In Proceedings of the Eighth European Workshop on Multiagent Systems (EUMAS), Paris, France, December 2010.

A system of argumentation for reasoning about trust

Yuqing Tang[1], Kai Cai[1], Elizabeth Sklar[1,2], Peter McBurney[3], and Simon Parsons[1,2]

[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 Computer & Information Science, Brooklyn College,
City University of New York, 2900 Bedford Avenue, Brooklyn, NY 11210 USA
{sklar,parsons}@sci.brooklyn.cuny.edu
[3] Department of Computer Science, University of Liverpool,
Ashton Building, Ashton Street, Liverpool, L69 3BX, United Kingdom
mcburney@liverpool.ac.uk

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 about trust. We explore some of the simple properties of the system and we illustrate its application on a benchmark problem 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 [4, 15]. Thus, for example, we see work on trust in peer-to-peer networks, including the EigenTrust algorithm [18] — a variant of PageRank [24] 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 — and the work in [1] that prevents peers manipulating their trust values to get preferential downloads. [33] 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 [27], on mechanisms to determine which sources to trust when faced with multiple conflicting sources [32], and mechanisms for identifying which individuals to trust based on their past activity [2]. One interesting development is the idea of having individuals indemnify each other by placing some form of financial guarantee on transactions that others enter into [6, 7].

Trust is an especially important issue from the perspective of autonomous agents and multiagent systems [30]. 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 [29].

In such work it is common to assume that agents 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 the transitivity of trust relations [28, 31] and more complex relationships like "co-citation" [16] have been studied, and in many cases empirically validated [16, 19, 20].

In a previous paper [25], we suggested that, given the role that provenance plays in trust [12, 13], argumentation — which tracks the origin of data used in reasoning — might play a role. [21] also argues for the utility of argumentation in the context of reasoning about trust. In this paper we develop a general system of argumentation that can represent trust information, and be used in combination with a trust network. 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 used as premises in the arguments. We present one instantiation of the system, and demonstrate that it gives intuitively appealing results on an illustrative example.

2 Background

2.1 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 [8]. 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 *hypergraphs*, graphs in which (hyper)edges join three or more vertices. Thus if $V = \{v_1, v_2, v_3, v_4, v_5\}$, E might be $\{\{v_1, v_2\}, \{v_3, v_4\}, \{v_1, v_3, v_5\}\}$. 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 edge.

A *path* in an undirected graph G is just a graph $path$ that is a subgraph of G and 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 simply $\langle v_0, v_1, v_2, \ldots, v_k \rangle$. 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 vertices, and if that hypergraph is directed, the first edge in the pair must be one of the initial vertices, and the second must be a terminal vertex of the edge in question.

A *connected* graph is one for which there is a path between any pair of vertices. If the graph is directed, it is connected if there is an undirected path between any 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.

2.2 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. In particular, we are interested in how these agents trust one another. Following the usual presentation (for example [16, 19, 20, 28, 31]), 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*

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

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 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 another member of the group. A directed path between agents in the trust network indicates that one agent indirectly trusts another. 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 on the trust that one agent has in another, so we have:

$$tr : Ags \times Ags \mapsto \Re$$

where tr gives a suitable trust value. For example, we might have:

$$tr^{[0,1]} : Ags \times Ags \mapsto [0, 1]$$

with a value of 0 indicating that there is no trust between the agents in question and a value of 1 indicating the fullest possible degree of trust between the agents. We assume that tr and τ match, so that:

$$\begin{aligned} tr(Ag_i, Ag_j) \neq 0 &\Leftrightarrow (Ag_i, Ag_j) \in \tau \\ tr(Ag_i, Ag_j) = 0 &\Leftrightarrow (Ag_i, Ag_j) \notin \tau \end{aligned}$$

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 directed path $\langle Ag_i, Ag_{i+1}, \ldots Ag_j\rangle$ in the trust network $\mathcal{T}$ then Ag_i* trusts *Ag_j* according to *$\mathcal{T}$*

The notion of trust embodied here is exactly Jøsang's "indirect trust" or "derived trust" [17] and the process of inferring it is what [16] calls "direct propagation".

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

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

for some function $\otimes^{tr}$. Here we follow [31] in using the symbol $\otimes$, to stand for this operation, 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 these two paths are

$$\langle Ag_i, Ag'_{i+1}, \ldots Ag_j\rangle \qquad \langle Ag_i, Ag''_{i+1}, \ldots Ag_j\rangle$$

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:

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

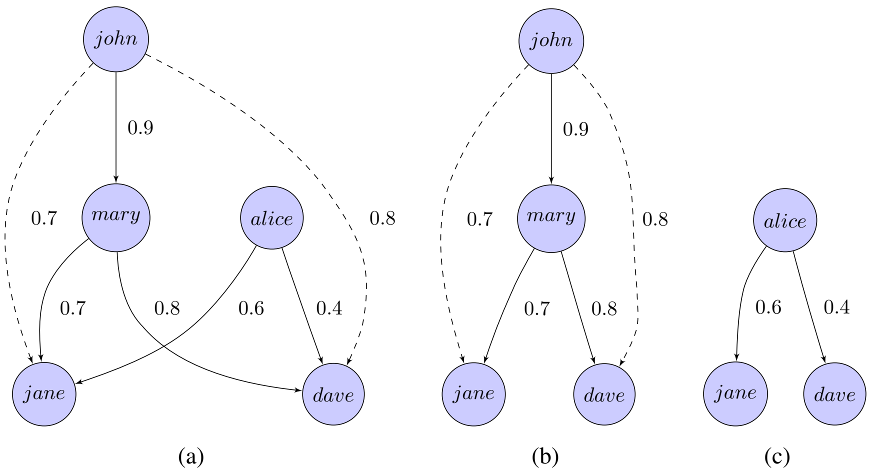


Fig. 1. Example trust graphs. (a) shows a complete graph, (b) shows an agent-centric version from John's point of view, and (c) shows an agent-centric view from Alice's point of view.

again using the standard notation $\oplus$ for a function that combines trust measures along two paths [31]. The literature contains several instantiations of $\otimes$ and $\oplus$. For example, [28] discusses using multiplication or minimum for $\otimes$ and using maximum for $\oplus$, while [19] uses a weighted average that in essence adopts multiplication for $\otimes$ and addition for $\oplus$[4]. [17] and [31] use operators derived from Dempster-Shafer theory.

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 [19] 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 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

[4] Trust values are multiplied along paths and summed across paths, but the result is weighted by the value of the first link in each path, and a threshold is applied.

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 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 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 a link 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 $\mathcal{T}$.

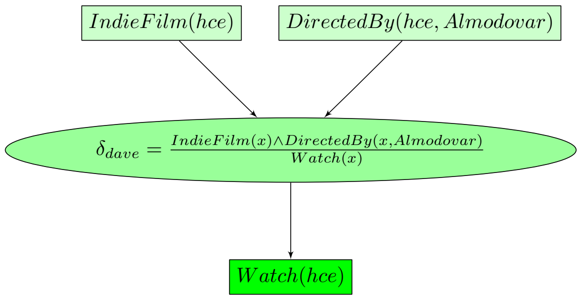
Thus an Ag_i-centric trust network that is derived from a trust network $\mathcal{T}$ exactly identifies the agents that, according to $\mathcal{T}$, Ag_i trusts. As a result, when we consider what Ag_i reasons about, we lose nothing by ignoring the parts of $\mathcal{T}$ that aren't in the Ag_i-centric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

2.3 Argumentation

Now we turn our attention to the structure of the agents. An agent Ag has 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 δ each of which is of the form:

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

where every p_i and c 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 [19], these rules will be much like the normal default rules used in that work. However, for the purposes of this formalism Δ can be any set of inference rules, for example the natural deduction style rules of [22]. We can represent inference using these rules as another graph.



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

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

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 [19]). 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* $\delta' \in E^r$*.*

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. If:

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

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.

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 was more than one rule in H with h as its conclusion, h would not be a leaf.*

Thus an argument does not have any duplicated reasoning that supports its conclusion, and an argument is thus 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 going to often deal with information that is not certain, we will assume that each agent Ag_i has a function which assigns a degree of belief — which might be established as described in [21] — to elements of the logical language it uses for premises and rules:

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

Furthermore, as with trust, we will often use a function that returns degrees of belief between 0 and 1:

$$bel_i^{[0,1]} : \mathcal{L} \mapsto [0, 1]$$

As we build arguments, we need to combine the degrees of belief assigned to premises and rules. For a rule:

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

we assume that there is a conjunction operation, $\otimes^{bel}$, which establishes the degree of belief in c as:

$$bel_i(c) = \big(bel_i(p_1) \otimes^{bel} \cdots \otimes^{bel} bel_i(p_m)\big) \otimes^{bel} bel_i(\delta)$$

In other words, the degree of belief an agent has in the conclusion of an inference rule is the belief it has that all the premises hold and the rule holds.

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

Definition 7. *An argument $\langle h_1, H_1 \rangle$* attacks *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* $h_1 \equiv \neg(p_1 \wedge \ldots \wedge p_n \rightarrow c)$.

Our definition of an argument is similar to both that of Garcia and Simari [11] and that of Prakken [26]. Both [11] and [26] record the inference rules in the argument, with [11] just recording which rules are used while [26], like we do here, keeps the entire structure of the derivation. A minor syntactic difference between our work and [26] is that we define arguments via a reasoning network instead of a recursively through sub-arguments. This is done deliberately to enable the framework to combine arguments and trust relationships uniformly into a network on which we can perform inference by an extension of existing trust propagation mechanisms and which we can use to explain arguments and their relationship to trust in specfic individuals.

3 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 together.

3.1 Trust and belief

We assume that an agent Ag_i is interested in using information not only from its own knowledge base Σ, 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. To do this Ag_i needs to take into account of its degree of trust in Ag_j, and we assume that it does this by taking the trust value that it can compute for Ag_j through the trust network that joins them, as outlined in Section 2.2. Having computed this value, which we will call $tr(\phi)$ 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:

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

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 — that would be correct if the trust values Ag_i has for every Ag_j is simply Ag_i's subjective belief that what Ag_j says is true — which is the notion of trust in [10, 23]. 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, $bel(\phi)$, 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 belief that Ag_i has in anything Ag_j says is true, which itself is computed from the trust that Ag_i has in Ag_j.

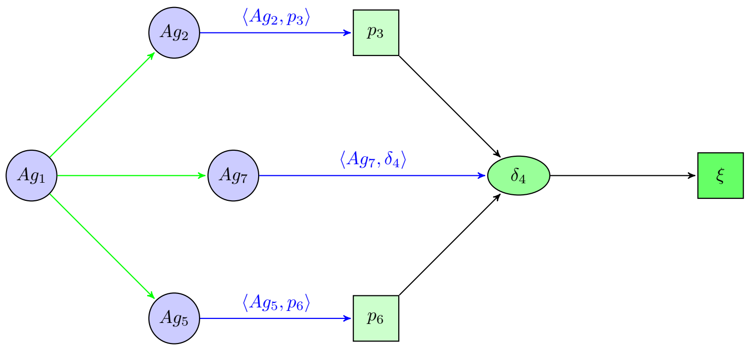

Ag_1
Ag_2
Ag_7
Ag_5
$\langle Ag_2, p_3 \rangle$
$\langle Ag_7, \delta_4 \rangle$
$\langle Ag_5, p_6 \rangle$
p_3
p_6
δ_4
ξ

Fig. 3. A trust-extended proof network

3.2 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, which marries trust networks and argument graphs, to be useful. This concept is formally defined below, and an example is given in Figure 3.

Definition 8. *An* 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 $\delta_i \in \Delta_k$ and $p_j \in \Sigma_k$ respectively.*

An example of a trust-extended proof network is given in Figure 3.

The idea behind the trust-extended proof network is that it relates the premises of an argument to their source. 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 the 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 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, though there may be many leaves of the trust network that aren't 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 of a proof network and its conclusion — with the trust information of a trust-extended proof network:

Definition 9. *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 $Ags = \{Ag_i, \ldots Ag_n\}$, 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.

The last element of our model is a *trust-extended argument graph*. This is a set of trust-extended arguments with the attack relationships between the arguments denoted by labelled links. There are four kinds of attack link, one for each of the kinds of link identified in Definition 7. Such a graph is shown in Figure 4.

This particular graph, which corresponds to the example discussed in the next section, shows two arguments that John can develop. The blue section in the middle of the graph is a trust network which shows the relationship between John, Mary, Dave and Jane. The two green sections on either side 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 relations (the two arguments rebut each other).

4 An Example

In this section we show how our system can capture the example from [19] which considers reasoning in the FilmTrust [9, 14] database. In this 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 [5]:

$$IndieFilm(hce), SpanishFilm(hce), DirectedBy(hce, Almodovar)$$

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

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

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

[5] "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.

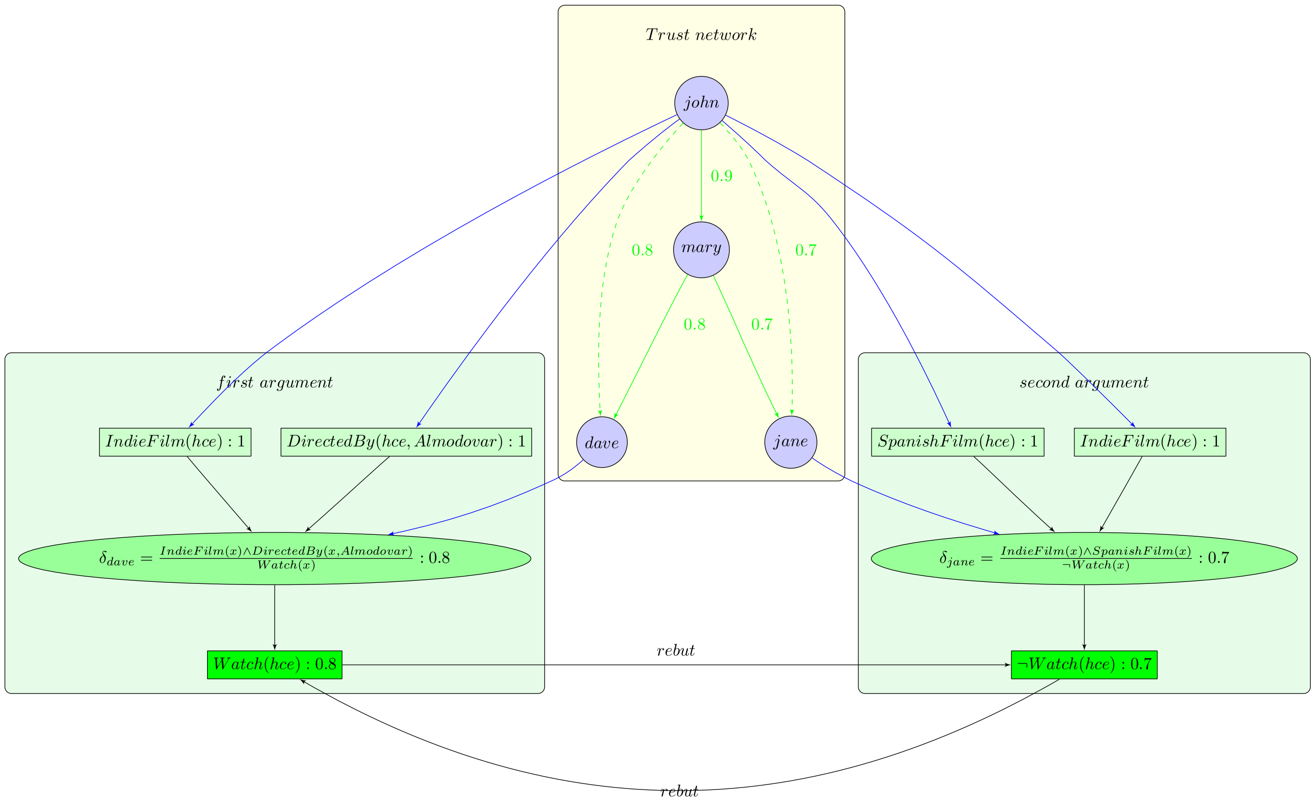

Trust network
john
mary
dave
jane
0.9
0.8
0.7
0.8
0.7
first argument
IndieFilm(hce) : 1
DirectedBy(hce, Almodovar) : 1
$\delta_{dave} = \frac{IndieFilm(x) \wedge DirectedBy(x, Almodovar)}{Watch(x)} : 0.8$
Watch(hce) : 0.8
second argument
SpanishFilm(hce) : 1
IndieFilm(hce) : 1
$\delta_{jane} = \frac{IndieFilm(x) \wedge SpanishFilm(x)}{\neg Watch(x)} : 0.7$
¬Watch(hce) : 0.7
rebut
rebut

Fig. 4. A trust-extended argumentation graph capturing John's view of the film example

From this information, John can construct two simple arguments which are depicted in Figure 4, and which we can gloss 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.
>
> 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 we can use the trust information to determine which argument defeats the other, but to do that we have to make some choices about the trust measure tr that we use. In particular we need to instantiate the combination operation $\otimes^{tr}$. Here we will follow [28] in using minimum for $\otimes^{tr}$, which, along with the trust graph in Figure 1(a) tells us that:

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

and these are the values we see in Figure 4.

Now, these values need to be turned into belief values. For now we choose to handle beliefs using possibility theory [5] — which is basically equivalent to the approach adopted by [3] to handle variable strength arguments. In addition, we choose to 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 (as in [10, 23]). 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:

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

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

$$\begin{aligned} bel_{john}(IndieFilm(hce)) &= 1 \\ bel_{john}(SpanishFilm(hce)) &= 1 \\ bel_{john}(DirectedBy(hce, Almodovar) &= 1 \end{aligned}$$

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

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

from which John may wish to conclude that he should watch the film.

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 [19].

5 Summary

In this paper we have introduced a general approach to combining argumentation with information about trust in a way that allows us to take information about the degree to which other agents are trusted when reasoning with information obtained from them. The approach we introduced makes no commitment to a specific approach to computing trust — it can be instantiated with any of a number of numerical systems for propagating trust information such as [19, 28, 31]. In addition to introducing this system, we explored a number of the basic properties of the system, and illustrated it use, in combination with one specific choice for propagating trust information, on an example from [19], showing how our system obtains the same solution as [19].

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 are grateful to the reviewers for their many helpful comments.

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*, Banff, Alberta, May 2007.
3. 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.
4. 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.
5. S. Benferhat, D. Dubois, and H. Prade. Representing default rules in possibilistic logic. In *Proceedings of the 3rd International Conference on Knowledge Representation and Reasoning*, pages 673–684, San Mateo, CA, 1992. Morgan Kaufmann.
6. 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.
7. D. B. DeFigueiredo and E. T. Barr. TrustDavis: A non-explotable online reputation system. In *Proceedings of the 7th IEEE International Conference on E-Commerce Technology*, 2005.
8. R. Diestel. *Graph Theory*. Springer-Verlag, Heidelberg, 4th edition, 2010.
9. `http://trust.mindswap.org/FilmTrust/`.
10. D. Gambetta. Can we trust them? In D. Gambetta, editor, *Trust: Making and breaking cooperative relations*, pages 213–238. Blackwell, Oxford, UK, 1990.
11. A. J. Garcia and G. R. Simari. Defeasible logic programming: an argumentative approach. *Theory and Practice of Logic Programming*, 4(2):95–138, 2004.
12. 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*, Atlanta, April 2006.
13. 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.

14. 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.
15. T. Grandison and M. Sloman. A survey of trust in internet applications. *IEEE Communications Surveys and Tutorials*, 4(4):2–16, 2000.
16. 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*, 2004.
17. A. Jøsang, C. Keser, and T. Dimitrakos. Can we manage trust? In *Proceedings of the 3rd International Conference on Trust Management*, Paris, May 2005.
18. 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.
19. Y. Katz and J. Golbeck. Social network-based trust in prioritzed default logic. In *Proceedings of the 21st National Conference on Artificial Intelligence*, 2006.
20. 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.
21. 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.
22. 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.
23. 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.
24. 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.
25. S. Parsons, P. McBurney, and E. Sklar. Reasoning about trust using argumentation: A position paper. In *Proceedings of the Workshop on Argumentation in Multiagent Systems*, Toronto, Canada, May 2010.
26. H. Prakken. Coherence and flexibility in dialogue games for argumentation. *Journal of Logic and Computation*, 15(6):1009–1040, 2005.
27. 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.
28. M. Richardson, R. Agrawal, and P. Domingos. Trust management for the semantic web. In *Proceedings of the 2nd International Semantic Web Conference*, 2003.
29. J. Sabater and C. Sierra. Review on computational trustand reputation models. *AI Review*, 23(1):33–60, September 2005.
30. W. T. L. Teacy, G. Chalkiadakis, A. Rogers, and N. R. Jennings. Sequential decision making with untrustworthy service providers. In *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems*, 2008.
31. 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.
32. 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.
33. 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.