. H. Mamdani, D. Dubois, and H. Prade, editors, *Non-Standard Logics for Automated Reasoning*, pages 253–275. Academic Press, London, UK, 1988.

[84] R. Stranders, M. de Weerdt, and C. Witteveen. Fuzzy argumentation for trust. In F. Sadri and K. Satoh, editors, *Proceedings of the Eighth Workshop on Computational Logic in Multi-Agent Systems*, volume 5056 of *Lecture Notes in Computer Science*, pages 214–230. Springer Verlag, 2008.

[85] P. Sztompka. *Trust: A Sociological Theory*. Cambridge University Press, Cambridge, UK, 1999.

[86] Y. Tang, K. Cai, E. Sklar, P. McBurney, and S. Parsons. A system of argumentation for reasoning about trust. In *Proceedings of the 8th European Workshop on Multi-Agent Systems*, Paris, France, December 2010.

[87] C-Y Teng, D. Lauterbach, and L. Adamic. I rate you. You rate me. Should we do so publicly? In *Proceedings of the 3rd Workshop on Online Social Networks*, Boston, June 2010.

[88] T. van Gelder. The rationale for Rationale™. *Law, Probability and Risk*, 6:23–42, 2007.

[89] R. Walton, C. Gierl, H. Mistry, M. P. Vessey, and J. Fox. Evaluation of computer support for prescribing (CAPSULE) using simulated cases. *British Medical Journal*, 315:791–795, 1997.

[90] Y. Wang and M. P. Singh. Trust representation and aggregation in a distributed agent system. In *Proceedings of the 21st National Conference on Artificial Intelligence*, 2006.

[91] S. Ye and S. F. Wu. Measuring message propagation and social influence on Twitter.com. In *Proceedings of the 2nd International Conference on Social Informatics*, Laxenburg, Austria, 2010.

[92] X. Yin, J. Han, and P. S. Yu. Truth discovery with multiple conflicting information providers on the web. In *Proceedings of the Conference on Knowledge and Data Discovery*, 2007.

[93] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[94] L. A. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1:1–28, 1978.

[95] C. Zhang. Co-operation under uncertainty in distributed expert systems. *Artificial Intelligence*, 56:21–69, 1992.

[96] S. Zhong, J. Chen, and Y. R. Yang. Sprite: A simple cheat-proof, credit-based system for mobile ad-hoc networks. In *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*, 2003.